Journal of Global Optimization (2005) 33: 313–336
DOI 10.1007/s10898-004-6095-8

Decomposition Methods for Solving Nonconvex Quadratic Programs via Branch and Bound*

RICCARDO CAMBINI and CLAUDIO SODINI*
Department of Statistics and Applied Mathematics, University of Pisa, Via Cosimo Ridolfi 10, 56124 Pisa, Italy (e-mail: cambric@ec.unipi.it, csodini@ec.unipi.it)

(Received 10 February 2003; accepted in revised form 10 November 2004)

Abstract. The aim of this paper is to suggest branch and bound schemes, based on a relaxation of the objective function, to solve nonconvex quadratic programs over a compact polyhedral feasible region. The various schemes are based on different d.c. decomposition methods applied to the quadratic objective function. To improve the tightness of the relaxations, we also suggest solving the relaxed problems with an algorithm based on the so called "optimal level solutions" parametrical approach.

AMS Mathematics Subject Classification (2000). 90C20, 90C26, 90C31.

JEL Classification System (1999). C61, C63.

Key words: Branch and bound, d.c. Decomposition, Quadratic programming

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 [2–4, 6, 12–17, 22, 24, 25]) have been approached in the literature in several ways and in [1, 7, 11, 18, 23] 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 paper, we study various d.c. decompositions of the objective function $f(x) = \frac{1}{2}x^T Ax + c^T x$, which are different from the ones proposed in [1, 7, 18, 23]. For these decompositions we suggest both convex and nonconvex relaxations.

In Section 2 we first study how to decompose the matrix A in the form $A = Q - \sum_{i=1}^{n-v_+(A)} d_i d_i^T$, where $Q \in \Re^{n\times n}$ is positive definite, $d_i \in$

*This paper has been partially supported by M.I.U.R. and C.N.R.

$\Re^n \forall i$ and $\nu_+(A)$ is the number of positive eigenvalues of A. In order to decrease numerical errors and computational complexity, such a decomposition is obtained without the use of eigenvalues and eigenvectors. By using this decomposition the following d.c. form of the objective function is obtained:

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

and a relaxation is given linearizing the quadratic form $\sum_{i=1}^{n-\nu_+(A)}(d_i^T x)^2$. This allows us to suggest a branch and bound scheme based on a partition of the current feasible region. Finally, it is shown that the algorithm proposed in [5] allows us to solve, in the branch and bound scheme, subproblems that have a tighter nonconvex relaxation.

In Section 3 the particular case of box constrained problems is studied. A branch and bound scheme based on the decomposition of matrix A in the form $A = Q - dd^T - \mathrm{diag}(w)$, where Q is positive definite and $\mathrm{diag}(w)$ is a positive semidefinite diagonal matrix with diagonal elements given by the components of vector w, is given by means of the linearization of the quadratic form $x^T\ \mathrm{diag}(w)x$. Some decomposition procedures are proposed and compared with respect to the tightness of the corresponding relaxations.

2. Generic Compact Polyhedral Region

Let us consider the following definition.

DEFINITION 2.1. We define the following quadratic program:

$$P: \begin{cases} \min f(x) = \tfrac{1}{2}x^T Ax + c^T x \\ x \in X = \{x \in \Re^n : Bx \geqslant b\}, \end{cases}$$

where X is a compact polyhedron, $B \in \Re^{m\times n}$, $b \in \Re^m$, $c \in \Re^n$ and $A \in \Re^{n\times n}$ is any symmetric matrix. From now on we will denote also with (ν_+, ν_-, ν_0) the inertia of A, where $\nu_+(A) + \nu_-(A) + \nu_0(A) = n$. In other words, $\nu_+(A)$ is the number of positive eigenvalues of A, $\nu_-(A)$ is the number of negative ones, $\nu_0(A)$ is the algebraic multiplicity of the zero eigenvalue.

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 section is to propose a branch and bound scheme to solve problem P when A is not positive definite.

2.1. PRELIMINARIES

It is well known that the decomposition method of Lagrange (see [10]), based on the "Law of Inertia"[1], provides for any symmetric matrix A a decomposition of the kind $A = Q - \sum_{i=1}^{h} d_i d_i^T$, where Q is positive semidefinite with $\text{rank}(Q) = \nu_+(A), h = \nu_-(A)$ and $d_1, \ldots, d_h$ are linearly independent.

Such a procedure can be slightly modified as described in procedure `ModLagrange` (A,Q,k,$d_1, \ldots, d_k$), in order to obtain a decomposition of the kind

$$A = Q - \sum_{i=1}^{k} d_i d_i^T,$$

where Q is positive definite, $k = \nu_-(A) + \nu_0(A) = n - \nu_+(A)$ and $d_1, \ldots, d_k$ are linearly independent. Note that in this procedure we denote with row[T, r] the r-th row of matrix T.

Procedure **ModLagrange**(*inputs*: A; *outputs:* Q, k, $d_1, \ldots, d_k$)
T:=A; Q:=0; k:=0; used:= [1, 1, ... , 1, 1] $\in \Re^n$;
while $T \neq 0\, d_0$
  if $T[i, i] = 0\ \forall i \in \{1, \ldots, n\}$
  then select $r \in \{1, \ldots, n\}$ *such that* row[T,r] $\neq 0$;
    Q[r,r]:=Q[r,r]+1; T[r,r]:=−1;
  else select $r \in \{1, \ldots, n\}$ *such that* $T[r, r] \neq 0$;
  end if;
  used[r]:=0; v:=row[T,r]; $\alpha := T[r,r]$; $T := T - \frac{1}{\alpha} vv^T$; $Q := Q + \frac{1}{|\alpha|} vv^T$;
  if $\alpha < 0$ *then* k:=k+1; $d_k := \sqrt{\frac{-2}{\alpha}}$ v *end if*;
end do;
for i *from* 1 *to* n *do*
  if used[i]=1 *then* Q[i,i]:=Q[i,i]+1; k:=k+1; $d_k := 0$; $d_k[i] := 1$; *end if*;
end do;
end proc.

[1](The Law of Inertia for symmetric matrices [10].) 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 $1 \leqslant r \leqslant n$ linearly independent vectors such that

$$A = \sum_{i=1}^{r} \alpha_i u_i u_i^T,$$

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 $u_1, \ldots, u_r$.

REMARK 2.1. The `ModLagrange` (A,Q,k,$d_1, \ldots, d_k$) procedure starts initializing a temporary matrix $T := A$, a null matrix Q, a zero index k and a flags vector *used* := $[1, 1, \ldots, 1, 1]$.

The aim of the procedure is to modify matrices T and Q and to create linearly independent vectors $d_1, \ldots, d_k$ so that, at every iterative step, it is $A = Q + T - \sum_{i=1}^{k} d_i d_i^T$, with Q positive semidefinite.

In the various iterations of the while cycle a row of T, say the r-th row, is selected (*used* $[r] := 0$ and $v := row[T, r]$); then, in the light of the method of Lagrange [10], matrices T and Q are updated so that the r-th row and column of T become equal to the null vector and Q increases its rank by 1 ($T := T - \frac{1}{T[r,r]} vv^T$ and $Q := Q + \frac{1}{|T[r,r]|} vv^T$); finally if $T[r, r] < 0$ a new vector d_k is created ($d_k := \sqrt{\frac{-2}{T[r,r]}} v$). It is worth observing that the case of a non-zero matrix T having all the diagonal elements equal to zero is managed in a different way with respect to the method of Lagrange [10].

