AN EFFECTIVE ALGORITHM FOR QUADRATIC MINIMIZATION PROBLEMS

M. J. Best and K. Ritter

Mathematics Research Center
University of Wisconsin—Madison
610 Walnut Street
Madison, Wisconsin 53706

October 1976

Received September 3, 1976

Sponsored by
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park
North Carolina 27709

Approved for public release
Distribution unlimited
AN EFFECTIVE ALGORITHM FOR QUADRATIC MINIMIZATION PROBLEMS

M. J. Best and K. Ritter

Technical Summary Report # 1691
October 1976

ABSTRACT

An algorithm is described that determines a stationary point of a quadratic minimization problem in a finite number of steps. This finite termination property is based on the use of conjugate directions. The main feature of the algorithm is a new update procedure which preserves conjugate directions if the set of active constraints changes.

AMS(MOS) Subject Classification - 90C20

Key Words: Quadratic programming, conjugate directions, finite termination.

Work Unit No. 5 - Mathematical Programming and Operations Research

Sponsored by the United States Army under Contract No. DAAG29-75-C-0024.
AN EFFECTIVE ALGORITHM FOR QUADRATIC MINIMIZATION PROBLEMS

M. J. Best and K. Ritter

1. Introduction

A method of conjugate directions is presented for the solution of quadratic minimization problems with linear inequality constraints. The algorithm terminates after a finite number of steps with a stationary point. It is a modification of methods of conjugate directions for general nonlinear objective functions described in [1] and [3].

With each point \( x \) determined by the algorithm an \((n,n)\)-matrix is associated, where \( n \) is the number of variables. If \( q < n \) constraints are active at \( x \), then \( n - q \) columns of this matrix are conjugate and orthogonal to the gradients of all constraints active at \( x \). This property allows an easy construction of search directions which are either Newton directions or are conjugate to certain previous search directions. Combined with an appropriate policy for dropping active constraints this choice of search directions results in the finiteness of the algorithm.

A critical feature of the method to be presented is the procedure used to update the matrix associated with \( x \). If at the next point \( \hat{x} \), constructed by the algorithm, no new constraint becomes active this matrix is updated in the same way as the basis matrix is updated in the simplex-method. If however, a new constraint becomes active at \( \hat{x} \), the normal update procedure results in the loss of the conjugate directions. Therefore, a new update formula is developed which allows the preservation of conjugate directions in a simple and computationally efficient way.

Sponsored by the United States Army under Contract No. DAAG29-75-C-0024.
2. General description of the algorithm

We consider the following quadratic minimization problem: Minimize

\[ Q(x) = c'x + \frac{1}{2} x'Cx \]

subject to the constraints

\[ Ax \leq b , \]

where \( c, x \in \mathbb{R}^n, b \in \mathbb{R}^m, C \) is a symmetric \((n,n)\)-matrix and \( A \) is an \((m,n)\)-matrix.

If \( x^\ast \) is a local or an optimal solution to this problem it follows from the Kuhn-Tucker-Theorem (see e.g [2]) that there is a vector \( u^\ast \in \mathbb{R}^m \) such that

\[
\begin{align*}
(2.1) & \quad c + Cx^\ast = A'u^\ast \\
(2.2) & \quad u^\ast'(Ax^\ast - b) = 0 , \quad u^\ast \leq 0 \\
(2.3) & \quad A x^\ast \leq b .
\end{align*}
\]

Any point satisfying the conditions (2.1) - (2.3) is called a stationary point. If \( C \) is positive semi-definite then \( Q(x) \) is convex and every stationary point is an optimal solution to the given problem.

Throughout the paper we assume that for any \( x \) with \( Ax \leq b \) the gradients of all constraints, active at \( x \), are linearly independent and that the set \( \{ x | Ax \leq b \text{ and } Q(x) \leq Q(x_0) \} \) is bounded.

Let \( x_j \) with \( Ax_j \leq b \) be a point determined by the algorithm. For ease of notation we assume that

\[ a^i_j x_j = (b)_i , \quad i = 1, \ldots, q \]

and

\[ a^i_j x_j < (b)_i , \quad i = q+1, \ldots, m , \]

-2-
where $a'_1, \ldots, a'_m$ denote the rows of $A$.

If $q < n$, set

$$T_j = \{ x | a'_i x = 0, \quad i = 1, \ldots, q \}.$$

We first assume that $C$ is positive definite. Then we can construct a set of $n-q$ vectors

$$c_{ij} \in T_j, \quad i = q+1, \ldots, n$$

which form a basis of the subspace $T_j$ and are conjugate with respect to $C$, i.e., have the property

$$c^i_j C c^k_j = 0, \quad c^i_j C c^i_j = \theta^{-1}_j, \quad i \neq k, \quad i, k = q+1, \ldots, n.$$

With

$$(2.5) \quad D_j = (a'_1, \ldots, a'_q, \theta_{q+1}, \ldots, \theta_n)$$

it follows from $(2.4)$ and $(2.5)$ that

$$D^{-1}_j = (c_{ij}', \ldots, c_{nj}')$$

exists and has the vectors $(2.4)$ as its last $n-q$ columns.

In the algorithm this matrix $D^{-1}_j$ and the numbers

$$\theta_{ij} = [c^i_j C c^i_j]^{-1}, \quad i = q+1, \ldots, n$$

are associated with $x_j$.

Given the point $x_j$, the algorithm determines a new point

$$x_{j+1} = x_j - \sigma_j s_j,$$

where $s_j \in \mathbb{E}^n$ is the search direction and $\sigma_j \in \mathbb{E}^1$ is the step size.

In order to motivate the choice of $s_j$ (see Step 1 of the algorithm) we write the gradient of $Q(x)$ at $x_j$ in the form

$$(2.7) \quad c + C x_j = \sum_{i=1}^{q} \lambda_{ij} a'_i + \sum_{i=q+1}^{m} \lambda_{ij} C c^i_j.$$

If $\lambda_{ij} = 0, \quad i = q+1, \ldots, n$, i.e., if the orthogonal projection of $c + Cx_j$
onto $T_j$ is zero, then $x_j$ is said to be a quasi-stationary point.

