Annals of Operations Research 81(1998)75–95

A block active set algorithm for large-scale quadratic programming with box constraints

L. Fernandes[a], A. Fischer[b], J. Júdice[c], C. Requejo[c] and J. Soares[c]

[a]*Escola Superior de Tecnologia de Tomar, 2300 Tomar, Portugal*

[b]*Institut für Numerische Mathematik, Technische Universität Dresden, D-01062 Dresden, Germany*

[c]*Departamento de Matemática da Universidade de Coimbra, 3000 Coimbra, Portugal*

An algorithm for computing a stationary point of a quadratic program with box constraints (BQP) is proposed. Each iteration of this procedure comprises a guessing strategy which forecasts the active bounds at a stationary point, the determination of a descent direction by means of solving a reduced strictly convex quadratic program with box constraints and an exact line search. Global convergence is established in the sense that every accumulation point is stationary. Moreover, it is shown that the algorithm terminates after a finite number of iterations, if at least one iterate is sufficiently close to a stationary point which satisfies a certain sufficient optimality condition. The algorithm can be easily implemented for sparse large-scale BQPs. Furthermore, it simplifies for concave BQPs, as it is not required to solve strictly convex quadratic programs in this case. Computational experience with large-scale BQPs is included and shows the appropriateness of this type of methodology.

Keywords: quadratic programming, box constraints, large-scale problems, sparse matrices

1. Introduction

In this paper we consider the box constrained quadratic program (BQP)

$$f(x) := q^T x + \tfrac{1}{2} x^T M x \to \min \quad \text{subject to } x \in K, \tag{1}$$

where $q \in \mathbb{R}^n$, $M \in \mathbb{R}^{n\times n}$ is a symmetric matrix, the feasible set

$$K := \{x \in R^n \mid l_i \le x_i \le u_i, i \in I\} \quad \text{with } I := \{1, \ldots, n\}$$

is assumed to be nonempty, and all lower and upper bounds are finite. Finding a global minimum of the program (1) is in general a very hard problem when M is an indefinite or negative semi-definite matrix. Enumerative approaches [2, 10, 14, 20] have been proposed for the solution of this problem. These procedures require good lower and

upper bounds to alleviate the overall search for a global minimum. Algorithms that are able to efficiently find local minima (or stationary points that are not local maxima) are quite important in this context and have been proposed in the literature [1, 3–5, 8, 13, 18]. In this paper, we suggest a new algorithm which combines ideas from active set and Newton-type methods. The algorithm can be considered as an application of the active set Newton's method (ASN) suggested in [21] for general nonlinear programs with box constraints to the case of quadratic programming. Both algorithms generate only feasible iterates. However, there are some important differences. The first is that the new algorithm employs an exact line search to determine the stepsize and reduces the objective function in each step. In contrast to this, the ASN method combines an Armijo-type line search with a nonmonotone stabilization technique, i.e., the objective function does not necessarily decrease monotonously. In order to generate subproblems with a strictly convex quadratic function, the ASN method includes a technique for perturbing the occurring Hessians of the nonlinear objective function. The algorithm to be presented in this paper avoids such perturbations. The new algorithm is also able to exploit concave parts within the objective function in order to reduce the size of the subproblem that has to be solved in each iteration. In particular, concave BQPs (i.e., M is negative semi-definite) can be processed without the solution of strictly convex quadratic programs.

Starting with a feasible vector x^0, each step of the new algorithm is of the general form

$$x^{k+1} = x^k + \theta_k d^k, \quad k = 0, 1, \ldots.$$

The search direction d^k is computed in two main steps. First, some components of d^k are set by means of a guessing technique which forecasts the active bounds at a stationary point close to x^k. Then the remaining components of d^k are determined by solving a strictly convex box-constrained quadratic program. The Hessian of the objective function belonging to this latter program is a positive definite principal submatrix of M, and the constraints guarantee that $x^k + d^k$ is feasible. The stepsize θ_k is determined by an exact line search which computes the minimum of the objective function f along the direction d^k. Since f is quadratic, the numerical expense required by this line search is very low in comparison with an inexact, e.g., Armijo-type line search.

After describing the algorithm in section 2, global convergence to a stationary point is established in section 3. The analysis is similar to the one employed in [21]. Section 4 presents a result concerning the finite termination of the algorithm. If an iterate x^k is sufficiently close to a stationary point which satisfies a sufficient optimality condition, i.e., a second-order condition together with strict complementarity, then we show that the algorithm yields this stationary point after a finite number of iterations.

In section 5, we will deal with the case when the matrix M occurring in the objective function f is negative semi-definite. Then, no strictly convex quadratic

program has to be solved during the entire algorithm. As a consequence of this, some additional simple rules for guessing the active bounds provide a reduction in the size of the subproblems in the indefinite case. Section 6 reports the results of our computational experiments with the algorithm. As a first step, the quadratic programs with box constraints contained in the CUTE collection [6] were tested. Since these problems are convex, ten test matrices of different sizes (positive definite, negative definite and indefinite) were used for generating test problems such that both the magnitude of the bounds ($\max_{i\in I}\{u_i - l_i\}$) and the number of active bounds can be controlled. Moreover, the same test problems were solved by the aformentioned ASN method [21] and two further well-known codes, see [13, 15, 17]. The results of the experiments indicate that the new algorithm is quite efficient for solving all the test problems. In particular, it is a significant improvement in comparison with the ASN algorithm [21].

The following notation will be used throughout the paper. Let $J, K \subseteq I$ be arbitrary index sets. If the matrix M has the elements m_{jk} with $j, k \in I$, then M_{JK} denotes the submatrix of M consisting of all m_{jk} with $j \in J$ and $k \in K$. In analogy to this, for any $x \in \mathbb{R}^n$, the subvector x_J contains the elements x_j with $j \in J$. A superscript (k in general) will be used as iteration index. Finally, $\|\cdot\|$ stands for the euclidean vector norm or the corresponding subordinate matrix norm.