Notice also that, for the Law of Inertia, the while cycle is iterated exactly $\nu_-(A) + \nu_+(A)$ times until T becomes the null matrix. A decomposition of the kind $A = Q - \sum_{i=1}^{\nu_-(A)} d_i d_i^T$, where Q is positive semidefinite with rank equal to $\nu_-(A) + \nu_+(A)$ and the vectors $d_1, \ldots, d_{\nu-(A)}$ are linearly independent, is then provided.

If $\nu_0(A) = 0$ then Q is positive definite and the procedure stops; otherwise, by scanning the flags vector *used*, $\nu_0(A)$ diagonal elements of Q are updated in order to make it positive definite ($Q[i, i] := Q[i, i] + 1$) and $\nu_0(A)$ more vectors d_k are created ($d_k := 0$ and $d_k[i] := 1$).

As a consequence, the output of the procedure is a decomposition of the kind $A = Q - \sum_{i=1}^{k} d_i d_i^T$, with $k = \nu_-(A) + \nu_0(A)$ and Q is nonsingular and positive definite.

EXAMPLE 2.1. Applying procedure `ModLagrange` (A,Q,k,$d_1, \ldots, d_k$) to

$$A = \begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix}$$

we have:

- T:=A; Q:=0; k:=0; used:=[1,1];
- T[i,i]=0, i=1,2, hence:
  r:=1; used[1]:=0; Q[1,1]:=1; T[1,1]:=−1; v:=[−1, 2]; $\alpha := -1$;

$$T := \begin{bmatrix} -1 & 2 \\ 2 & 0 \end{bmatrix} + \begin{bmatrix} 1 & -2 \\ -2 & 4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 4 \end{bmatrix}$$

$$Q := \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & -2 \\ -2 & 4 \end{bmatrix} = \begin{bmatrix} 2 & -2 \\ -2 & 4 \end{bmatrix}$$

- $\alpha < 0$ hence: k:=1; $d_1^T := \sqrt{2}[-1, 2]$

- T[2,2] $\neq 0$ hence: r:=2; used[2]:=0; v:=[0,4]; $\alpha := 4$;

$$T := \begin{bmatrix} 0 & 0 \\ 0 & 4 \end{bmatrix} - \frac{1}{4}\begin{bmatrix} 0 & 0 \\ 0 & 16 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$
$$Q := \begin{bmatrix} 2 & -2 \\ -2 & 4 \end{bmatrix} + \frac{1}{4}\begin{bmatrix} 0 & 0 \\ 0 & 16 \end{bmatrix} = \begin{bmatrix} 2 & -2 \\ -2 & 8 \end{bmatrix}$$

- since T=0 and used=0 we finally have:

$$A = Q - d_1 d_1^T = \begin{bmatrix} 2 & -2 \\ -2 & 8 \end{bmatrix} - \begin{bmatrix} 2 & -4 \\ -4 & 8 \end{bmatrix}$$

The previous procedure can be applied to matrix A of function f, so that function f can then be rewritten as:

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

Note finally that, by means of $2n - 2\nu_+(A)$ linear programs we can also compute the following values:

$$\bar{l}_i := \min_{x \in X} d_i^T x, \quad \bar{u}_i := \max_{x \in X} d_i^T x, \quad i = 1, \ldots, n - \nu_+(A) \tag{2.1}$$

so that:

$$\bar{l}_i \leqslant d_i^T x \leqslant \bar{u}_i \quad \forall x \in X, \ \forall i = 1, \ldots, n - \nu_+(A).$$

REMARK 2.2. The problem of decomposing a symmetric matrix $A \in \Re^{n \times n}$ as the difference of two positive semidefinite matrices have been generally approached in the literature using the diagonal form of A (see for example [1,7,18,23]), so that it is necessary to compute the n eigenvectors of A (hence to resolve n homogeneous linear systems). All the decompositions proposed in this paper are computed directly by means of at most n "pivoting-like" operations and without the need of computing eigenvalues and eigenvectors. Clearly, this chosen approach results to be less "expensive", with respect to both time-computing and numerical errors, than the one based on the diagonal form of A.

2.2. RELAXATIONS

In the branch and bound algorithm we propose to partition the feasible region using the hyperplanes $d_i^T x$, $i = 1, \ldots, n - \nu_+(A)$, whose number equals the number of nonpositive eigenvalues of the original quadratic form. In particular, in

the current step of the branch and bound algorithm the considered partition of the feasible region is of the kind $X \cap Y$ where:

$$\bar{Y} = \{x \in \Re^n : \bar{l}_i \leqslant d_i^T x \leqslant \bar{u}_i \forall i = 1, \ldots, n - \nu_+(A)\},$$
$$Y = \{x \in \Re^n : l_i \leqslant d_i^T x \leqslant \bar{u}_i \forall i = 1, \ldots, n - \nu_+(A)\} \subseteq \bar{Y}.$$

Obviously, $X \cap \bar{Y} = X$ holds. As a consequence, function f can be relaxed over the current partition just by linearizing the concave form $-\sum_{i=1}^{n-\nu_+(A)} (d_i^T x)^2$ over Y, and thus we obtain the function:

$$f_Y(x) = \frac{1}{2} x^T Q x - \frac{1}{2} \sum_{i=1}^{n-\nu_+(A)} [d_i^T x (l_i + u_i) - l_i u_i] + c^T x,$$
$$= \frac{1}{2} x^T Q x + \tilde{c}^T x + \tilde{c}_0,$$

with:

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

In particular, we have:

$$f(x) - f_Y(x) = \frac{1}{2} \sum_{i=1}^{n-\nu_+(A)} (u_i - d_i^T x)(d_i^T x - l_i),$$
$$= -\frac{1}{8} \sum_{i=1}^{n-\nu_+(A)} (2 d_i^T x - (l_i + u_i))^2 + \frac{1}{8} \sum_{i=1}^{n-\nu_+(A)} (u_i - l_i)^2.$$

Hence, the maximum error due to the linearization of the concave form $-\sum_{i=1}^{n-\nu_+(A)} (d_i^T x)^2$ over the current partitions Y is:

$$\mathrm{Err}(f_Y) = \frac{1}{8} \sum_{i=1}^{n-\nu_+(A)} (u_i - l_i)^2. \tag{2.2}$$

In other words, for all $x \in X \cap Y$ we have:

$$0 \leqslant f(x) - f_Y(x) \leqslant \mathrm{Err}\ (f_Y).$$

Notice that the maximum error $\mathrm{Err}(f_Y)$ is attained at feasible points x such that $d_i^T x = \frac{l_i + u_i}{2} \forall i = 1, \ldots, n - \nu_+(A)$ (these points exist since vectors d_i are linearly independent).

REMARK 2.3. It is worth comparing the proposed approach with those reported in the literature [1, 7, 18, 23]. First of all, in these papers both

the objective function and the feasible region are usually transformed by means of a change of variables based on the eigenvectors of A, while in our approach just the objective function is decomposed leaving both the variables and the feasible region untouched. This implies that the original structure of the feasible region is maintained and that all numerical errors due to the computing of the eigenvectors are avoided.

Finally, note that our approach refers to a general compact feasible region (not necessarily a box constrained one) and that the relaxed problems are strictly convex ones (generally more easy to be solved than convex ones).

2.3. BRANCH AND BOUND

The results of the previous subsection allow us to relax problem P over the current partitions Y with the following problem:

$$P_Y : \begin{cases} \min f_Y(x) \\ x \in X \cap Y. \end{cases}$$

Note that P_Y is a strictly convex quadratic problem, hence it can be solved with any of the algorithms known in the literature.