First we assume that $x_j$ is not a quasi-stationary point. If we set

$$s_j = \sum_{i=q+1}^{n} (\theta_{ij} c'_i g_j) c_j \quad \text{with} \quad g_j = c + Cx_j,$$

then

$$x_j - s_j$$

is a quasi-stationary point. Indeed, it follows from (2.4) and (2.5) that the vectors $c_{q+1}, \ldots, c_n$ form a basis for $T_j$. Furthermore, for every $k \in \{q+1, \ldots, n\}$,

$$c'_{k_j} \nabla Q(x_j - s_j) = c'_{k_j} (c + Cx_j - Cs_j)$$

$$= c'_{k_j} g_j - c'_{k_j} C \left[ \sum_{i=q+1}^{n} (\theta_{ij} c'_i g_j) c_i \right]$$

$$= c'_{k_j} g_j - c'_{k_j} C c_{k_j} c'_{k_j} g_j = 0.$$

Assuming that $A(x_j - s_j) \leq b$, we can choose $\sigma_j = 1$ and $x_{j+1}$ is a quasi-stationary point which in the case of a convex objective function is an optimal solution to the problem

$$\min \{ Q(x) | a_i x = (b)_i, \quad i = 1, \ldots, q, \quad a_i^1 x < (b)_i, \quad i = q+1, \ldots, m \}.$$

Since the same constraints are active at $x_j$ and at $x_{j+1}$, we have $T_j = T_{j+1}$ and can, therefore, choose $D_{j+1}^{-1} = D_j^{-1}$ (Step 3 of the algorithm).

Next we assume that $x_j$ is a quasi-stationary point. Multiplication of (2.7) with $c_{ij}$, $i = 1, \ldots, q$, gives

$$c'_{ij} g_j = \lambda_{ij}.$$

Thus, either $x_j$ is a stationary point or there is at least one positive number $c'_{ij} g_j$. Let

$$c_{qj} = \max \{ c'_{ij} g_j | i = 1, \ldots, q \} > 0.$$

If we set $s_j = c_{qj}$, it follows from (2.6) and the definition of the inverse matrix
that \( s_j \) is orthogonal to \( C c_{ij} \), \( i = q+1, \ldots, n \). Therefore, for all \( \sigma \),
\[
\nabla Q(x_j - \sigma s_j) \text{ is orthogonal to } C c_{q+1,j}, \ldots, c_{nj}.
\]

If \( \hat{\sigma}_j \) is the optimal step size, i.e.,
\[
Q(x_j - \hat{\sigma}_j s_j) = \min\{Q(x_j - \sigma s_j) | \sigma \geq 0\},
\]
then \( \nabla Q(x_j - \hat{\sigma}_j s_j) \) is orthogonal to \( s_j = c_{qj}, c_{q+1,j}, \ldots, c_{nj} \). Assuming that \( A(x_j - \hat{\sigma}_j s_j) \leq b \), we can choose \( \sigma_j = \hat{\sigma}_j \). Then \( x_{j+1} \) is again a quasi-stationary point. Since
\[
T_{j+1} = \{x \mid a^i x = 0, \ i = 1, \ldots, q-1\}
\]
we obtain the matrix \( D'_{j+1} \) from \( D'_j \) by replacing \( a_q \) with \( \theta_{qj} c_{qj} \). Then
\[
D'_{j+1}^{-1} = (c_{1,j+1}, \ldots, c_{n,j+1}),
\]
where (see Step 4 of the algorithm)
\[
c_{i,j+1} = c_{ij} - \frac{c'_{ij} C c_{qj}}{c'_{qj} C c_{qj}} c_{qj}, \quad i = 1, \ldots, q-1
\]
\[
c_{i,j+1} = c_{ij}, \quad i = q, \ldots, n.
\]
Thus the vectors \( c_{q,j+1}, \ldots, c_{nj} \) are again conjugate with respect to \( C \).

Therefore, if no new constraint becomes active at \( x_{j+1} \), then \( x_{j+1} \) is a quasi-stationary point and it is easy to obtain a set of conjugate directions forming a basis for \( T_{j+1} \). If a new constraint becomes active at \( x_{j+1} \), the situation is completely different. To be specific assume that \( s_j \) is given by (2.8) and that \( a_{q+1} \) is the gradient of the new active constraint. In order to guarantee that \( s_{j+1} \) is a feasible search direction, the vectors \( a_1, \ldots, a_{q+1} \) have to be among the columns of \( D'_{j+1} \), i.e., we have to replace one of the vectors \( C c_{q+1,j}, \ldots, C c_{nj} \) with \( a_{q+1} \). Assume
\[
D'_{j+1} = (a_1, \ldots, a_{q+1}, \theta_{q+2,j} c_{q+2,j}, \ldots, \theta_{nj} c_{nj}).
\]
Then the columns of $D_{j+1}^{-1}$ are given by

$$c_{q+1,j+1} = \frac{c_{q+1,j}}{c_{q+1,j}^*} a_{q+1}$$

$$c_{i,j+1} = c_{ij} - \frac{c_{ij}^* a_{q+1}}{c_{q+1,j}^*} c_{q+1,j}^* i = 1, \ldots, n, i \neq q.$$ 

Thus the vectors $c_{q+2,j+1}, \ldots, c_{n,j+1}$ are conjugate if and only if $c_{ij}^* a_{q+1} = 0$, $i = q+2, \ldots, n$, i.e., if

$$a_{q+1} \in \text{span}(a_1, \ldots, a_q, c_{q+1,j}).$$

A similar argument applies to the case where $s_j = c_{qj}$.

Of course it is possible to construct a new set of conjugate directions which form a basis of $T_{j+1}$. However, if $q$ is much smaller than $n$, then this is a time consuming procedure and the resulting algorithm cannot be expected to be very efficient. Therefore, it is important to develop an update formula for $D_{j}^{-1}$ which allows us to easily "transfer" conjugate directions from the subspace $T_j$ into the subspace $T_{j+1}$ in the case where a new constraint is active at $x_{j+1}$. Such a procedure can be based on the following lemma.