2. The algorithm

The algorithm we are going to present in this paper aims at finding a stationary point of the program (1) such that the first-order necessary conditions (2) for a local minimum are satisfied. Let us recall that a vector $\bar{x} \in K$ is said to be a stationary point of problem (1), if

$$\begin{aligned} &\nabla_i f(\bar{x}) \geq 0 \quad \text{for all } i \in \overline{L} := \{i \in I \mid l_i = \bar{x}_i\}, \\ &\nabla_i f(\bar{x}) = 0 \quad \text{for all } i \in \overline{F} := \{i \in I \mid l_i < \bar{x}_i < u_i\}, \\ &\nabla_i f(\bar{x}) \leq 0 \quad \text{for all } i \in \overline{U} := \{i \in I \mid \bar{x}_i = u_i\}, \end{aligned} \tag{2}$$

where ∇f is the gradient vector function of f. Strict complementarity is said to hold at $\bar{x}$ if $\nabla_i f(\bar{x}) > 0$ for all $i \in \overline{L}$ and $\nabla_i f(\bar{x}) < 0$ for all $i \in \overline{U}$. Additionally, if $\bar{x}$ is a local minimum, then the following necessary condition holds:

$$M_{\overline{F}\,\overline{F}} \text{ is a positive semi-definite matrix.} \tag{3}$$

Thus, for concave quadratic programs, the set $\overline{F}$ must be empty and this implies an interesting property, namely all the components of any local minimum are equal to a certain bound l_i or u_i. For further discussion and consequences, see section 5. We now describe the algorithm, starting with the definition of the search direction d^k at a current feasible iterate x^k. The direction d^k is computed in two stages. At first, based on the following forecasts of the sets $\overline{L}$ and $\overline{U}$,

$$\begin{aligned} L(x^k) &:= \{i \in I \,|\, x_i^k \le l_i + a_i(x^k)\nabla_i f(x^k)\}, \\ U(x^k) &:= \{i \in I \,|\, x_i^k \ge u_i + b_i(x^k)\nabla_i f(x^k)\}, \end{aligned} \tag{4}$$

the corresponding components of d^k are set as follows:

$$d_i^k := l_i - x_i^k \qquad i \in L(x^k), \tag{5}$$

$$d_i^k := u_i - x_i^k \qquad i \in U(x^k). \tag{6}$$

The functions a_i and b_i are required to be nonnegative, continuous and bounded above on K such that $x_i = l_i$ or $x_i = u_i$ implies $a_i(x) > 0$ or $b_i(x) > 0$, respectively. If strict complementarity holds at a stationary point $\bar{x}$, the sets defined in (4) locally coincide with $\overline{L}$ and $\overline{U}$, respectively [21]. Otherwise, the guessing technique is, nevertheless, a good forecast. To explain how the remaining components d_i^k of the search direction d^k are determined, let us consider the following partition of the index set $F(x^k) := I \setminus (L(x^k) \cup U(x^k))$:

$$F(x^k) = F^k \cup L^k \cup U^k \cup Z^k. \tag{7}$$

The index set F^k has to be chosen as large as possible such that $M_{F^kF^k}$ is a positive definite matrix. Then the remaining index sets L^k, U^k and Z^k are defined by

$$\begin{aligned} L^k &:= \{i \in (F(x^k)\setminus F^k) \,|\, \nabla_i f(x^k) > \delta\}, \\ U^k &:= \{i \in (F(x^k)\setminus F^k) \,|\, \nabla_i f(x^k) < -\delta\}, \\ Z^k &:= \{i \in (F(x^k)\setminus F^k) \,|\, |\nabla_i f(x^k)| \le \delta\} \end{aligned}$$

for some positive scalar δ. The set F^k may be determined while performing an LDL^T Cholesky factorization of the matrix $M_{F(x^k)F(x^k)}$. The standard way to implement the Cholesky factorization is an n-step process, which determines a diagonal element of D and a column of L at each step. When a diagonal element of D, say d_{ii}, is found to be not (sufficiently) positive, then that particular step of the factorization is ignored, i.e., the row and column i of $M_{F(x^k)F(x^k)}$ are discarded. At the end of the process, the LDL^T factorization of a positive definite principal submatrix $M_{F^kF^k}$ of $M_{F(x^k)F(x^k)}$ is obtained. With regard to the partition (7) of $F(x^k)$, we define the following components of d^k:

$$d_i^k := l_i - x_i^k \qquad i \in L^k, \tag{8}$$

$$d_i^k := u_i - x_i^k \qquad i \in U^k, \tag{9}$$

$$d_i^k := \mathrm{proj}(-\nabla_i f(x^k), [l_i - x_i^k, u_i - x_i^k]) \quad i \in Z^k. \tag{10}$$

where $\mathrm{proj}(\cdot, [a, b])$ is the one-dimensional projection into the interval $[a, b]$. Finally, the remaining part $d_{F^k}^k$ of the search direction is defined as the unique solution of the following reduced strictly convex quadratic program with box constraints:

$$m(d_{F^k}) = \nabla_{F^k} f(x^k)^T d_{F^k} + \tfrac{1}{2} d_{F^k}^T M_{F^k F^k} d_{F^k} \to \min$$
$$\text{subject to } (l - x^k)_{F^k} \le d_{F^k} \le (u - x^k)_{F^k}. \tag{11}$$

Since this program is strictly convex, it can be solved by the Block Principal Pivoting algorithm suggested in [16]. The numerical tests in that paper and elsewhere [12,21] indicate that this algorithm is particularly effective for solving large-scale strictly convex quadratic programs with box constraints.

It will be shown in lemma 1 that the direction d^k is a descent direction. Therefore, the stepsize θ_k defined as the global solution of the following one-dimensional minimization problem

$$\varphi(\theta) := f(x^k + \theta d^k) \to \min \quad \text{subject to } \theta \in [0,1] \tag{12}$$

is positive. Since the objective function φ is quadratic, the solution θ_k can easily be computed by

$$\theta_k = \begin{cases} 1 & \text{if } d^{k^T} M d^k \le 0, \\ \min\left(1, -\dfrac{\nabla f(x^k)^T d^k}{(d^k)^T M d^k}\right) & \text{if } d^{k^T} M d^k > 0. \end{cases} \tag{13}$$

Based on these considerations, we get the following algorithm BAS for solving the box constrained quadratic program (1).

Block Active Set algorithm (BAS)

Data: Choose $x^0 \in K$ and $\Delta_0 \ge 0$, $\beta \in (0, 1)$, $\delta > 0$.
Define the functions a_i and b_i for $i \in I$.

Initialization: Set $k = 0$ and $j = 0$.

Iteration:

Step 1. Compute d^k according to (5), (6), (8)–(10) and (11).
If $d_k = 0$, then stop.

Step 2. If ($\| d^k \| < \Delta_j$) and $f(x^k + d^k) < f(x^k)$,
then set $\theta_k = 1$, $\Delta_{j+1} = \beta \Delta_j$ and $j = j + 1$.
Otherwise, compute the stepsize θ_k according to (13).