We are now able to suggest a branch and bound scheme based on the proposed relaxation. The main procedure "Solve$_1$()" just initialize the algorithm, call the recursive subprocedure "Explore$_1$()" and then provides the optimal solution. As usual in the branch and bound schemes, the tolerance accepted for the maximum error $\mathrm{Err}(f_Y)$ has to be fixed *a priori.* In the procedure this is done by means of a positive value ϵ which provides the desired precision. In other words, in order to guarantee the finite convergence of the branch and bound scheme, the partitioning of Y is stopped whenever the maximum error $\mathrm{Err}(f_Y)$ is lower than or equal to the chosen precision $\epsilon > 0$.

Procedure **Solve**$_1(P)$

determine a decomposition $A = Q - \sum_{i=1}^{n-\nu_+(A)} d_i d_i^T$;

determine $\bar{l}_i$ and $\bar{u}_i \forall i = 1, \ldots, n - \nu_+(A)$;

fix the positive value ϵ; $UB := +\infty$;

Explore$_1(\bar{Y})$;

x^* is the provided solution and UB is its value;

end proc.

The core of the algorithm is the recursive procedure "Explore$_1$()" which is described below. This procedure is based on a generalization of the so called "rectangular partitioning method" (see [9, 25]). Note that we denote with σ the separating value used in the bipartition of the current feasible region.

Procedure **Explore**$_1(Y)$
if $X \cap Y \neq \emptyset$ *then*
let $\bar{x}$ be the optimal solution of P_Y;
if $f(\bar{x}) < UB$ *then* $UB := f(\bar{x})$; $x^* := \bar{x}$ *end if*;
if $f_Y(\bar{x}) < UB$ and $\mathrm{Err}(f_Y) > \epsilon$ *then*
let $i \in \{1, \ldots, n - \nu_+(A)\}$ be such that:

$$u_i - l_i = \max_{j \in \{1,\ldots,n-\nu_+(A)\}, l_j < d_j^T \bar{x} < u_j} \{u_j - l_j\} \tag{2.3}$$

determine $\sigma \in (l_i, u_i)$;
define $Y_1 = \{x \in Y : l_i \leqslant d_i^T x \leqslant \sigma\}$;
define $Y_2 = \{x \in Y : \sigma \leqslant d_i^T x \leqslant u_i\}$;
Explore$_1(Y_1)$;
Explore$_1(Y_2)$;
end if;
end if;
end proc.

Procedure "Explore$_1(Y)$" first looks for the optimal solution $\bar{x}$ of the corresponding relaxed problem P_Y; if $\bar{x}$ allows to decrease the upper bound UB it is chosen as the new incumbent optimal solution. Then the value of the relaxed function $f_Y(\bar{x})$ is analyzed; if it is not lower than UB, that is:

$$f(\bar{x}) \geqslant \min_{x \in X \cap Y} f(x) \geqslant \min_{x \in X \cap Y} f_Y(x) = f_Y(\bar{x}) \geqslant UB,$$

then it is impossible to improve the incumbent optimal solution by exploring Y furthermore. In the case:

$$f_Y(\bar{x}) < UB \leqslant f(\bar{x}),$$

then there exists at least one index $j \in \{1, \ldots, n - \nu_+(A)\}$ such that $l_j < d_j^T \bar{x} < u_j$ (if such an index does not exist, that is $(u_j - d_j^T x)(d_j^T x - l_j) = 0$ $\forall j \in \{1, \ldots, n - \nu_+(A)\}$, it is $f_Y(\bar{x}) = f(\bar{x})$), hence it is possible to partition the set Y into two subsets Y_1 and Y_2 as shown before. The incumbent optimal solution can then be improved using the relaxed functions f_{Y_1} and f_{Y_2}, defined over the sets Y_1 and Y_2, respectively, since these functions provide a better approximation of f than f_Y. We now will show how the separating value σ can be chosen:

- $\sigma = d_i^T \bar{x}$: in this case the branch and bound scheme follows the lines proposed in [9], where the finite convergence has been fully proved; it

is worth noticing also that in this case we always have $X \cap Y_1 \neq \emptyset$ and $X \cap Y_2 \neq \emptyset$ since $\bar{x} \in Y_1 \cap Y_2$;

- $\sigma = \frac{1}{2}(l_i + u_i)$: in this case the maximum errors $\mathrm{Err}(f_{Y_1})$ and $\mathrm{Err}(f_{Y_2})$ in the subsequent subproblems are reduced by $\frac{3}{4}(u_i - l_i)^2$ with respect to $\mathrm{Err}(f_Y)$; the finite convergence then follows trivially since a positive precision value ϵ has been fixed and the index i is chosen as described in (2.3); note that in this case it might happen that $X \cap Y_1 = \emptyset$ or $X \cap Y_2 = \emptyset$.

Finally, note that in the case condition $\mathrm{Err}(f_Y) \leqslant \epsilon$ is never used to stop the branch progress, that is to say that the branching is always stopped since $f_Y(\bar{x}) \geqslant UB$ or $X \cap Y = \emptyset$, then the provided solution is actually the optimal solution of the problem.

EXAMPLE 2.2. Let us solve with the described approach the next problem:

$$\begin{cases} \min f(x_1, x_2) = \frac{1}{2}[x_1, x_2] \begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ (x_1, x_2) \in X = \{(x_1, x_2) \in \Re^2 : -1 \leqslant x_1 \leqslant 3, \quad -2 \leqslant x_2 \leqslant 3\} \end{cases}$$

- The matrix in the quadratic form of the objective function is decomposed as described in Example 2.1, so that

$$f(x_1, x_2) = \frac{1}{2}[x_1, x_2] \begin{bmatrix} 2 & -2 \\ -2 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - (-x_1 + 2x_2)^2$$

- we get

$$\bar{l}_1 = \min_{(x_1, x_2) \in X} (-x_1 + 2x_2) = -7, \quad \bar{u}_1 = \max_{(x_1, x_2) \in X} (-x_1 + 2x_2) = 7$$

attained in $(x_1, x_2) = (3, -2)$ and $(x_1, x_2) = (-1, 3)$, respectively

- $\epsilon = \frac{1}{10}$; $UB := +\infty$; $\bar{Y} = \{(x_1, x_2) \in \Re^2 : -7 \leqslant -x_1 + 2x_2 \leqslant 7\}$;

$(P_{\bar{Y}})$ $f_{\bar{Y}}(x_1, x_2) = x_1^2 + 4x_2^2 - 2x_1x_2 - 49$; the optimal solution is (0,0); $UB := f(0, 0) = 0$; $x^* := (0, 0)$; since $f_{\bar{Y}}(0, 0) = -49 < UB$ and $\mathrm{Err}(f_{\bar{Y}}) = \frac{49}{2} > \epsilon$ we get $Y_1 = \{(x_1, x_2) \in \bar{Y} : -7 \leqslant -x_1 + 2x_2 \leqslant 0\}$ and $Y_2 = \{(x_1, x_2) \in \bar{Y} : 0 \leqslant -x_1 + 2x_2 \leqslant 7\}$

(P_{Y_1}) $f_{Y_1}(x_1, x_2) = x_1^2 + 4x_2^2 - 2x_1x_2 - 7x_1 + 14x_2$; the optimal solution is $(\frac{7}{3}, -\frac{7}{6})$; $UB := f(\frac{7}{3}, -\frac{7}{6}) = -\frac{49}{9}$; $x^* := (\frac{7}{3}, -\frac{7}{6})$; since $f_{Y_1}(\frac{7}{3}, -\frac{7}{6}) = -\frac{49}{3} < UB$ and $\mathrm{Err}(f_{Y_1}) = \frac{49}{8} > \epsilon$ we get $Y_{1,1} = \{(x_1, x_2) \in Y_1 : -7 \leqslant -x_1 + 2x_2 \leqslant -\frac{14}{3}\}$ and $Y_{1,2} = \{(x_1, x_2) \in Y_1 : -\frac{14}{3} \leqslant -x_1 + 2x_2 \leqslant 0\}$