**Lemma 1**

Let $r \in \{1, \ldots, n\}$ and let $G$ be a symmetric $(n,n)$-matrix such that there are vectors

$$p_1, \ldots, p_r \in \mathbb{R}^n$$

with

$$p_i^T G p_k = 0, \quad 1 \leq i < k \leq r$$

and

$$p_i^T G p_i > 0, \quad i = 1, \ldots, r.$$
Let $a \in \mathbb{R}^n$ be such that
\[ p = \sum_{i=1}^{r} \frac{p_i a}{p_i G p_i} \quad p_i \neq 0 . \]

Let $\ell \in \{1, \ldots, r\}$. If
\[ p_{\ell} G p_{\ell} p' - (p_{\ell} a)^2 = 0 , \]
set
\[ q_i = p_i , \quad i = 1, \ldots, r , \quad i \neq \ell , \]
otherwise set
\[ q_i = p_i - \frac{p_{\ell} a(1 - t p_{\ell} a)}{p' a} p - t p_{\ell} a p_{\ell} , \quad i = 1, \ldots, r , \quad i \neq \ell , \]
where $t$ is a solution of the equation
\[ t^2 (p_{\ell} G p_{\ell} p' - (p_{\ell} a)^2) + 2 t p_{\ell} a - 1 = 0 . \]

Then

i) $a' q_i = 0 \quad i = 1, \ldots, r , \quad i \neq \ell$

ii) $q_i' G q_k = 0 , \quad i, k \in \{1, \ldots, r\} - \{\ell\} , \quad i \neq k$

iii) $q_i' G q_i = p_{\ell} G p_{\ell} > 0 , \quad i = 1, \ldots, r , \quad i \neq \ell$

iv) $\text{span}(p, q_i, i = 1, \ldots, r , i \neq \ell) = \text{span}(p_1, \ldots, p_r)$ .

Proof.

Suppose $p_{\ell} G p_{\ell} p' - (p_{\ell} a)^2 = 0$.
Since
\[ p_{\ell} G p_{\ell} p' a = p_{\ell} G p_{\ell} \sum_{i=1}^{r} \frac{(p_i a)^2}{p_i G p_i} + (p_{\ell} a)^2 \]
it follows that $p_{\ell} a = 0 , \quad i = 1, \ldots, r , \quad i \neq \ell$ . Hence the first 3 statements of the lemma are true. If $p_{\ell} G p_{\ell} p' a - (p_{\ell} a)^2 \neq 0$ , then the quadratic equation defined in the lemma has two distinct real solutions. Set
\[ (2.9) \quad d = G p , \quad d_i = G p_i , \quad \beta_i = p_i a , \quad i = 1, \ldots, r \]
and observe that $p'a = p'd$ and
\[ p'd_i = p' G p_i = d_i' p_i = a' p_i = \beta_i , \quad i = 1, \ldots, r . \]
Thus
\[ a'q_i = d'q_i = d'(p_i - \frac{\beta_i(1 - t^2\beta_{\ell})}{d'p} p - t \beta_1 p_{\ell}) \]
\[ = d'p_i - \beta_1 + t \beta_1 \ell - t \beta_1 d'p_{\ell} = 0 \quad \text{for } i = 1, \ldots, r, \ i \neq \ell. \]

Furthermore, for \( i, k \in \{1, \ldots, r\} - \{\ell\} \),
\[ q_i' G q_k = (p_i - \frac{\beta_i(1 - t^2\beta_{\ell})}{d'p} p - t \beta_1 p_{\ell})'(d_k - \frac{\beta_k(1 - t^2\beta_{\ell})}{d'p} d - t \beta_k d_{\ell}) \]
\[ = p_i'd_k - \beta_i(1 - t^2\beta_{\ell}) \frac{\beta_k}{d'p} + \beta_i(1 - t^2\beta_{\ell}) \frac{\beta_k}{d'p} \beta_1 \beta_1 + t^2 \beta_1 \beta_1 \beta_1 p_{\ell} d_{\ell} \]
\[ = p_i'd_k + \beta_i \beta_k \frac{d^2}{d'p} [t^2(d'p d_{\ell} p_{\ell} - \beta_k^2) + 2\beta_k^2 t^2 - 1] \]
\[ = p_i' G p_k. \]

This proves parts ii) and iii) of the lemma. In order to prove the last statement of the lemma, it suffices to show that the vectors \( p, q_1, \ldots, q_{\ell - 1}, q_{\ell + 1}, \ldots, q_r \) are linearly independent. Let
\[ \lambda p + \lambda_1 q_1 + \cdots + \lambda_{\ell - 1} q_{\ell - 1} + \lambda_{\ell + 1} q_{\ell + 1} + \cdots + \lambda_r q_r = 0. \]

Scalar multiplication with \( Gp \) and \( Gp_k \), respectively, gives
\[ \lambda p G p = 0 \quad \text{and} \quad \lambda_k p_k G p_k = 0 \]
from which it follows that \( \lambda = \lambda_1 = \cdots = \lambda_{\ell - 1} = \lambda_{\ell + 1} = \cdots = \lambda_r = 0. \)

In order to show how this lemma can be used to construct a basis of conjugate directions for \( T_{j+1} \), we assume that \( a_{q+1} \) is the gradient of the constraint that becomes active at \( x_{j+1} \). Setting \( G = C, a = a_{q+1}, r = n - q, \ell = 1 \) and \( p_1 = c_{q+1,1}, i = 1, \ldots, r \), we conclude from the lemma that the vectors
\[ (2.10) \quad c_{q+1,i+1} = q_i, \quad i = 2, \ldots, r, \]
form a basis of conjugate directions for
\[ T_{j+1} = \{ x \mid a_1' x = 0, \ i = 1, \ldots, q+1 \}. \]
If \( s_j \) is given by (2.8), then the first \( q+1 \) constraints are active at \( x_{j+1} \) and \( D'_{j+1} = (a_1, \ldots, a_{q+1}, \theta_{q+2}, x_{j+1} C \mathbf{c}_{q+2}, j+1' \ldots, \theta_n, j+1 C \mathbf{c}_{n, j+1}) \).