Step 3. Set $x^{k+1} = x^k + \theta_k d^k$ and $k = k + 1$. Go to step 1.

3. Global convergence

Lemma 1 states some properties of the algorithm BAS, in particular that every search direction d^k turns out to be a descent direction of the objective function f if and only if x^k is not a stationary point.

Lemma 1. Let $\{x^k\}$ be any sequence generated by the algorithm BAS. Then, for all $k = 0, 1, \ldots$:

(1) $x^k \in K$,

(2) $\nabla f(x^k)^T d^k \leq -\gamma \| d^k \|^2$ for some fixed positive scalar γ,

(3) $d^k = 0$ if and only if x^k is a stationary point.

Proof. (1) By the definition of the direction d^k, we have

$$\begin{aligned}
&(x^k + d^k)_i = l_i && i \in L(x^k) \cup L^k,\\
&(x^k + d^k)_i = u_i && i \in U(x^k) \cup U^k,\\
&d_i^k \in [l_i - x_i^k, u_i - x_i^k] && i \in Z^k,\\
&(x^k + d^k)_i \in [l_i, u_i] && i \in Z^k.
\end{aligned}$$

Hence, $x^k + d^k \in K$ for all k. Now, since $x^0 \in K$ and $x^1 = x^0 + \theta_0 d^0$ for some $\theta_0 \in [0, 1]$, we have that $x^1 \in K$. By induction, we can conclude that $x^k \in K$ for all k.

(2) With regard to the definition of F^k, all occurring matrices $M_{F^kF^k}$ are positive definite. Since the set of the principal submatrices of M which are positive definite is finite, there exists a positive scalar ρ such that, for $k = 0, 1, \ldots$,

$$\rho \| z \|^2 \leq z^T M_{F^kF^k} z \quad z \in \mathbb{R}^{|F^k|}. \tag{14}$$

Now, as $M_{F^kF^k}$ is positive definite, the subvector $d_{F^k}^k$ is the optimal solution of (11) if and only if, for $i \in F^k$,

$$\begin{aligned}
l_i - x_i^k = d_i^k &\Rightarrow (\nabla_{F^k} f(x^k) + M_{F^kF^k} d_{F^k}^k)_i \geq 0,\\
l_i - x_i^k < d_i^k < u_i - x_i^k &\Rightarrow (\nabla_{F^k} f(x^k) + M_{F^kF^k} d_{F^k}^k)_i = 0,\\
d_i^k = u_i - x_i^k &\Rightarrow (\nabla_{F^k} f(x^k) + M_{F^kF^k} d_{F^k}^k)_i \leq 0.
\end{aligned} \tag{15}$$

Since $x^k \in K$, i.e., $l_i - x_i^k \leq 0$ and $u_i - x_i^k \geq 0$ for all $i \in I$, relations (15) yield

$$\begin{aligned}
\nabla_{F^k} f(x^k)^T d_{F^k}^k + \tfrac{1}{2} {d_{F^k}^k}^T M_{F^kF^k} d_{F^k}^k &\leq \nabla_{F^k} f(x^k)^T d_{F^k}^k + {d_{F^k}^k}^T M_{F^kF^k} d_{F^k}^k\\
&\leq (\nabla_{F^k} f(x^k) + M_{F^kF^k} d_{F^k}^k)^T d_{F^k}^k\\
&\leq 0.
\end{aligned}$$

Hence, with (14) it follows that

$$\nabla_{F^k} f(x^k)^T d_{F^k}^k \leq -\rho \| d_{F^k}^k \|^2. \tag{16}$$

Now we prove that a positive scalar γ exists such that

$$\nabla f_i(x^k) d_i^k \leq -\gamma (d_i^k)^2 \quad k = 0, 1, 2, \ldots \tag{17}$$

for all $i \in I \setminus F^k$. If $d_i^k = 0$, the inequality holds trivially. Therefore, we always assume that $d_i^k \neq 0$.

If $i \in L(x^k)$, then $x^k \in K$ and $d_i^k \neq 0$ imply $d_i^k = l_i - x_i^k < 0$. Thus, the definition of the set $L(x^k)$ yields

$$a_i(x^k) \nabla f_i(x^k) \geq -d_i^k > 0. \tag{18}$$

Consequently,

$$\nabla f_i(x^k) d_i^k \leq -\frac{1}{a_i(x^k)} (d_i^k)^2.$$

Taking into account that the functions a_i and b_i are assumed to be bounded above on K, there is $\gamma_0 > 0$ such that

$$0 \leq a_i(x) \leq \gamma_0, \quad 0 \leq b_i(x) \leq \gamma_0 \qquad x \in K, i \in I.$$

Hence (17) is valid for $\gamma \in (0, \frac{1}{\gamma_0}]$ and $i \in L(x^k)$. The proof for $i \in U(x^k)$ is similar.

If $i \in L^k$, the definition of L^k yields $\nabla_i f(x^k) > \delta > 0$. Hence,

$$\frac{\nabla_i f(x^k)}{x_i^k - l_i} > \frac{\delta}{x_i^k - l_i} \geq \frac{\delta}{u_i - l_i}.$$

Setting $\gamma_1 = \delta/(u_i - l_i)$, we get $\nabla_i f(x^k) > -\gamma_1 d_i$. Multiplying this inequality by $d_i^k = l_i - x_i^k < 0$, relation (17) follows with $\gamma \in (0, \gamma_1]$ for all $i \in L^k$. As can easily be seen, the same holds for $i \in U^k$. Finally, the case when $i \in Z^k$ yields $d_i^k = \text{proj}(-\nabla_i f(x^k), [l_i - x_i^k, u_i - x_i^k])$. This leads to $\nabla_i f(x^k) d_i^k \leq -(d_i^k)^2$. Thus, (17) is satisfied for $\gamma \in (0, 1]$ and $i \in Z^k$. Therefore, setting $\gamma = \min\{\rho, \frac{1}{\gamma_0}, \gamma_1, 1\}$, relation (17) follows for all $i \in I \setminus F^k$. Moreover, keeping (16) and (17) in mind, we obtain

$$\begin{aligned} \nabla f(x^k)^T d^k &= \nabla_{F^k} f(x^k)^T d_{F^k}^k + \sum_{i \in I \setminus F^k} \nabla_i f(x^k) d_i^k \\ &\leq -\gamma \sum_{i=1}^{n} (d_i^k)^2 \\ &\leq -\gamma \| d^k \|^2. \end{aligned}$$

(3) Let $d^k = 0$ be satisfied. Then we prove that x^k must fulfil the first-order necessary optimality conditions (2). Consider an arbitrary index $i \in L(x^k)$. Then $0 = d_i^k = l_i - x_i^k$. This implies $x_i^k = l_i$. Hence,

$$0 = -d_i^k \leq a_i(x^k) \nabla_i f(x^k), \quad a_i(x^k) > 0$$

follows. Therefore, $\nabla_i f(x^k) \geq 0$ for $i \in L(x^k)$. By analogous arguments, we have $\nabla_i f(x^k) \leq 0$ for $i \in U(x^k)$. This means that the first or the third relation in (2) is valid for $i \in L(x^k) \cup U(x^k) = I \setminus F(x^k)$.