($P_{Y_{1,1}}$) $f_{Y_{1,1}}(x_1, x_2) = x_1^2 + 4x_2^2 - 2x_1x_2 - \frac{35}{3}x_1 + \frac{70}{3}x_2 + \frac{98}{3}$; the optimal solution is (3,−2); $UB := f(3,-2) = -12$; $x^* := (3,-2)$; $f_{Y_{1,1}}(3,-2) = -12 \geqslant UB$

($P_{Y_{1,2}}$) $f_{Y_{1,2}}(x_1, x_2) = x_1^2 + 4x_2^2 - 2x_1x_2 + \frac{14}{3}x_1 - \frac{28}{3}x_2$; the optimal solution is (0, 0); $f(0,0) = 0$; $f_{Y_{1,2}}(0,0) = 0 > UB$

(P_{Y_2}) $f_{Y_2}(x_1, x_2) = x_1^2 + 4x_2^2 - 2x_1x_2 + 7x_1 - 14x_2$; the optimal solution is $(-1, \frac{3}{2})$; $f(-1, \frac{3}{2}) = -3$; since $f_{Y_2}(-1, \frac{3}{2}) = -15 < UB$ and $\mathrm{Err}(f_{Y_2}) = \frac{49}{8} > \epsilon$ we get $Y_{2,1} = \{(x_1, x_2) \in Y_2 : 0 \leqslant -x_1 + 2x_2 \leqslant 4\}$ and $Y_{2,2} = \{(x_1, x_2) \in Y_2 : 4 \leqslant -x_1 + 2x_2 \leqslant 7\}$

($P_{Y_{2,1}}$) $f_{Y_{2,1}}(x_1, x_2) = x_1^2 + 4x_2^2 - 2x_1x_2 + 4x_1 - 8x_2$; the optimal solution is $(-1, \frac{3}{4})$; $f(-1, \frac{3}{4}) = -\frac{3}{2}$; $f_{Y_{2,1}}\left(-1, \frac{3}{4}\right) = -\frac{21}{4} > UB$

($P_{Y_{2,2}}$) $f_{Y_{2,2}}(x_1, x_2) = x_1^2 + 4x_2^2 - 2x_1x_2 + 11x_1 - 22x_2 + 28$; the optimal solution is $(-1, \frac{5}{2})$; $f(-1, \frac{5}{2}) = -5$; $f_{Y_{2,2}}(-1, \frac{5}{2}) = -7 > UB$

- the provided solution is $x^* := (3, -2)$, which is also the optimal solution of the problem since the condition $\mathrm{Err}(f_Y) \leqslant \epsilon$ has never been used to stop the branch progress. It is finally $f(3,-2) = -12$.

2.4. FURTHER ENHANCEMENTS

In [5] a finite algorithm based on the so called "optimal level solutions" approach [5,8] has been proposed to solve problems of the kind

$$\begin{cases} \min g(x) = \frac{1}{2}x^T Qx - \frac{1}{2}(d^T x)^2 + c^T x \\ x \in X = \{x \in \Re^n : Bx \geqslant b\}, \end{cases} \tag{2.4}$$

where X is a compact polyhedron, $B \in \Re^{m \times n}$, $b \in \Re^m$, $c, d \in \Re^n$ and $Q \in \Re^{n \times n}$ is a positive definite symmetric matrix.

This algorithm can be used to improve the tightness of the relaxations used in the branch and bound scheme described in the previous subsections.

Let us first recall that in the previous subsections function f has been decomposed as:

$$f(x) = \frac{1}{2}x^T Qx - \frac{1}{2}\sum_{i=1}^{k}(d_i^T x)^2 + c^T x \qquad \text{where } k = n - \nu_+(A),$$

with a maximum error, due to the initial relaxation, given by:

$$\mathrm{Err}(f_{\bar{Y}}) = \frac{1}{8}\sum_{i=1}^{k}(\bar{u}_i - \bar{l}_i)^2.$$

In order to decrease as much as possible the error caused by the relaxations, let us denote with $j \in \{1, \ldots, k\}$ an index such that:

$$\bar{u}_j - \bar{l}_j = \max_{i=1,\ldots,k} \{\bar{u}_i - \bar{l}_i\}, \tag{2.5}$$

where $\bar{u}_i$ and $\bar{l}_i$ are the bounds defined in (2.1). The partition of the feasible region used in the current step of the branch and bound algorithm is now of the kind $X \cap Y_j$ where:

$$\begin{aligned} \overline{Y}_j &= \{x \in \Re^n : \bar{l}_i \leqslant d_i^T x \leqslant \bar{u}_i \quad \forall i = 1, \ldots, k, \quad i \neq j\}, \\ Y_j &= \{x \in \Re^n : l_i \leqslant d_i^T x \leqslant u_i \quad \forall i = 1, \ldots, k, \quad i \neq j\} \subseteq \overline{Y}_j. \end{aligned}$$

Function f can then be relaxed over the current partition just linearizing the concave form $-\sum_{i=1,i\neq j}^{k} (d_i^T x)^2$ over Y_j with the function:

$$\begin{aligned} g_{Y_j}(x) &= \frac{1}{2} x^T Q x - \frac{1}{2} (d_j^T x)^2 - \frac{1}{2} \sum_{i=1,i\neq j}^{k} [d_i^T x (l_i + u_i) - l_i u_i] + c^T x, \\ &= \frac{1}{2} x^T Q x - \frac{1}{2} (d_j^T x)^2 + \bar{c}^T x + \bar{c}_0, \end{aligned}$$

where:

$$\bar{c} = c - \frac{1}{2} \sum_{i=1,i\neq j}^{k} d_i (l_i + u_i) \text{ and } \bar{c}_0 = \frac{1}{2} \sum_{i=1,i\neq j}^{k} l_i u_i.$$

Note that function g_{Y_j} is of the same kind of the objective function of problem (2.4). Then it follows that:

$$f(x) - g_{Y_j}(x) = -\tfrac{1}{8} \sum_{i=1,i\neq j}^{k} (2 d_i^T x - (l_i + u_i))^2 + \tfrac{1}{8} \sum_{i=1,i\neq j}^{k} (u_i - l_i)^2.$$

Hence, the maximum error in linearizing the concave form $-\sum_{i=1,i\neq j}^{k} (d_i^T x)^2$ over the current partition Y_j is:

$$\mathrm{Err}(g_{Y_j}) = \frac{1}{8} \sum_{i=1,i\neq j}^{k} (u_i - l_i)^2 = \mathrm{Err}(f_{Y_j}) - \frac{1}{8} (\bar{u}_j - \bar{l}_i)^2. \tag{2.6}$$

In other words, for all $x \in X \cap Y_j$ we have:

$$0 \leqslant f(x) - g_{Y_j}(x) \leqslant \mathrm{Err}(g_{Y_j}) < \mathrm{Err}(f_{Y_j}),$$

hence g_{Y_j} is more tight than f_{Y_j}.

Problem P can then be relaxed over the current partition Y_j with the problem:

$$G_{Y_j}: \begin{cases} \min g_{Y_j}(x) \\ x \in X \cap Y_j, \end{cases}$$

which can be solved by means of the algorithm proposed in [5]. Note that this relaxation is tighter than P_{Y_j} but, from a computational point of view, G_{Y_j} (which is not a positive definite quadratic program) is more time-expensive to be solved than P_{Y_j} (which is a strictly convex program). Obviously, if $\nu_+(A) = n-1$, that is to say that A has one nonpositive eigenvalue, the problem can be solved directly by the algorithm proposed in [5], without any branch and bound steps.