Therefore, the remaining columns of \( D^{-1}_{j+1} \) are given by (see Step 5 of the algorithm and Lemma 2).

\[
\begin{align*}
c_{q+1, j+1} &= \frac{p}{p'a_{q+1}^2} \\
c_{i, j+1} &= c_{ij} - \frac{c_{ij} a_{q+1}}{p'a_{q+1}^2} p, \quad i = 1, \ldots, q.
\end{align*}
\]

If on the other hand \( s_j \) is equal to \( c_{q+1, j} \), the \( q \)-th constraint is not active at \( x_{j+1} \). Thus we need a basis of conjugate directions for

\[
T^*_{j+1} = \{x | a'_1 x = 0, a'_i x = 0, i = 1, \ldots, q-1\}.
\]

In order to apply the lemma we observe that by (2.6) the vectors \( c_{q+1}, \ldots, c_{nj} \) are conjugate with respect to \( C \) and are elements of the subspace

\[
\{x | a'_1 x = 0, \quad i = 1, \ldots, q-1\}.
\]

With \( G = C, a = a_{q+1}, r = n - q + 1, \ell = 1 \) and \( p_i = c_{q-1+i, j}, i = \ldots, r \) it follows from Lemma 1 that the vectors

\[
c_{q-1+i, j+1} = q_i, \quad i = 2, \ldots, r
\]

form a basis of conjugate directions for \( T^*_{j+1} \). Since

\[
(2.11) \quad D'_{j+1} = (a_1, \ldots, a_{q-1}, a_{q+1}, \theta_{q+1}, x_{j+1} C \mathbf{c}_{q+2}, j+1', \ldots, \theta_n, j+1 C \mathbf{c}_{n, j+1})
\]

the remaining columns of \( D^{-1}_{j+1} \) are given by

\[
\begin{align*}
c_{q, j+1} &= \frac{p}{p'a_{q+1}^2} \\
c_{i, j+1} &= c_{ij} - \frac{c_{ij} a_{q+1}}{p'a_{q+1}^2} p, \quad i = 1, \ldots, q-1.
\end{align*}
\]

So far we have assumed that \( C \) is positive definite. If this is not the case and if \( s_j = c_{q+1, j} \) it can happen that \( s_j' C s_j \leq 0 \) since then \( Q(x_j - \sigma s_j) \)
is linear or strictly concave no optimal step size exists and a new constraint becomes active at $x_{j+1}$. Since $c'_{q_j} C c_{q_j} < 0$ we cannot apply Lemma 1 directly. However, we observe that the vectors (2.10) obtain by applying Lemma 1 to $c_{q+1,j}, \ldots, c_{n,j}$ together with

$$c_{q+1,j+1} = c_{q,j} - \frac{c'_{q_j} a_{q+1,j+1}}{p' a_{q+1,j+1}} p$$

form a basis of conjugate directions for $T_{j+1}^*$. Indeed, since all these vectors are in $T_{j+1}^*$ and since the $q_i$ are conjugate it suffices to show that

$$q'_i C c_{q_j} = q'_i C p = 0, \quad i = 2, \ldots, r.$$  

Observing that $T_{j+1}^* \subset \{ x \mid a'_i x = 0, \quad i = 1, \ldots, q \}$ we conclude from (2.6) and the definition of $D_{j+1}^{-1}$ that $q'_i C c_{q_j} = 0$ for all $q \in T_{j+1}$. Furthermore, since $d'q'_1 = a'q'_1$ it follows from (2.9) and part i) of the lemma that $q'_i C p = 0$ for $i = 2, \ldots, r$.

Since $p' C c_{q_j} = 0$, we have

$$(2.12) \quad c'_{q+1,j+1} C C_{q+1,j+1} = c'_{q,j} C c_{q,j} + \frac{(c'_{q,j} a_{q+1,j+1})^2}{p' a_{q+1,j+1}} p' C p.$$  

If this number is positive, then $C$ is positive definite on the subspace $T_{j+1}^*$. As in the case of a positive definite matrix $C$, the matrix $D_{j+1}$ is given by (2.11). However, since $c_{q+1,j+1} C$ is constructed differently $p$ and $c_{q+1,j+1}$ are in general not conjugate. This results in a more complicated formula for the vectors $c_{1,j+1}, \ldots, c_{q,j+1}$. (See Step 7d of the algorithm). If (2.12) is not positive we choose

$$D'_{j+1} = (a'_1, \ldots, a'_{q-1}, a'_{q+1,1} a'_{q+2,1} C c_{q+2,1}, \ldots, 0_{n,j+1} C C_{n,j+1}).$$
Then (see Step 7c of the algorithm)

\[ c_{i,j+1} = c_{ij} - \frac{c_{ij} a_{q+1}}{p_i a_{q+1}} \quad i = 2, \ldots, r. \]

Note that \( a_q \) is the \((q+1)\)-th column of \( D^j_{q+1} \) even though the \( q \)-th constraint is not active at \( x^j_{q+1} \). In order to correct this situation we choose \( s_{j+1} \) parallel to \( c_{q+1,j+1} \), since \( Q(x^j_{j+1} - \sigma s^j_{j+1}) \) is concave no optimal step size exists and an additional constraint is active at \( x^j_{j+2} \). Thus after a finite number of steps we have either an extreme point (Step 7a of the algorithm) or a positive number \((2,12)\). (Step 7d of the algorithm). In either case \( a_q \) is removed from the corresponding matrix \( D^j_{j+q} \).

3. Detailed statement of the algorithm.

It is assumed that the algorithm starts with a feasible extreme point of the set \( \{x|Ax \leq b\} \). If such a point is not available it can be determined by solving a linear programming problem.