Now, we examine the case when $i \in F(x^k)$. If $\nabla_i f(x^k) = 0$, then $i \in F(x^k) = I \setminus (L(x^k) \cup U(x^k))$ implies $l_i < x_i^k < u_i$. Thus, the second relation in (2) holds. Therefore, we will assume that $\nabla_i f(x^k) \neq 0$. For $i \in L^k$, we have $0 = d_i^k = l_i - x_i^k$ and $\nabla_i f(x^k) > \delta > 0$, i.e., the first relation in (2) is satisfied. Analogously, for $i \in U^k$ we obtain that the third relation holds. If $i \in Z^k$, then $d_i^k = \text{proj}(-\nabla_i f(x^k), [l_i - x_i^k, u_i - x_i^k])$. Because $d_i^k = 0$ and $\nabla_i f(x^k) \neq 0$, it follows that either $d_i^k = u_i - x_i^k$ or $d_i^k = l_i - x_i^k$. In the first case, we have $x_i^k = l_i$ and $\nabla_i f(x^k) > 0$ from the definition of d_i^k. In the latter case, a similar reasoning provides $x_i^k = u_i$ and $\nabla_i f(x^k) < 0$. Therefore, if $i \in Z^k$, the first or the third relation in (2) is fulfilled.

Finally, let us consider $i \in F^k$. Suppose that $x_i^k = l_i$. Then $\nabla_i f(x^k) < 0$ because $i \notin L(x^k)$. Now, define $\bar{d}_{F^k}$ by

$$(\bar{d}_{F^k})_j = \begin{cases} 0 & \text{if } j \in F^k \setminus \{i\}, \\ \varepsilon & \text{if } j = i \end{cases}$$

for some small enough $\varepsilon > 0$ such that $\bar{d}_{F^k}$ is feasible with respect to the reduced quadratic program (11). Since the objective function of this program is strictly convex and $m_k(\bar{d}_{F^k}) = \nabla_i f(x^k)\varepsilon + 1/2\varepsilon^2 m_{ii}$, the scalar $\varepsilon > 0$ can be chosen as small as necessary such that $m_k(\bar{d}_{F^k}) < 0$. This is a contradiction because $d_{F^k}^k = 0$ is the optimal solution and $m_k(d_{F^k}^k) = 0$ is the corresponding optimal function value of the quadratic program (11). Consequently, $l_i < x_i^k$ must be valid for all $i \in F^k$. Similarly, $x_i^k < u_i$ can be shown for all $i \in F^k$. Therefore, having in mind that $0 = d_{F^k}^k$ satisfies the first-order necessary conditions of the quadratic program (11), $\nabla_{F^k} f(x^k) + M_{F^k F^k} d_{F^k}^k = \nabla_{F^k} f(x^k) = 0$ follows, i.e., the second relation in (2) is satisfied for all $i \in F^k$. Thus, $d^k = 0$ implies that x^k is a stationary point of problem (1).

Conversely, let x^k be a stationary point. Using (2), it can easily be seen that

$$\nabla f(x^k)^T d \geq 0$$

is valid for any feasible direction d with $x^k + d \in K$. Then $d^k = 0$ by statement (2) of this lemma. □

To prove the main result of this section, we need the following property.

Lemma 2. Let $\{x^k\}$ be any sequence generated by the algorithm BAS. If, for some subsequence $\{x^k\}_{k \in \mathcal{K}}$,

$$\lim_{k \in \mathcal{K}} x^k = \bar{x}, \quad \lim_{k \in \mathcal{K}} d^k = 0, \tag{19}$$

then $\bar{x}$ is a stationary point of the program (1).

Proof. Because the set I contains a finite number of elements, index sets $L, U, Z, F \subseteq I$ and some infinite set $\mathcal{M} \subseteq \mathcal{K}$ exist such that $L \cup U \cup Z \cup F = I$ and

$$L = L(x^k) \cup L^k, \quad U = U(x^k) \cup U^k, \quad Z = Z(x^k), \quad F = F^k$$

for all $k \in \mathcal{M}$. Taking into account the definition of the index sets which depend on k, it follows from (19) that

$$\bar{x}_L = l_L, \nabla_L\ f(\bar{x}) \geq 0, \quad \bar{x}_U = u_U, \nabla_U\ f(\bar{x}) \leq 0. \tag{20}$$

Consider $i \in Z$. Since $d_i^k = \text{proj}(-\nabla_i f(x^k), [l_i - x_i^k, u_i - x_i^k])$, one of the following three conditions is fulfilled:

$$\begin{aligned} &l_i < \bar{x}_i < u_i, && \nabla_i f(\bar{x}) = 0, \\ &\bar{x}_i = l_i, && \nabla_i f(\bar{x}) \geq 0, \\ &\bar{x}_i = u_i, && \nabla_i f(\bar{x}) \leq 0. \end{aligned} \tag{21}$$

For the remaining index set F, we recall that d_F^k is the solution of the reduced quadratic program (11). Having (19) in mind, (11) can be regarded (for $k \in \mathcal{M}$) as a perturbation of

$$\begin{gathered} \nabla_F f(\bar{x})^T d_F + \tfrac{1}{2}\, d_F^T M_{FF} d_F \ \to \min \\ \text{subject to } \ (l - \bar{x})_F \leq d_F \leq (u - \bar{x})_F. \end{gathered} \tag{22}$$

Since the feasible set of the latter program has an interior point, theorem 4.4 in [7] can be applied and, with regard to (19), we obtain that the zero vector is the unique solution of (22). By using arguments similar to those used for proving statement (3) of lemma 1, we get $\nabla_F f(\bar{x}) = 0$. Moreover, the constraints in (22) imply $l_F \leq \bar{x}_F \leq u_F$. These facts together with (20) and (21) show that $\bar{x}$ satisfies the relations (2), i.e., $\bar{x}$ is a stationary point of the program (1). □

We are now in position to establish the following result concerning the global convergence of the algorithm BAS.