The branch and bound scheme based on the relaxation G_{Y_j} is analogous to the one described in the previous section and its finite convergence follows on the same lines described in Section 2.3. The main procedure becomes:

Procedure **Solve**$_2(P)$
  determine a decomposition $A = Q - \sum_{i=1}^{n-\nu_+(A)} d_i d_i^T$;
  determine $\bar{l}_i$ and $\bar{u}_i \forall i = 1, \ldots, n - \nu_+(A)$;
  determine j as in (2.5);
  fix the positive value ϵ; $UB := +\infty$;
  Explore$_2(\overline{Y}_j)$;
  x^* is the provided solution and UB is its value;
end proc.

while the recursive procedure "Explore$_2$()" is:

Procedure **Explore**$_2(Y_j)$
  if $X \cap Y_j \neq \emptyset$ *then*
    let $\bar{x}$ be the optimal solution of G_{Y_j};
  if $f(\bar{x}) < UB$ *then* $UB := f(\bar{x})$; $x^* := \bar{x}$ *end if*;
  if $g_{Y_j}(\bar{x}) < UB$ *and* $\mathrm{Err}(g_{Y_j}) > \epsilon$ *then*
    let $i \in \{1, \ldots, n - \nu_+(A), i \neq j\}$ be such that:

$$u_i - l_i = \max_{j \in \{1,\ldots,n-\nu+(A),\ i \neq j\},\ l_j < d_j^T \bar{x} < u_j} \{u_j - l_j\} \tag{2.7}$$

  determine $\sigma \in (l_i, u_i)$;
  define $Y_j^1 = \{x \in Y_j : l_i \leqslant d_i^T x \leqslant \sigma\}$;
  define $Y_j^2 = \{x \in Y_j : \sigma \leqslant d_i^T x \leqslant u_i\}$;
  Explore$_2(Y_j^1)$;

Explore$_2(Y_j^2)$;
end if;
end if;
end proc.

In "Explore$_2$()" the value σ can be chosen as described in Section 2.3.

3. Box Constrained Case

Let us consider now the following particular case of problem P:

$$P_B: \begin{cases} \min f(x) = \frac{1}{2}x^T Ax + c^T x \\ x \in X_B = \{x \in \Re^n : \tilde{l}_i \leqslant x_i \leqslant \tilde{u}_i, \ \ i = 1, \ldots, n\}, \end{cases}$$

that is the case, where P is a box constrained problem.

Obviously, problem P_B can be solved as described in the previous section; recall that those approaches require that $2n - 2v_+(A)$ linear programs are solved in the initialization of the branch and bound.

In this section, we aim to show that different decompositions of matrix A allow us to avoid the computing of the solutions of such linear programs.

3.1. RELAXATIONS AND BRANCH AND BOUND

Our approach is based on the decomposition of matrix A in the following form[2]:

$$A = Q - dd^T - \text{diag}(w), \tag{3.1}$$

where $Q \in \Re^{n\times n}$ is symmetric and positive definite, $d, w \in \Re^n$, $w \geqslant 0$ and diag (w) is the positive semidefinite diagonal matrix with diagonal elements given by the components of vector w. Note that the vector d can be equal to the null vector.

Such a decomposition allows us to rewrite function f as follows:

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

The feasible region can now be partitioned using the variables $x_i, i = 1, \ldots, n$, such that $w_i > 0$. In particular, in the current step of the branch and bound algorithm the considered partition of the feasible region is of the kind $X_B \cap W$ where:

[2]Note that in [11,18] a decomposition of this kind (with $d = 0$ and diag $(w) = kI_n, k > 0$ large enough) was studied. Another decomposition of this kind with $d = 0$, based on diagonal dominance, has been proposed in [18].

$$\overline{W} = \{x \in \Re^n : \tilde{l}_i \leqslant x_i \leqslant \tilde{u}_i \forall i \text{ such that } w_i > 0\} \supseteq X_B,$$
$$W = \{x \in \Re^n : l_i \leqslant x_i \leqslant u_i \ \forall i \text{ such that } w_i > 0\} \subseteq \overline{W}.$$

As a consequence, problem P_B can be relaxed over the current partition just by linearizing the concave form $-\frac{1}{2}\sum_{i=1}^{n} w_i x_i^2$ over W, and thus we obtain the problem:

$$P_{BW} : \begin{cases} \min f_W(x) \\ x \in X_B \cap W, \end{cases}$$

where:

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

with:

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

Note that:

- in the case of $d = 0$ problem P_{BW} is a strictly convex quadratic problem and problem P_B can be solved with a branch and bound scheme analogous to the one described in Subsection 2.3,
- in the case of $d \neq 0$ problem P_{BW} is of the kind (2.4) and can be solved by means of the algorithm proposed in [5], hence problem P_B can be approached with a branch and bound scheme analogous to the one described in Subsection 2.4.

Since:

$$f(x) - f_W(x) = \frac{1}{2}\sum_{i=1}^{n} w_i(u_i - x_i)(x_i - l_i),$$
$$= -\frac{1}{8}\sum_{i=1}^{n} w_i(2x_i - (l_i + u_i))^2 + \frac{1}{8}\sum_{i=1}^{n} w_i(u_i - l_i)^2,$$

the maximum error done linearizing the concave form $-\frac{1}{2}\sum_{i=1}^{n} w_i x_i^2$ over the current partition W is:

$$\mathrm{Err}(f_W) = \frac{1}{8}\sum_{i=1}^{n} w_i(u_i - l_i)^2, \tag{3.2}$$

which is attained at points $\tilde{x} \in X_B \cap W$ such that:

$$\tilde{x}_i = \frac{1}{2}(l_i + u_i) \quad \forall i \in \{1, \ldots, n\} \text{ such that } w_i > 0.$$

In particular, for all $x \in X_B \cap W$ we have:

$$0 \leqslant f(x) - f_W(x) \leqslant \mathrm{Err}(f_W).$$

It is now clear that in order to decrease the maximum error $\mathrm{Err}(f_W)$ in the various branch and bound steps we have to use decompositions of the kind (3.1) with a vector w having many zero components and positive ones with a small value.

The branch and bound scheme which solves problem P_B is analogous to the ones described in the previous sections and its finite convergence follows on the same lines described in Section 2.3. The main procedure is:

Procedure **Solve**$_3(P_B)$
  determine a decomposition $A = Q - dd^T - \mathrm{diag}(w)$;
  fix the positive value ϵ; $UB := +\infty$;
  Explore$_3(\overline{W})$;
  x^* is the provided solution and UB is its value;
end proc.

The core of the algorithm is in the following recursive procedure "Explore$_3$()":

Procedure **Explore**$_3(W)$
  if $X \cap W \neq \emptyset$ *then*

    Let $\bar{x}$ be the optimal solution of P_{BW}[3];
    if $f(\bar{x}) < UB$ *then* $UB := f(\bar{x})$; $x^* := \bar{x}$ *end if*;
    if $f_W(\bar{x}) < UB$ *and* $\mathrm{Err}(f_W) > \epsilon$ *then*
      let $i \in \{1, \ldots, n\}$ be such that:

$$w_i(u_i - l_i)^2 = \max_{j \in \{1,\ldots,n\},\ w_j > 0,\ l_j < \bar{x}_j < u_j} \{w_j(u_j - l_j)^2\} \tag{3.3}$$

      determine $\sigma \in (l_i, u_i)$;
      define $W_1 = \{x \in W : l_i \leqslant x_i \leqslant \sigma\}$;

[3]If $d = 0$ the optimal solution $\bar{x}$ of P_{BW} is determined by means of any of the known algorithm for positive definite quadratic problems; in the case $d \neq 0$ it is determined by means of the algorithm proposed in [5].

define $W_2 = \{x \in W : \sigma \leqslant x_i \leqslant u_i\}$;
Explore$_3(W_1)$;
Explore$_3(W_2)$;
end if;
end if;
end proc.