We describe now a general cycle of the algorithm. At the beginning of the \( j \)-th cycle the following data are available: a feasible \( x_j \), the gradient \( g_j = c + Cx_j \) of \( Q(x) \) at \( x_j \), the numbers \( \beta_j = 0 \) or 1, \( \delta_j = 0 \) or 1, the matrix \( D^{-1}_j = (c_{1j}, \ldots, c_{nj}) \) and the set \( J(x_j) = \{a_{1j}, \ldots, a_{nj}\} \).

The \( a_{ij} \)'s are nonnegative integers. If \( a_{ij} = 0 \), then \( c_{ij} \) is a conjugate direction. If \( a_{ij} > 0 \), then the constraint with subscript \( a_{ij} \) is active at \( x_j \) and the \( i \)-th column of \( D_j \) is equal to the gradient of this constraint.

\( \beta_j = 1 \) if and only if a new constraint became active at \( x_j \). \( \delta_j \) is equal to 1 if and only if \( s_j = c_{ij} \) is a search direction along which \( Q(x) \) is not strictly convex. Step 7 of the algorithm is used to deal with this situation.

Finally, \( \gamma_j = 0 \) or -1 is determined in Step 1 of the algorithm. \( \gamma_j = 0 \) if
and only if the search direction $s_j$ is such that all constraints active at $x_j$ will also be active at $x_{j+1}$.

Step 1.

a) Compute

$$c_{ij}^t g_j$$ for all $i$ with $\alpha_{ij} = 0$.

If

$$c_{ij}^t g_j = 0$$ for all $i$ with $\alpha_{ij} = 0$

go to Step 2b, otherwise set

$$s_j = \sum_{\alpha_{ij} > 0} (\theta_{ij} c_{ij}^t g_j) c_{ij}, \quad \gamma_j = 0$$

and go to Step 2.

b) Compute $\ell$ such that

$$c_{ij}^t g_j > c_{ij}^t g_j$$ for all $i$ with $\alpha_{ij} > 0$.

If $c_{ij}^t g_j \leq 0$, stop; otherwise set

$$s_j = c_{ij}^t, \quad \gamma_j = -1$$

and go to Step 2.

Step 2.

Compute

$$a_i^t s_j, \quad i = 1, \ldots, m.$$ 

If

$$a_i^t s_j \geq 0, \quad i = 1, \ldots, m$$

set

$$\sigma_j^* = \infty;$$

otherwise compute $k$ such that

$$\frac{a_k^t x_j - (b)_k}{a_k^t s_j} \leq \frac{a_i^t x_j - (b)_i}{a_i^t s_j} \quad \text{for all } i \text{ with } a_i^t s_j < 0.$$
and set

\[ \sigma_j^* = \frac{a_j' x - (b)_j}{a_j' s_j} \]

Set

\[ \hat{\sigma}_j = \begin{cases} 
1 & \text{if } \gamma_j = 0 \\
\infty & \text{if } s_j' C s_j \leq 0 \text{ and } \gamma_j = -1 \\
g_j's_j & \text{if } s_j' C s_j > 0 \text{ and } \gamma_j = -1 
\end{cases} \]

Set

\[ \sigma_j = \min \{\sigma_j^*, \hat{\sigma}_j\} \]
\[ x_{j+1} = x_j - \sigma_j s_j, \quad g_{j+1} = C x_{j+1} \]
\[ \beta_{j+1} = \begin{cases} 
1 & \text{if } \sigma_j^* \leq \hat{\sigma}_j \\
0 & \text{if } \sigma_j^* > \hat{\sigma}_j 
\end{cases} \]

If \( \delta_j = 1 \) go to Step 7a, otherwise do the following. If \( \beta_{j+1} = 0 \) and \( \gamma_j = 0 \), go to Step 3. If \( \beta_{j+1} = 0 \) and \( \gamma_j = -1 \), go to Step 4. If \( \beta_{j+1} = 1 \) and \( \gamma_j = 0 \), go to Step 5a. If \( \beta_{j+1} = 1 \) and \( \gamma_j = -1 \), go to Step 6.

**Step 3.** \( (\beta_{j+1} = \gamma_j = 0, \text{ no change in set of active constraints}) \)

Set

\[ D_{j+1}^{-1} = D_j^{-1}, \quad J(x_{j+1}) = J(x_j) \quad \text{and} \]
\[ \theta_{i,j+1} = \theta_{i,j} \quad \text{for all } i \text{ with } \alpha_{ij} = 0 \]

Replace \( j \) with \( j+1 \) and go to Step 1b.

**Step 4.** \( (\beta_{j+1} = 0, \gamma_j = -1 \text{ "dropping a constraint"}) \)

Set

\[ c_{i,j+1} = c_{ij} \quad \text{for all } i \text{ with } \alpha_{ij} = 0 \]
\[ c_{l,j+1} = c_{lj} \]
\[ c_{i,j+1} = c_{i,j} - \frac{c'_{ij}(g_j - g_{j+1})}{c'_{lj}(g_j - g_{j+1})} c_{lj} \quad \text{for all } i \neq \ell \text{ with } a_{ij} > 0 \]

\[ a_{i,j+1} = a_{ij}, \quad i = 1, \ldots, n, \quad i \neq \ell \]

\[ a_{\ell,j+1} = 0 \]

\[ \theta_{i,j+1} = \theta_{ij} \quad \text{for all } i \text{ with } a_{ij} = 0 \]

\[ \theta_{\ell,j+1} = \{s'_j \subseteq s_j \}^{-1} \]

\[ D_{j+1}^{-1} = (c_{1,j+1}, \ldots, c_{n,j+1}), \quad J(x_{j+1}) = \{a_{1,j+1}, \ldots, a_{n,j+1}\}. \]

Replace \( j \) with \( j+1 \) and go to Step 1b.