Lemma 1. Let $\{x^k\}$ be any sequence generated by the algorithm BAS. Each accumulation point of $\{x^k\}$ is a stationary point of the program (1).

Proof. As $\{x^k\}$ remains in a compact set, it has at least one accumulation point. Let $\bar{x} = \lim_{k \in \mathcal{K}} x^k$ denote such a point, where $\mathcal{K} \subseteq \mathbb{N}$ is a suitable infinite set. We prove, by contradiction, that $\lim_{k \in \mathcal{K}} d^k = 0$. The desired result then follows from lemma 2. Suppose that there is an infinite set $\mathcal{N} \subseteq \mathcal{K}$ such that

$$\lim_{k \in \mathcal{N}} d^k = \bar{d} \neq 0. \tag{23}$$

By lemma 1 and the continuous differentiability of f, there is $\eta > 0$ such that

$$\lim_{k \in \mathcal{N}} \nabla f(x^k)^T d^k = \nabla f(\bar{x})^T \bar{d} = -\eta < 0. \tag{24}$$

This means that the test in step 2 of algorithm BAS can be satisfied only finitely often for all $k \in \mathcal{N}$. Otherwise, $\lim_{j \to \infty} \Delta_j = 0$. This would imply that $\{d^k\}_{k \in \mathcal{N}}$ contains a

subsequence converging to zero, which contradicts (23). Hence, for all $k \in \mathcal{N}$ sufficiently large, the stepsize θ_k is defined as the optimal solution of the one-dimensional quadratic problem (12). Because $\{f(x^k)\}$ is a decreasing and bounded sequence, it converges. Thus,

$$\lim_{k \in \mathcal{N}} \left(f(x^{k+1}) - f(x^k) \right) = 0,$$

or equivalently,

$$\lim_{k \in \mathcal{N}} \left(\theta_k \nabla f(x^k)^T d^k + \tfrac{1}{2} \theta_k^2 (d^k)^T M d^k \right) = 0. \tag{25}$$

The definition of θ_k in (13) and statement (2) of lemma 1 imply, for all sufficiently large $k \in \mathcal{N}$,

$$\theta_k (d^k)^T M d^k \leq -\nabla f(x^k)^T d^k.$$

Therefore, it follows that

$$\theta_k \nabla f(x^k)^T d^k + \theta_k^2 (d^k)^T M d^k \leq 0$$

and

$$\theta_k \nabla f(x^k)^T d^k + \tfrac{1}{2} \theta_k^2 (d^k)^T M d^k \leq -\tfrac{1}{2} \theta_k^2 (d^k)^T M d^k. \tag{26}$$

The definition of θ_k in (13) together with (23) and (24) yields, for all sufficiently large $k \in \mathcal{N}$,

$$\theta_k \geq \min \left(1, \frac{\eta}{2 \bar{d}^T M \bar{d}} \right) > 0.$$

This, (24) and (25) imply

$$\liminf_{k \in \mathcal{N}} \theta_k (d^k)^T M d^k > 0.$$

Hence, (26) contradicts (25). Therefore, (23) cannot hold and $\lim_{k \in \mathcal{K}} d^k = 0$ follows. □

4. Local analysis

In this section, we establish the finite termination of algorithm BAS if some iterate is close enough to a point where a certain sufficient optimality condition holds. Let the index set $\bar{B} \subseteq I$ and the vector $b \in \mathbb{R}^n$ be defined by

$$\bar{B} = \bar{L} \cup \bar{U},$$

$$b_i = l_i \quad i \in \bar{L}, \quad b_i = u_i \quad i \in \bar{U}, \quad b_i = 0 \quad i \in I \backslash \bar{B}.$$

Taking into account that the set F^k contains as much as possible indices such that $M_{F^k F^k}$ is positive definite, the following lemma can be derived from theorem 3 in Soares et al. [21].

Lemma. Let the vector $\bar{x}$ be a stationary point of (1) which satisfies the second-order condition

$$M_{\bar{F}\bar{F}} \text{ is positive definite} \tag{27}$$

and the strict complementarity condition

$$\nabla_i f(\bar{x}) \neq 0 \quad i \in \bar{B}. \tag{28}$$

Then there is $\varepsilon_1 > 0$ such that

$$\bar{F} = F^k, \ \bar{L} = L(x^k), \ \bar{U} = U(x^k)$$

for all $x^k \in \mathcal{B}(\bar{x}, \varepsilon_1) = \{x \in \mathbb{R}^n | \ \|\bar{x} - x\| \leq \varepsilon_1\}$.

To prove the finite termination of algorithm BAS, we first show that the unit stepsize is accepted in a sufficiently small neighbourhood of $\bar{x}$.

Lemma 4. Let the assumptions of lemma 3 be satisfied. Then there is $\varepsilon_2 \in (0, \varepsilon_1]$ such that $x^k \in \mathcal{B}(\bar{x}, \varepsilon_2)$ implies $\theta_k = 1$.

Proof. As in [21], we consider the following additional perturbed problem:

$$\begin{gathered} \nabla_{\bar{F}} f(\bar{x}^k)\bar{d}_{\bar{F}} + \tfrac{1}{2}\, \bar{d}_{\bar{F}}^T M_{\bar{F}\bar{F}} \bar{d}_{\bar{F}} \to \min \\ \text{subject to } \ (l - x^k)_{\bar{F}} \leq \bar{d}_{\bar{F}} \leq (u - x^k)_{\bar{F}}, \end{gathered} \tag{29}$$

where

$$\bar{x}^k = \begin{pmatrix} \bar{x}_{\bar{B}} \\ x^k_{\bar{F}} \end{pmatrix}.$$

By (27), the program (29) has a unique solution which is denoted by $\bar{d}^k_{\bar{F}}$. Incidentally, lemma 3 justifies to say that (29) is a perturbation of (11). Since $\nabla_{\bar{F}} f(\bar{x}) = 0$ and $(\bar{x} - \bar{x}^k)_{\bar{B}} = 0$, it follows that

$$\begin{aligned} 0 &= \nabla_{\bar{F}} f(\bar{x}^k + (\bar{x} - \bar{x}^k)) \\ &= \nabla_{\bar{F}} f(\bar{x}^k) + M_{\bar{F}I}(\bar{x} - \bar{x}^k) \\ &= \nabla_{\bar{F}} f(\bar{x}^k) + M_{\bar{F}\bar{F}}(\bar{x} - \bar{x}^k)_{\bar{F}}. \end{aligned} \tag{30}$$