Note that since $f_W(\bar{x}) < UB$ there exists at least one index $i \in \{1, \ldots, n\}$ satisfying (3.3). In fact, if $w_j(u_j - \bar{x}_j)(\bar{x}_j - l_j) = 0 \ \forall j \in \{1, \ldots, n\}$ then $f_W(\bar{x}) = f(\bar{x}) \geqslant UB$ which is a contradiction. Note finally that, in "Explore$_3$()", the value σ can be chosen as follows:

$$\sigma = \bar{x}_i \ \text{ or } \ \sigma = \frac{l_i + u_i}{2}.$$

3.2. DECOMPOSITION PROCEDURES

Let us now provide some procedures which decompose matrix A in the form:

$$A = Q - dd^T - \text{diag}(w),$$

as described in the previous subsection. Note that, in order to obtain relaxations as tight as possible, the proposed procedures are aimed to determine a vector w with as few positive components as possible.

The first procedure, that is `Minor`(A,Q,w), provides a decomposition of the kind (3.1) with $d = 0$ and is based on the well-known characterization of positive definite matrices based on the positiveness of all their NW minors. In this procedure the following notations are used:

- Q_k is the $k \times k$ NW submatrix of Q,
- $Q_{k\setminus i}$ is the $k \times k$ NW submatrix of Q without its i-th row and column.

Procedure **Minor**(*inputs*: A; *outputs*: Q,w)
Q:=A; w:=0;
if Q[1,1] $\leqslant$ 0 *then* w[1]:=1−Q[1,1]; Q[1,1]:=1 *end if*;
for k *from* 2 *to* n *do*
if det(Q_k) $\leqslant$ 0
then if Q[k,k] $\leqslant$ 0 *then* h:=k; den:=det(Q_{k-1});
else found:= *false*;
for i *from* 1 *to* k − 1 *do*
if not(found) *and* w[i]> 0 *and* det($Q_{k\setminus i}$) > 0
then h:=i; den:=det($Q_{k\setminus i}$); found:= *true*;

end if;
end do;
if not(found) *then* h:=k; den:=det(Q_{k-1})*end if*;
end if;
val:= $1-\dfrac{\det(Q_k)}{\text{den}}$; w[h]:=w[h]+val; Q[h,h]:=Q[h,h]+val;
end if;
end do;
end proc.

In procedure `Minor`(A,Q,w) the "for" cycle is repeated $n-1$ times. At the k-th iteration the k-th NW minor of Q is checked and, if the case, forced to be positive by updating a diagonal element of Q; the decomposition is maintained by increasing properly an element of w. Note that, in order to obtain a vector w with as few positive components as possible, if the k-th NW minor of Q is nonpositive the procedure `Minor`(A,Q,w) first try to make it positive by increasing an already positive component of w, while a new positive component of w is created just if this is not possible.

The limit of this approach is that the diagonal elements are analyzed in a fixed order, that is from the first to the last one. Better results may be obtained with algorithms which consider the diagonal elements not in a fixed order, such as the procedure `Decomp1`(A,Q,w) which provides a decomposition of the kind (3.1) with $d=0$ and which is based on pivoting operations similar to the ones used in procedure `ModLagrange`.

Procedure **Decomp1**(*inputs*: A; *outputs*: Q,w)
T:=A; Q:=0; w:=0; used:=$[1,1,\ldots,1,1]\in\Re^n$;
while T$\neq$0 *do*
if T$[i,i]\leqslant 0 \forall i\in\{1,\ldots,n\}$
then select $r\in\{1,\ldots,n\}$ *such that* row[T,r]$\neq$0;
w[r]:=1−T[r,r]; T[r,r]:=1;
else select $r\in\{1,\ldots,n\}$ such that $T[r,r]>0$ and $T-\dfrac{1}{T[r,r]}vv^T$ has as much positive diagonal elements as possible, where v=row[T,r];
end if;
v:=row[T,r]; α:=T[r,r]; T:=T$-\dfrac{1}{\alpha}$vv^T; Q:=Q$+\dfrac{1}{\alpha}$vv^T; used [r]:=0;
end do;
w:=w+used; Q:=Q+diag(used);
end proc.

The previous procedure differs from `ModLagrange` the way where $T[r,r] \leqslant 0$ is approached; in facts, in procedure `Decomp1` when $T[r,r] \leqslant 0$ the r-th component of w is updated and no d_k vector is created.

Note that in order to obtain a vector w with as few positive components as possible, the procedure `Decomp1`(A,Q,w) first uses the positive diagonal elements of the temporary matrix T, then it analyzes the nonpositive ones.

In the next example the procedures `Minor`(A,Q,w) and `Decomp1`(A,Q,w) are applied to a given matrix A, showing that `Decomp1`(A,Q,w) may provide a decomposition with a matrix diag(w) having smaller rank.

EXAMPLE 3.1. Let us apply procedures `Minor`(A,Q,w) and `Decomp1` (A,Q,w) to the next matrix:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix}$$

We obtain the following decompositions:

$$\text{minor}: \quad \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 5 & 0 \\ 3 & 0 & 46 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 45 \end{bmatrix}$$

$$\text{decomp1}: \quad \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 14 & 2 & 3 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 13 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

hence, in this case, `Decomp1`(A,Q,w) provides a vector w with a number of positive components smaller than `Minor`(A,Q,w).

A vector w with a smaller number of positive components can be obtained with the procedure `Decomp2`(A,Q,d,w) which represents an improvement of the procedure `Decomp1`(A,Q,w). Note that this procedure provides a vector d which may be different from zero.

Procedure **Decomp2**(*inputs*: A; *outputs*: Q, d, w)
$T := A$; $Q := 0$; $w := 0$; used: $= [1,1,\ldots,1,1] \in \Re^n$; d $:= 0$; found $:=$ *false*;
while T $\neq 0$ *do*
  if T$[i,i] \leqslant 0 \forall i \in \{1,\ldots,n\}$
  then if found
    then select $r \in \{1,\ldots,n\}$ *such that* row [T,r] $\neq 0$;
      w[r] $:= 1 -$ T[r,r]; T[r,r]:=1;
    else select $r \in \{1,\ldots,n\}$ *such that* v $=$ row[T,r] $\neq 0$ and
      $T - \dfrac{1}{\alpha} v v^T$ has as much positive diagonal elements

as possible, where α is given by:

$$\alpha = \begin{cases} T[r,r] & \text{if } T[r,r] < 0 \\ -1 & \text{if } T[r,r] = 0 \end{cases};$$

if T[r,r]=0 *then* Q[r,r]:=Q[r,r]+1; T[r,r]:=−1 *end if*;

end if;

else select $r \in \{1,\ldots,n\}$ *such that* T[r,r]>0 and $T - \frac{1}{T[r,r]} vv^T$ has as much positive diagonal elements as possible, where v=row[T,r];

end if;

v:=row[T,r]; α:=T[r,r]; T:=T$-\frac{1}{\alpha}$vv^T; Q:=Q$+\frac{1}{|\alpha|}$vv^T; used[r]:=0;

if $\alpha < 0$ *then* d:=$\sqrt{\frac{-2}{\alpha}}$v; found:=*true*; *end if*;

end do;

w:=w+used; Q:=Q+diag(used);

end proc.