**Step 5.** (\( \beta_{j+1} = 1, \gamma_{j} = 0 \), "adding a constraint")

a) Choose any \( \ell \in \{1, \ldots, n \} \) such that \( a_{\ell j} = 0 \) and compute

\[ p_j = \sum_{a_{ij} = 0} (\theta_{ij} c'_i a_k) c_{ij}, \]

Set \( a_{\ell j} = k \) and go to Step 5b),

b) Compute \( \omega_j = p'_i a_k \). If \( \omega_j - \theta_{lj} (c'_j a_k)^2 = 0 \), set

\[ c_{i,j+1} = c_{ij} \quad \text{for all } i \text{ with } a_{ij} = 0 \]

and go to Step 5c), otherwise compute

\[ t_j = \frac{-\theta_{lj} c'_j a_k + \sqrt{\theta_{lj} \omega_j}}{\omega_j - \theta_{lj} (c'_j a_k)^2}, \]

set

\[ c_{i,j+1} = c_{ij} - \frac{c'_j a_k}{\omega_j} (1 - t_j c'_j a_k) p_j - (t_j c'_j a_k) c_{lj} \]

for all \( i \) with \( a_{ij} = 0 \) and go to Step 5c)

c) Set

\[ c_{\ell,j+1} = \frac{p_j}{\omega_j} \]

\[ c_{i,j+1} = c_{ij} - \frac{c'_i a_k}{\omega_j} \quad \text{for all } i \neq \ell \text{ with } a_{ij} > 0 \]
\[
\alpha_{i,j+1} = \alpha_{i,j}, \quad i = 1, \ldots, n, \quad i \neq \ell
\]

\[
\alpha_{\ell,j+1} = k
\]

\[
\theta_{i,j+1} = \theta_{i,j} \quad \text{for all } i \text{ with } \alpha_{i,j+1} = 0
\]

\[
D_{j+1}^{-1} = (c_{1,j+1}, \ldots, c_{n,j+1}), \quad J(x_{j+1}) = \{\alpha_{1,j+1}, \ldots, \alpha_{n,j+1}\}.
\]

Replace \( j \) with \( j+1 \) and go to Step 1a.

**Step 6.** \( (\beta_{j+1} = 1, \ \gamma_{j} = -1 "adding and dropping a constraint") \)

If \( s^{\prime}C_{s} \leq 0 \), go to Step 7a, otherwise set \( \delta_{j+1} = 0 \) and \( \theta_{\ell,j} = (s^{\prime}C_{s})^{-1} \).

Compute

\[
p_{j} = \sum_{\alpha_{i,j} = 0} (\theta_{i,j}c_{i}a_{i}^{c_{i}})c_{i,j} + (\theta_{i,j}c_{i}a_{i}^{c_{i}})c_{\ell,j}
\]

and go to Step 5b).

**Step 7.**

a) If \( \alpha_{i,j} = 0 \) for at least one \( i \) go to Step 7b), otherwise compute

\[
c_{\ell,j+1} = \frac{c_{\ell,j}}{c_{\ell,j}a_{k}^{c_{\ell,j}}}
\]

\[
c_{i,j+1} = c_{i,j} - \frac{c_{i}a_{i}^{c_{i}}}{c_{i,j}a_{k}^{c_{i,j}}}c_{\ell,j}, \quad i = 1, \ldots, n, \quad i \neq \ell
\]

Set

\[
\beta_{j+1} = 1, \quad \delta_{j+1} = 0
\]

\[
\alpha_{i,j+1} = \alpha_{i,j}, \quad i = 1, \ldots, n, \quad i \neq \ell, \quad \alpha_{\ell,j+1} = k
\]

\[
J(x_{j+1}) = \{\alpha_{1,j+1}, \ldots, \alpha_{n,j+1}\}, \quad D_{j+1}^{-1} = (c_{1,j+1}, \ldots, c_{n,j+1}),
\]

Replace \( j \) with \( j+1 \) and go to Step 1b.

b) Choose any \( r \in \{1, \ldots, n\} \) such that \( \alpha_{r,j} = 0 \), and compute

\[
p_{j} = \sum_{\alpha_{i,j} = 0} (\theta_{i,j}c_{i}a_{i}^{c_{i}})c_{i,j}, \quad \omega_{j} = p_{j}a_{k}
\]

and
\[ c_{r,j+1} = \frac{p_j}{\omega_j} \]

If \( \omega_j - 0 \), \( (c'_r a'_r) \) \( r_j k \) \( = 0 \), set
\[
c_{i,j+1} = c_{i,j}, \quad \text{for all } i \neq r \text{ with } a_{ij} = 0
\]
and go to Step 7c, otherwise compute
\[
t_j = \frac{-\theta_{r_j} c'_r a'_r + \sqrt{\theta_{r_j}} \omega_j}{\omega_j - \theta_{r_j} (c'_r a'_r)}
\]
Set
\[
c_{i,j+1} = c_{i,j} - \frac{c'_r a'_r}{\omega_j} (1 - t_{r_j} c'_r a'_r) p_j - (t_{r_j} c'_r a'_r) c_{r_j}
\]
for all \( i \neq r \) with \( a_{ij} = 0 \) and go to Step 7c.

C) Compute
\[
c_{\ell,j+1} = c_{\ell,j} - \frac{c'_r a'_r}{\omega_j} p_j \quad \text{and} \quad c_{\ell,j+1} = c_{\ell,j+1}.
\]
If \( c'_{\ell,j+1} C c_{\ell,j+1} > 0 \), set \( \delta_{j+1} = 0 \) and go to Step 7d), otherwise set
\[
\delta_{j+1} = 1 \quad \text{compute}
\]
\[
c_{i,j+1} = c_{i,j} - \frac{c'_r a'_r}{\omega_j} p_j \quad \text{for all } i \neq \ell \text{ with } a_{ij} > 0
\]
and go to Step 7e)