The Karush–Kuhn–Tucker conditions for problem (29) read as follows:

$$\begin{aligned} &\nabla_{\bar{F}} f(\bar{x}^k) + M_{\bar{F}\bar{F}} \bar{d}_{\bar{F}} + \upsilon - w = 0, \\ &(l - x^k)_{\bar{F}} \leq \bar{d}_{\bar{F}} \leq (u - x^k)_{\bar{F}}, \ \ 0 \leq \upsilon, \ \ 0 \leq w, \\ &\upsilon^T(\bar{d}_{\bar{F}} - (u - x^k)_{\bar{F}}) = 0, \ \ w^T(\bar{d}_{\bar{F}} - (l - x^k)_{\bar{F}}) = 0. \end{aligned} \tag{31}$$

Since the objective function in (29) is strictly convex, these conditions are necessary and sufficient for $\bar{d}^k_{\bar{F}}$ to be the (unique) optimal solution. Therefore, with regard to (30) and

$$(l - x^k)_{\overline{F}} \leq (\bar{x} - x^k)_{\overline{F}} \leq (u - x^k)_{\overline{F}},$$

the solution $(\overline{d}^k_{\overline{F}}, w^k, v^k) \in \mathbb{R}^{\overline{n}} \times \mathbb{R}^{\overline{n}} \times \mathbb{R}^{\overline{n}}$ of the system (31) is uniquely determined by

$$\overline{d}^k_{\overline{F}} = (\bar{x} - \bar{x}^k)_{\overline{F}} = (\bar{x} - x^k)_{\overline{F}}, \quad w^k = v^k = 0 \tag{32}$$

with $\overline{n} = |\overline{F}|$. Now, let us consider the search direction d^k used in algorithm BAS. From a classical stability result concerning strictly convex quadratic programs [7], it follows that there is $c_1 > 0$ such that

$$\| d^k_{\overline{F}} - \overline{d}^k_{\overline{F}} \| \leq c_1 \| x^k - \bar{x}^k \| \quad x^k \in \mathcal{B}(\bar{x}, \tilde{\varepsilon}_2) \tag{33}$$

with $\tilde{\varepsilon}_2 \in (0, \varepsilon_1]$ sufficiently small. Using the Taylor formula, we get

$$f(x^k - d^k) = f(\bar{x}) + \nabla f(\bar{x})^T (x^k + d^k - \bar{x}) + \tfrac{1}{2} (x^k + d^k - \bar{x})^T M (x^k + d^k - x)$$

and

$$f(x^k) = f(\bar{x}) + \nabla f(\bar{x})^T (x^k - \bar{x}) + \tfrac{1}{2} (x^k - \bar{x})^T M (x^k - \bar{x}).$$

Subtracting both equations yields

$$f(x^k + d^k) - f(x^k) = \nabla f(\bar{x})^T d^k + \tfrac{1}{2} (d^k)^T M d^k + (d^k)^T M (x^k - \bar{x}). \tag{34}$$

Since, in view of lemma 3,

$$d^k_{\overline{B}} = (\bar{x} - x^k)_{\overline{B}}, \tag{35}$$

it follows with (32) that

$$\begin{aligned} M(x^k - \bar{x}) &= -Md^k + M(d^k + x^k - \bar{x}) \\ &= -Md^k + M_{I\overline{F}} (d^k + x^k - \bar{x})_{\overline{F}} \\ &= -Md^k + M_{I\overline{F}} (d^k_{\overline{F}} - \overline{d}^k_{\overline{F}}). \end{aligned}$$

Consequently, we get from (34)

$$f(x^k + d^k) - f(x^k) \leq \nabla f(\bar{x})^T d^k - \tfrac{1}{2} (d^k)^T M d^k + (d^k)^T M_{I\overline{F}} (d^k_{\overline{F}} - \overline{d}^k_{\overline{F}}). \tag{36}$$

Moreover, using (35), (33) and (32), we have

$$\begin{aligned} \| d^k \| &\leq \| d^k_{\overline{B}} \| + \| d^k_{\overline{F}} \| \\ &\leq \| (\bar{x} - x^k)_{\overline{B}} \| + \| d^k_{\overline{F}} - \overline{d}^k_{\overline{F}} \| + \| \overline{d}^k_{\overline{F}} \| \\ &\leq \| \bar{x} - x^k \| + c_1 \| x^k - \bar{x}^k \| + \| \bar{x} - x^k \| \\ &\leq (2 + c_1) \| \bar{x} - x^k \|. \end{aligned} \tag{37}$$

If we set $c_2 = c_1(2 + c_1)\|M\|$, then (36), (33) and the last inequality yield

$$f(x^k + d^k) - f(x^k) \leq \nabla f(\bar{x})^T d^k - \tfrac{1}{2}(d^k)^T M d^k + c_2 \|\bar{x} - x^k\| \|\bar{x}^k - x^k\| \quad (38)$$

for all $x^k \in \mathcal{B}(\bar{x}, \tilde{\varepsilon}_2)$. With regard to lemma 3 and to the definitions of the index sets $\bar{B}$, $\bar{F}$, $L(x^k)$, $U(x^k)$, we obtain

$$\begin{aligned}
\nabla f(\bar{x})^T d^k &= \sum_{i \in \bar{B} \cup \bar{F}} \nabla_i f(\bar{x}) d_i^k \\
&= \sum_{i \in \bar{B}} \nabla_i f(\bar{x})(b_i - x_i^k) \\
&\leq -\min_{i \in \bar{B}} \{|\nabla_i f(\bar{x})|\} \sum_{i \in \bar{B}} |b_i - x_i^k| \\
&\leq -c_3 \|(b - x^k)_{\bar{B}}\| \\
&\leq -c_3 \|\bar{x}^k - x^k\|
\end{aligned}$$

for some $c_3 > 0$. This and (38) imply

$$\begin{aligned}
f(x^k + d^k) - f(x^k) &\leq -\tfrac{1}{2}(d^k)^T M d^k + \|\bar{x}^k - x^k\| (c_2 \|\bar{x}^k - x^k\| - c_3) \\
&\leq -\tfrac{1}{2}(d^k)^T M d^k
\end{aligned} \quad (39)$$

for all $x^k \in \mathcal{B}(\bar{x}, \varepsilon_2)$ with $\varepsilon_2 \in (0, \tilde{\varepsilon}_2]$ sufficiently small. Thus, using the Taylor formula, we have

$$f(x^k + d^k) - f(x^k) = \nabla f(x^k)^T d^k + \tfrac{1}{2}(d^k)^T M d^k \leq -\tfrac{1}{2}(d^k)^T M d^k.$$

Therefore,

$$\nabla f(x^k)^T d^k + (d^k)^T M d^k \leq 0$$

for all $x^k \in \mathcal{B}(\bar{x}, \varepsilon_2)$. This means that $\theta_k = 1$ for all the iterates x^k in this neighbourhood of $\bar{x}$. □

We are now able to prove the main result of this section.