Procedure `Decomp2` (A,Q,d,w) represents a hybrid of the two procedures `ModLagrange` and `Decomp1`; in fact, the scheme of `ModLagrange` is followed until a vector $d \neq 0$ is created, then the lines of `Decomp1` are used.

Finally, it is worth studying how many positive components the vector w in (3.1) may have.

THEOREM 3.1. *Let* $A \in \Re^{n \times n}$ *be a symmetric matrix and denote with* $n_+(A)$ *the number of positive diagonal elements of A and with* A_+ *the submatrix of A obtained deleting the rows and the columns of A corresponding to its nonpositive diagonal elements. Consider also the following decompositions:*

$$A = Q_1 - \text{diag}(w_1),$$
$$A = Q_2 - dd^T - \text{diag}(w_2), \quad d \neq 0,$$

where Q_1 *and* Q_2 *are positive definite,* $d \in \Re^n, d \neq 0$, *and* $w_1, w_2 \in \Re^n$, w_1, $w_2 \geqslant 0$. *Then:*

i) $\text{rank}(\text{diag}(w_1)) \geqslant n - \nu_+(A_+) \geqslant n - \min\{n_+(A), \nu_+(A)\}$,
ii) $\text{rank}(\text{diag}(w_2)) \geqslant \nu - (A) + \nu_0(A) - 1 = n - \nu_+(A) - 1$.

Proof. i) Matrix A, by means of a permutations of its rows and columns, can be rewritten as:

$$\Pi A \Pi^T = \begin{bmatrix} A_+ & A_{12} \\ A_{21} & A_{22} \end{bmatrix}.$$

Since $Q = A + \text{diag}(w)$ is a positive definite matrix we have that

$$\Pi Q \Pi^T = \begin{bmatrix} A_+ + \text{diag}(w_{11}) & A_{12} \\ A_{21} & A_{22} + \text{diag}(w_{22}) \end{bmatrix}$$

is positive definite too. As a consequence A_+ +diag (w_{11}) is positive definite so that, for the "Law of Inertia", w_{11} has at least $n_+(A) - \nu_+(A+)$ positive components and it follows that the diagonal elements of A_{22} +diag (w_{22}) are all positive. Hence, the diagonal elements of A_{22} are nonpositive and the $n - n_+(A)$ components of w_{22} are positive. The obtained conditions imply that vector w has at least $n - \nu_+(A_+)$ positive components. The whole result then follows noticing that both $n_+(A) \geqslant \nu_+(A_+)$ and $\nu_+(A) \geqslant \nu_+(A_+)$.

ii) Follows directly from the "Law of Inertia".

The next property follows directly from procedures `Decomp1`(A,Q,w) and `Decomp2`(A,Q,d,w) and from the "Law of Inertia".

PROPERTY 3.1. Let $A \in \Re^{n \times n}$ *be a symmetric matrix which is not positive definite* (*hence* $\nu_+(A) \leqslant n-1$ *and let* $A = Q - dd^T - \text{diag}(w)$ *be a decomposition obtained with procedure* "*decomp2*". *Then*:

i) $d \neq 0$,

ii) *if* $\nu_-(A) \leqslant 1$ *then* $\text{rank}(\text{diag}(w)) = n - \nu_+(A) - 1$.

Let finally $A = Q_1 - \text{diag}(w_1)$ *be a decomposition obtained with procedure* "decomp1"; *then*:

iii) $\text{rank}(\text{diag}(w)) \leqslant \text{rank}(\text{diag}(w_1)) - 1$.

EXAMPLE 3.2. Let us apply procedures "minor", "decomp1" and "decomp2" to the following matrix:

$$A = \begin{bmatrix} -2 & -2 & -2 \\ -2 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix}$$

We obtain the following decompositions:

$$\begin{matrix} \text{minor}: \\ \text{decomp1}: \end{matrix} \begin{bmatrix} -2 & -2 & -2 \\ -2 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & -2 & -2 \\ -2 & 5 & 1 \\ -2 & 1 & 14 \end{bmatrix} - \begin{bmatrix} 3 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 14 \end{bmatrix}$$

$$\text{decomp2}: \begin{bmatrix} -2 & -2 & -2 \\ -2 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 2 & 2 \\ 2 & 4 & 5 \\ 2 & 5 & 8 \end{bmatrix} - \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix} [2, 2, 2] - \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 4 \end{bmatrix}$$

hence, the use of vector $d \neq 0$ in the decomposition allows us to decrease the rank of diag(w) from 3 to 1.

3.3. SOLUTION SCHEMES

In Subsection 3.1 we have already pointed out that the maximum error given by the relaxed function f_W over the current partition W is:

$$\text{Err}(f_W) = \frac{1}{8} \sum_{i=1}^{n} w_i (u_i - l_i)^2.$$

As a consequence, in order to decrease the maximum error in the various branch and bound steps we have to use decompositions of the kind

$$A = Q - dd^T - \text{diag}(w),$$

with a vector w having many zero components and few positive ones with a small value. In this light several schemes, corresponding to different ways of decomposing matrix A, can be suggested to solve problem P_B:

(1) A is decomposed with procedure `Minor`(A,Q,w) and $d = 0$.
(2) A is decomposed with procedure `Decomp1`(A,Q,w) and $d = 0$. It has been shown in Example 3.1 that this decomposition may provide smaller errors than the ones given by (1).
(3) A is decomposed with procedure `Decomp1`(A,Q,w), a positive component of w, say w_j, is chosen, the vector $d = \sqrt{w_j} e_j \neq 0$ is defined[4] and the j-th component of w is then set to zero, that is $w_j := 0$. For example, in order to decrease the error as much as possible we can choose the index $j \in \{1, \ldots, n\}$ such that:

$$w_j (\tilde{u}_j - \tilde{l}_j)^2 = \max_{i=1,\ldots,n} \{w_i (\tilde{u}_i - \tilde{l}_i)^2\}. \tag{3.4}$$

This is clearly an improvement over the scheme suggested in (2).
(4) A is decomposed with procedure `Decomp2`(A,Q,d,w). It has been shown in Example 3.2 that this decomposition may provide smaller errors than the ones given by (1), (2) and (3).

[4]We denote with e_j the j-th column of the $n \times n$ identity matrix.

(5) Given a predefined vector $d \neq 0$, $A + dd^T$ is decomposed with procedure `Decomp1`$(A + dd^T$,Q,w). This scheme could be useful when the vector d is chosen *a priori*, for example in order to avoid initialization problems in the solving algorithm.

REMARK 3.1. In [11,18], decompositions of the kind $Q = A - \mathrm{diag}(w)$ (that is of the kind (3.1) with $d = 0$) are suggested with $\mathrm{diag}(w) = kI_n$ or with $\mathrm{diag}(w)$ computed using diagonal dominance properties; these decompositions provide a vector w having many (or all) positive components with large values. As a consequence, the obtained relaxations are affected by a large error $\mathrm{Err}(f_W)$.

The procedures `Minor`(A,Q,w), `Decomp1`(A,Q,w) and `Decomp1` (A,Q,d,w) described in the previous subsection try to reduce as much as possible the maximum error $\mathrm{Err}(f_W)$ by determining a vector w with many zero components and few positive ones with a small value. In this way, we may obtain convex relaxations tighter than the ones given in [11,18].