D) Set
\[
\theta_{\ell,j+1} = [c'_{\ell,j+1} C c'_{\ell,j+1}]^{-1}
\]
\[
\rho_{ij} = \theta_{\ell,j+1} [c'_{\ell,j+1} C c_{ij} - \frac{c'_r a'_r}{\omega_j} c'_{\ell,j+1} C p_j]
\]
and
\[
c_{i,j+1} = c_{i,j} - \frac{c'_r a'_r}{\omega_j} p_j - \rho_{ij} c'_{\ell,j} + 1 \quad \text{for all } i \neq \ell \text{ with } a_{ij} > 0
\]
and go to Step 7e).
e) Set \( \beta_{j+1} = 1, \ \gamma_{j+1} = -1 \)
\( a_{i,j+1} = a_{ij}, \quad i = 1, \ldots, n, \ i \neq \ell, \ i \neq r \)
\( a_{r,j+1} = k \)
\( a_{\ell,j+1} = 0 \quad \text{if} \ \delta_{j+1} = 1 \)
\( a_{\ell,j+1} = 0 \quad \text{if} \ \delta_{j+1} = 0 . \)

Set \( \theta_{l,j+1} = 0_{ij} \) for all \( i \neq \ell \) with \( a_{l,j+1} = 0 \)

\( J(x_{j+1}) = \{a_{1,j+1}, \ldots, a_{n,j+1}\}, \ \mathbf{D}_{j+1}^{-1} = (c_{1,j+1}, \ldots, c_{n,j+1}) \)

If \( \delta_{j+1} = 0 \), replace \( j \) with \( j+1 \) and go to Step 1. If \( \delta_{j+1} = 1 \), set

\( c_{\ell,j+1} \quad \text{if} \quad c_{\ell,j+1} g_{j+1} \geq 0 \)
\( s_{j+1} = -c_{\ell,j+1} \quad \text{if} \quad c_{\ell,j+1} g_{j+1} < 0 . \)

Replace \( j \) with \( j+1 \) and go to Step 2.

Remark.

It follows immediately from Step 2, that every \( x_j \) is feasible. Furthermore, if \( s_j \) is determined by Step 1, then \( g_j s_j > 0 \) and \( Q(x_{j+1}) < Q(x_j) \) unless \( \sigma_j = 0 \). This could happen if several new constraints became active at \( x_j \) since only one is used in the update procedure for \( D_{j-1}^{-1} \). If \( s_j \) is computed in Step 7 of the algorithm, then \( Q(x_j - \sigma s_j) \) is concave and \( g_j s_j \geq 0 \). Thus we have always \( Q(x_{j+1}) \leq Q(x_j) \).
4. Termination properties.

In this section we prove that the algorithm terminates after a finite number of steps with a stationary point. The following lemma establishes the properties of the matrix $D_j^{-1}$ on which this result is based.

**Lemma 2.**

Let $D_j^{-1} = (c_{ij}, \ldots, c_{nj})$, $J(x_j) = \{\alpha_{ij}, \ldots, \alpha_{nj}\}$ and $\theta_{ij}$ be determined by the algorithm. Set

$$T_j = \{x \mid a_i^T x = 0 \text{ for all } i \text{ with } \alpha_{ij} > 0\}.$$ 

Then

i) $T_j = \text{span}\{c_{ij} \mid \text{all } i \text{ with } \alpha_{ij} = 0\}$

$$c_{ij}^T C c_{ij} = 0 \quad \text{for all } i \neq k$$

with $\alpha_{ij} = \alpha_{kj} = 0$,

i.e. the vectors $c_{ij}$, for all $i$ with $\alpha_{ij} = 0$, form a basis of conjugate directions for $T_j$.

ii) $c_{kj}^T C c_{ij} = 0$ for all $k$ with $\alpha_{kj} > 0$ and all $i$ with $\alpha_{ij} = 0$.

iii) $a_i^T c_{ij} = 1$, $a_i^T c_{kj} = 0$ for all $i \neq k$ with $\alpha_{ij} > 0$ and $\alpha_{kj} > 0$.

**Proof.**

The proof is by induction. Since $x_0$ is an extreme point the lemma is true for $j = 0$. Suppose it is true for $j \geq 0$. If Step 3 of the algorithm applies $D_j^{-1} = D_j^{-1}$ and there is nothing to prove. In the case that Step 4 of the algorithm is used we have

$$T_{j+1} = \{x \mid a_i^T x = 0 \text{ for all } i \neq \ell \text{ with } \alpha_{ij} > 0\}.$$
and
\[ c_{i,j+1} = c_{ij} \] for \( i = \ell \) and all \( i \) with \( \alpha_{ij} = 0 \).

This proves the first part of the lemma. Furthermore, \( c_{'i,j+1} C c_{k,j+1} = 0 \) for any \( k \neq \ell \) with \( \alpha_{kj} > 0 \) and any \( i \) with \( \alpha_{ij} = 0 \) because \( c_{'i,j} C c_{k,j} = 0 \) and \( c_{'i,j} C c_{\ell,j} = 0 \). The last statement of the lemmas follows from the observation that \( a_{'i,j} C c_{\ell,j} = 0 \) for any \( i \neq \ell \).

Now suppose \( D_{j+1}^{-1} \) is determined by either Step 5, 6 or 7c. Then the first part of the lemma follows from Lemma 1. For every \( i \neq \ell \) with \( \alpha_{ij} = 0 \) and every \( v \) with \( \alpha_{vj} > 0 \) we have
\[
 c_{i,j+1} C c_{v,j+1} = c_{i,j+1} C c_{v,j} - \frac{c_{'v,j} C a_k C c_{ij}}{\omega_j} C c_{i,j+1} C p_j .
\]
The first term on the right hand side of this equality is zero since
\[
 c_{i,j+1} \in \text{span} \{ c_{ij} \mid \text{all } i \text{ with } \alpha_{ij} = 0 \},
\]
the second term is zero because, by Lemma 1, \( c_{i,j+1} \) and \( p_j \) are conjugate with respect to \( C \). The last statement of the lemma is obvious since \( p_j \in T_j \).