Theorem 2. Let the assumptions of lemma 3 be satisfied. Then there is $\varepsilon > 0$ such that $x^k \in \mathcal{B}(\bar{x}, \varepsilon)$ implies the termination of algorithm BAS with a local minimizer of the quadratic programming problem (1) after a finite number of steps.

Proof. Without loss of generality, we can assume that the algorithm does not stop with $d^k = 0$ or $d^{k+1} = 0$. Consider $\varepsilon \in (0, \varepsilon_2]$, where ε_2 is the positive value used in the previous lemma. Then it follows from lemma 4 that $x^k \in \mathcal{B}(\bar{x}, \varepsilon)$ yields $\theta_k = 1$. Moreover, using (37), we get

$$\| x^{k+1} - \bar{x} \| = \| x^k + d^k - \bar{x} \| \leq \| x^k - \bar{x} \| + \| d^k \| \leq \varepsilon + (2 + c_1)\varepsilon = (3 + c_1)\varepsilon.$$

Setting $\varepsilon := (3 + c_1)^{-1}\varepsilon_2$, we obtain $\varepsilon \leq \varepsilon_2$ and $x^{k+1} \in \mathcal{B}(\bar{x}, \varepsilon_2)$. The latter yields $\theta_{k+1} = 1$, owing to lemma 4. With regard to lemma 3, we have

$$x_{\bar{B}}^{k+1} = (x^k + d^k)_{\bar{B}} = b_{\bar{B}} = \bar{x}_{\bar{B}}, \quad \bar{x}^{k+1} = x^{k+1}.$$

Thus,

$$d^{k+1} = \bar{d}^{k+1}, \quad d_{\bar{B}}^{k+1} = 0, \quad x_{\bar{B}}^{k+2} = x_{\bar{B}}^{k+1} = \bar{x}_{\bar{B}}$$

follows. Using (32) for $k + 1$ instead of k, we get

$$x_{\bar{F}}^{k+2} = (x^{k+1} + d^{k+1})_{\bar{F}} = \bar{x}_{\bar{F}}.$$

Hence, $x^{k+2} = \bar{x}$ and, by lemma 1, algorithm BAS stops with $d^{k+2} = 0$. Finally, note that under conditions (28) and (29) the stationary point $\bar{x}$ is a local minimizer of problem (1). □

5. Concave programs

Consider the case when matrix M in program (1) is negative semi-definite, i.e. $x^T M x \leq 0$ for all $x \in \mathbb{R}^n$. Then all the diagonal entries m_{ii} of M with $i \in I$ must be negative or zero. Furthermore, if $m_{ii} = 0$ for some $i \in I$, then all elements in the ith row and the ith column of M must be zero, see, e.g., [19]. Therefore, the variable x_i can be set equal to one of the bounds l_i or u_i depending on the sign of q_i. If $q_i > 0$, then x_i has to be equal to l_i. Otherwise, we set $x_i = u_i$. Hence, we can assume without loss of generality that all the diagonal elements of M are negative. Then, according to the definition of the index set F^k, this set must be empty for each iteration. So the algorithm only uses extreme points of the constraint set K of the quadratic program (1), that is, each iterate x^k satisfies

$$x_i^k = l_i \text{ or } x_i^k = u_i \quad \text{for all } i = 1, \ldots, n.$$

Each iteration k of the algorithm can be stated in a very simple form. To do this, consider the active sets associated with the iterate x^k,

$$AL(x^k) = \{i \in I \mid x_i^k) l_i\}, \quad AU(x^k) = \{i \in I \mid x_i^k = u_i\},$$

and let

$$L^k = \{i \in I \mid x_i^k = l_i \text{ and } \nabla_i f(x^k) < 0\},$$

$$U^k = \{i \in I \mid x_i^k = u_i \text{ and } \nabla_i f(x^k) > 0\}.$$

Now two cases may occur:

(i) If $L^k \cup U^k = \emptyset$, then x^k is the desired stationary point.

(ii) Otherwise, set

$$AL(x^{k+1}) = AL(x^k) \cup U^k,$$

$$AU(x^{k+1}) = AU(x^k) \cup L^k,$$

and

$$x_i^{k+1} = \begin{cases} l_i & \text{if } i \in AL(x^{k+1}), \\ u_i & \text{if } i \in AU(x^{k+1}). \end{cases} \tag{40}$$

It is interesting to note that then the algorithm BAS becomes the block algorithm that was introduced in [11] for the solution of a general linear complementarity problem with box constraints. Therefore, if the quadratic function is concave, then no strictly convex subproblem has to be solved during the whole procedure, and each iteration simply amounts to moving some variables from one bound to another according to the rules presented above. It is also possible to exploit these simple ideas in the general case of an indefinite matrix M. To do this, let $G \subseteq I$ be the set defined by

$$G := \{i \in I \mid m_{ii} < 0\}.$$

Then again $G \cap F^k = \emptyset$ for each iteration k. The algorithm BAS realizes this fact when performing the Cholesky factorization of $M_{F(x^k)F(x^k)}$. However, the computational effort can be substantially reduced if these indices $i \in G$ do not come to $F(x^k)$, but instead the corresponding variables x_i^k stay in one of the bounds according to the value of the component i of the gradient $\nabla f(x^k)$. So, we propose the following simple modification in the definitions of the index sets $L(x^k)$ and $U(x^k)$:

$$\begin{aligned} L(x^k) = {} & \{i \in I \mid x_i^k \le l_i + a_i(x^k)\nabla_i f(x^k)\} \cup \\ & \{i \in G \mid x_i^k = l_i, \nabla_i f(x^k) \ge \delta\} \cup \\ & \{i \in G \mid x_i^k = u_i, \nabla_i f(x^k) > \delta\}, \\ U(x^k) = {} & \{i \in I \mid x_i^k \ge u_i + b_i(x^k)\nabla_i f(x^k)\} \cup \\ & \{i \in G \mid x_i^k = l_i, \nabla_i f(x^k) < -\delta\} \cup \\ & \{i \in G \mid x_i^k = u_i, \nabla_i f(x^k) \le \delta\}. \end{aligned} \tag{41}$$

It can easily be seen that this redefinition of the index sets $L(x^k)$ and $U(x^k)$ does not modify the search direction d^k. Therefore, algorithm BAS with this modification maintains the convergence properties proved in the two previous sections.