EXAMPLE 3.3. Let us solve the problem described in Example 2.2 with the approach (1). With procedure `Minor`(A,Q,w) we get:

$$\begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 2 & 5 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix}$$

so that $f(x_1, x_2) = \left(\frac{1}{2}x_1^2 + \frac{5}{2}x_2^2 + 2x_1x_2\right) - \frac{1}{2}x_1^2 - \frac{5}{2}x_2^2$

- $\epsilon = \frac{1}{10}$; $UB := +\infty$; $\overline{W} = X$;

($P_{\bar{W}}$) $f_{\bar{W}}(x_1, x_2) = \left(\frac{1}{2}x_1^2 + \frac{5}{2}x_2^2 + 2x_1x_2\right) - x_1 - \frac{5}{2}x_2^2 - \frac{33}{2}$; the optimal solution is $(0, \frac{1}{2})$; $UB := f(0, \frac{1}{2})$=0; x^*:=$(0, \frac{1}{2})$; since $f_{\bar{W}}(0, \frac{1}{2}) = -\frac{137}{8} < UB$ and $\mathrm{Err}(f_{\bar{W}}) = \frac{141}{8} > \epsilon$ we get $W_1 = \{(x_1, x_2) \in \bar{W} : -2 \leqslant x_2 \leqslant \frac{1}{2}\}$ and $W_2 = \{(x_1, x_2) \in \bar{W} : \frac{1}{2} \leqslant x_2 \leqslant 3\}$

(P_{W_1}) $f_{W_1}(x_1, x_2) = \left(\frac{1}{2}x_1^2 + \frac{5}{2}x_2^2 + 2x_1x_2\right) - x_1 + \frac{15}{4}x_2 - 4$; the optimal solution is $(3, -\frac{39}{20})$; $UB := f(3, -\frac{39}{20})$=$-\frac{117}{10}$; $x^* := (3, -\frac{39}{20})$; since $f_{W_1}(3, -\frac{39}{20}) = -\frac{1921}{160} < UB$ and $\mathrm{Err}(f_{W_1}) = \frac{189}{32} > \epsilon$ we get $W_{1,1} = \left\{(x_1, x_2) \in W_1 : -2 \leqslant x_2 \leqslant -\frac{39}{20}\right\}$ and $W_{1,2} = \{(x_1, x_2) \in W_1 : -\frac{39}{20} \leqslant x_2 \leqslant \frac{1}{2}\}$

($P_{W_{1,1}}$) $f_{W_{1,1}}(x_1, x_2) = \left(\frac{1}{2}x_1^2 + \frac{5}{2}x_2^2 + 2x_1x_2\right) - x_1 + \frac{79}{8}x_2 + \frac{33}{4}$; the optimal solution is $(3, -2)$; $UB := f(3, -2) = -12$; $x^* := (3, -2)$; $f_{W_{1,1}}(3, -2) = -12 \geqslant UB$

$(P_{W_{1,2}})$ $f_{W_{1,2}}(x_1, x_2) = \left(\frac{1}{2}x_1^2 + \frac{5}{2}x_2^2 + 2x_1x_2\right) - x_1 + \frac{29}{8}x_2 + \frac{63}{16}$; the optimal solution is $(3, -\frac{77}{40})$; $f(3, -\frac{77}{40}) = -\frac{231}{20}$; $f_{W_{1,2}}(3, -\frac{77}{40}) = -\frac{7489}{640} > UB$;

(P_{W_2}) $f_{W_2}(x_1, x_2) = \left(\frac{1}{2}x_1^2 + \frac{5}{2}x_2^2 + 2x_1x_2\right) - x_1 - \frac{35}{4}x_2 + \frac{9}{4}$; the optimal solution is $(-1, \frac{43}{20})$; $f(-1, \frac{43}{20}) = -\frac{43}{10}$; $f_{W_2}(-1, \frac{43}{20}) = -\frac{1249}{160} > UB$

- the provided (optimal) solution is $x^* := (3, -2)$ with $f(3, -2) = -12$.

4. Acknowledgements

Careful reviews by the anonymous referees are gratefully acknowledged.

References

1. Barrientos, O. and Correa, R. (2000), An algorithm for global minimization of linearly constrained quadratic functions, *Journal of Global Optimization* 16, 77–93.
2. Best, M.J. and Ding, B. (1997), Global and local quadratic minimization", *Journal of Global Optimization* 10, 77–90.
3. Best, M.J. and Ding, B. (2000), A decomposition method for global and local quadratic minimization, *Journal of Global Optimization* 16, 133–151.
4. Bomze, I.M. and Danninger, G. (1994), A finite algorithm for solving general quadratic problems, *Journal of Global Optimization* 4, 1–16.
5. Cambini, R. and Sodini, C. (2002), A finite algorithm for particular d.c. quadratic programming problem, *Annals of Operations Research* 117, 33–49.
6. Churilov, L. and Sniedovich, M. (1999), A concave composite programming perspective on D.C. programming. In: Eberhard, A., Hill, R., Ralph, D. and Glover, B.M. (eds), *Progress in Optimization Applied Optimization*, Vol. 30, Kluwer Academic Publishers, Dordrecht.
7. De Angelis, P.L., Pardalos, P.M. and Toraldo, G. (1997), Quadratic Programming with Box Constraints. In: Bomze I.M., et. al. (eds), *Developments in Global Optimization*, Kluwer Academic Publishers, Dordrecht, pp. 73–93.
8. Ellero, A. (1996), The optimal level solutions method, *Journal of Information and Optimization Sciences* 17, (2), 355–372.
9. Falk, J.E. and Soland, R.M. (1969), An algorithm for separable nonconvex programming problems, *Management Science* 15(9), 550–569.
10. Gantmacher, F.R. (1960), *The Theory of Matrices*, Vol. 1, Chelsea Publishing Company, New York.
11. 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.
12. Horst, R. (1976), An algorithm for nonconvex programming problems, *Mathematical Programming* 10(3), 312–321.
13. Horst, R. (1980), A note on the convergence of an algorithm for nonconvex programming problems, *Mathematical Programming* 19(2), 237–238.
14. Horst, R. and Tuy, H. (1990), *Global Optimization*, Springer-Verlag, Berlin.

15. Horst, R., Pardalos, P.M. and Thoai, N.V. (1995), Introduction to global optimization, *Nonconvex Optimization and Its Applications*, Vol. 3, Kluwer Academic Publishers, Dordrecht.
16. Horst, R. and Nguyen Van Thoai. (1996), A new algorithm for solving the general quadratic programming problem, *Computational Optimization and Applications* 5(1), 39–48.
17. Konno, H. (1976), Maximization of a convex quadratic function under linear constraints, *Mathematical Programming* 11(2), 117–127.
18. 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* 11, 253–285.
19. Muu, L.D. and Oettli, W. (1991), An algorithm for indefinite quadratic programming with convex constraints, *Operations Research Letters* 10(6), 323–327.
20. Pardalos, P.M., Glick, J.H. and Rosen, J.B. (1987), Global minimization of indefinite quadratic problems, *Computing* 39(4), 281–291.
21. Pardalos, P.M. (1991), Global optimization algorithms for linearly constrained indefinite quadratic problems, *Computers and Mathematics with Applications* 21, 87–97.
22. 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*, 17(5), 215–220.
23. Rosen, J.B. and Pardalos, P.M. (1986), Global minimization of large scale constrained concave quadratic problems by separable programming, *Mathematical Programming* 34, 163–174.
24. Tuy, H. (1995), D.C. optimization: theory, methods and algorithms, In: Horst R., and Pardalos, P.M. (eds.), *Handbook of Global Optimization, Nonconvex Optimization and Its Applications*, Vol. 2, Kluwer Academic Publishers, Dordrecht, pp. 149–216.
25. Tuy, H. (1998) Convex analysis and global optimization, *Nonconvex Optimization and Its Applications*, Vol. 22, Kluwer Academic Publishers, Dordrecht.