If Step 7a applies, then \( x_{j+1} \) is an extreme point and \( D_{j+1}^{-1} \) has the required properties. Finally assume that Step 7d is used. Part i) of the lemma is a consequence of Lemma 1 and the equality
\[
 c_{\ell,j+1} C c_{i,j+1} = c_{\ell,j} C c_{i,j+1} - \frac{c_{'v,j} C a_k C c_{ij}}{\omega_j} \]
for all \( i \) with \( \alpha_{ij} = 0 \). Furthermore,
\[
 c_{i,j+1} C c_{v,j+1} = c_{i,j+1} C c_{v,j} - \frac{c_{'v,j} C a_k C c_{ij}}{\omega_j} C c_{i,j+1} C c_{\ell,j+1} = 0
\]
for every \( i \) with \( \alpha_{ij} = 0 \) and every \( v \neq \ell \) with \( \alpha_{vj} > 0 \). The last statement of the lemma is true because
\[
 a_{'ij} C p_j = 0 \quad \text{and} \quad a_{'ij} C c_{\ell,j+1} = 0
\]
for every \( i \neq \ell \) with \( \alpha_{ij} > 0 \).
The following two lemmas give conditions under which \( x_{j+1} \) is a quasi-stationary point.

**Lemma 3.**

i) If \( \sigma_j < \sigma^* \) and \( s_j \) is determined by Step 1a, then \( x_{j+1} \) is a quasi-stationary point.

ii) If \( \sigma_j < \sigma^* \), \( x_j \) is a quasi-stationary point and \( s_j \) is determined by Step 1b, then \( x_{j+1} \) is a quasi-stationary point.

**Proof.**

i) It suffices to show that \( g_{ij}^1 c_i = 0 \) for all \( i \) with \( a_{ij} = 0 \). We have

\[
c_{kj}^{j+1} = c_{kj}^j - c_{kj}^0 C s_j = c_{kj}^j - c_{kj}^0 C \left( \sum_{a_{ij} = 0} \left( \sum_{ij} (0, c_{ij}^0) c_{ij} \right) \right)
\]

for every \( k \) with \( a_{kj} > 0 \).

ii) Since \( x_j \) is a quasi-stationary point

\[
c_{ij}^0 g_j = 0 \quad \text{for all } i \text{ with } a_{ij} = 0.
\]

Thus \( s_j = c_{kj}^0 \) and \( c_{ij}^0 C c_{kj}^0 = 0 \) for all \( i \) with \( a_{ij} = 0 \) implies

\[
c_{ij}^0 g_{j+1} = 0 \quad \text{for all } i \text{ with } a_{ij} = 0.
\]

Finally, \( c_{ij}^0 g_{j+1} = 0 \) since \( \sigma_j = g_j^1 s_j / s_j^1 C s_j \).

**Lemma 4.**

i) If \( \sigma_j < \sigma^* \), then \( x_{j+1} \) is a quasi-stationary point.

ii) If \( x_j \) is not a quasi-stationary point, then either \( x_{j+1} \) is a quasi-stationary point or there are more constraints active at \( x_{j+1} \) than there are at \( x_j \).

iii) For every \( j \) at least one of the points \( x_j, x_{j+1}, \ldots, x_{j+n-1} \) is a quasi-stationary point.
Proof.

1) If \( \sigma_j < \sigma^* \), then \( s_j \) cannot be determined by Step 7e. Indeed, if \( s_j \) is determined by Step 7e, then \( \gamma_j = -1 \) and \( s_j C s_j \leq 0 \). Thus \( \sigma_j = \infty \) and \( \sigma_j = \sigma^* \).

Since \( x_0 \) is a quasi-stationary point, it follows from Lemma 3, that the statement is true for \( j = 0 \). Now assume that the statement is true for \( j - 1 \). Then either \( x_j \) is a quasi-stationary point or \( s_j \) is determined by Step 1a). In either case it follows from Lemma 3, that \( x_{j+1} \) is a quasi-stationary point provided \( \sigma_j < \sigma^* \).

2) If \( x_j \) is not a quasi-stationary point, then by the first part of the lemma, \( \sigma_{j-1} = \sigma_{j-1}^* \). Therefore, \( s_j \) is either determined by Step 1a or Step 7e. In either case all constraints that are active at \( x_j \) will also be active at \( x_{j+1} \) and the statement follows from part 1) of the lemma.

3) Since every extreme point of \( \{x \mid Ax \leq b\} \) is a quasi-stationary point, the last statement of the lemma follows immediately from part ii).

Theorem.

The algorithm terminates after a finite number of steps with a stationary point.

Proof.

Suppose the algorithm terminates with \( x_j \). By Step 1 of the algorithm either \( \sigma_{j-1} < \sigma_{j-1}^* \) or \( c_{ij} g_j = 0 \) for all \( i \) with \( a_{ij} = 0 \). Thus using part 1) of Lemma 4 we see that \( x_j \) is a quasi-stationary point. This means that there are numbers \( \lambda_i \) such that

\[
g_j = \nabla Q(x_j) = \sum_{a_{ij} > 0} \lambda_i a_{ij}
\]
Since by Lemma 2, and Step 1 of the algorithm

\[ \lambda_i = g'_i c_{ij} \leq c'_j g_j \leq 0 \quad \text{for all } i \text{ with } a_{ij} > 0 , \]

the Kuhn-Tucker-conditions are satisfied; i.e., \( x_j \) is a stationary point.

If \( C \) is positive definite, then there are only finitely many quasi-stationary points. If \( C \) is not positive definite then there can be infinitely many quasi-stationary points. However, there are only finitely many with different values of \( Q(x) \). If \( x_j \) is any quasi-stationary point such that \( x_{j+1} \neq x_j \), then \( Q(x_{j+1}) < Q(x_j) \). Therefore, it follows from part iii) of Lemma 4, that the algorithm terminates after a finite number of steps.

Remark.

A numerical study involving the method given in this paper and several other algorithms for quadratic minimization problems is currently undertaken at the University of Waterloo. The results will be reported elsewhere.

References


A method of conjugate directions is described which solves a quadratic minimization problem in a finite number of steps.