6. Computational experience

In this section we describe some experiments with algorithm BAS. This method was implemented in FORTRAN 77 using double precision arithmetic. All the

computations were done on a SUN SPARCstation 10 (48 MHz, 64Mb RAM). The parameters β and δ and the function $a_i(x)$ and $b_i(x)$, $i = 1,\ldots,n$, were set to $\beta = 10^{-7}$, $\delta = 10^{-7}$, $a_i(x) = 10^{-10}$, $b_i(x) = 10^{-10}$ for all $i = 1,\ldots,n$. Furthermore, for solving the strictly convex subproblem in each iteration of the BAS algorithm, we used the code of the block pivoting algorithm (see [12, 21], with parameter $p = 10$). The stopping criterion for the BAS algorithm employs the projected gradient $P(\nabla f(x^k))$, where

$$P_i(\nabla f(x^k)) = \begin{cases} \min(0, \nabla_i f(x^k)) & \text{if } x_i = l_i, \\ \nabla_i f(x^k) & \text{if } x_i \in (l_i, u_i), \\ \max(0, \nabla_i f(x^k)) & \text{if } x_i = u_i. \end{cases} \tag{42}$$

The algorithm terminates when $\|P(\nabla f(x^k))\| \leq 10^{-10}$. However, we also stopped the method if $\|d^k\| \leq 10^{-10}$. This second stopping criterion could be useful for highly degenerate problems and is justified by the property of the method that if $\|d^k\| = 0$, then x^k is a stationary point.

For the sake of comparison, we solved all the test problems by the algorithm ASN, that was implemented as described in [21]. We note that this latter procedure also includes the same code for the block pivoting method [21]. As discussed before, algorithm ASN was developed for general nonlinear programs with box constraints, but can also deal with quadratic programs without computing the Hessian of the objective function. Moreover, we tested the well-known code LANCELOT [17] and an efficient implementation [15] of the Fletcher–Jackson active-set method [13] ACTSET for each one of the test problems.

In the first experiments, we applied the four codes BAS, ASN, ACTSET and LANCELOT to all large-scale BQPs of the CUTE collection [6]. These problems are convex, that is, matrix M is positive semi-definite. The results are presented in table 1 and show that both algorithms ASN and BAS perform extremely well for these problems. With regard to the number of iterations, there is no clear winner between the two algorithms. However, algorithm BAS always needs less CPU time. This is essentially due to the fact that no perturbation technique is used in the BAS algorithm, whence the dimension of the strictly convex subproblem is always smaller for this algorithm. Another interesting fact that can be found in table 1 is the extremely low total number of inner iterations required by the block pivoting method for both algorithms. In many cases, the block algorithm required exactly one iteration for each of the subproblems. Actually, this type of behaviour also occurs for general nonlinear programs with box constraints [21]. It follows from the description of algorithm BAS that the number of function evaluations is equal to the number of the iterations plus one because each exact line search requires exactly one function evaluation. In contrast to this, the ASN method incorporates a nonmonotone Armijo-type criterion which, in general, increases the overall amount of function evaluations. This is another advantage of the BAS algorithm over the ASN method.

Table 1

CUTE quadratic programs.

	n	iterations				CPU time				inner iterations		function evaluations	
		BAS	ASN	LANC	ACTSET	BAS	ASN	LANC	ACTSET	BAS	ASN	BAS	ASN
JNLBRNGA	1024	7	7	7	664	0.97	2.49	1.63	3.48	7	7	8	15
JNLBRNGB	1024	3	3	3	570	0.37	1.49	2.23	2.46	3	3	4	11
NOBNDTOR	1024	9	9	9	1185	1.49	3.24	2.17	8.73	9	9	10	17
OBSTCLAE	1024	3	2	3	1020	1.2	2.33	15.54	8.73	11	15	4	6
OBSTCLAL	1024	7	7	7	1020	0.54	2.07	1.3	8.82	7	7	8	15
OBSTCLBL	1024	5	5	5	1083	0.98	3.27	2.55	9.17	16	15	6	10
OBSTCLBM	1024	1	3	3	1083	0.62	1.38	2.32	8.22	6	4	2	9
OBSTCLBU	1024	6	5	6	1083	0.62	1.76	1.33	8.34	10	6	7	10
TORSION1	1024	9	9	9	1332	0.79	2.54	1.51	9.68	9	9	10	17
TORSION2	1024	1	8	4	1332	0.77	2.51	2.1	9.62	10	8	2	16
TORSION3	1024	4	4	4	1644	0.15	1.39	0.87	6.62	4	4	5	12
TORSION4	1024	1	3	4	1664	0.56	1.49	0.94	6.68	7	13	2	8
TORSION5	1024	2	2	2	1676	0.05	1.01	0.61	4.48	2	2	3	10
TORSION6	1024	1	2	3	1676	0.51	1.21	0.8	4.51	5	5	2	6
JNLBRNG1	1024	10	4	5	664	1.39	1.91	1.3	3.49	12	4	11	8
JNLBRNG2	1024	7	3	3	602	0.85	1.56	1.01	2.77	7	3	8	7
TORSIONA	1024	9	9	9	1300	0.81	3.02	1.64	9.54	9	9	10	17
TORSIONB	1024	1	8	5	1300	0.75	2.65	2.53	9.58	7	8	2	16
TORSIONC	1024	4	4	4	1628	0.15	1.56	0.81	6.79	4	4	5	12
TORSIOND	1024	1	3	4	1628	0.37	1.63	0.95	6.96	6	13	2	8
TORSIONE	1024	2	2	2	1748	0.04	1.12	0.61	4.68	2	2	3	10
TORSIONF	1024	1	2	3	1748	0.5	3.49	0.75	4.46	5	5	2	6

As stated before, to gain a better impression of algorithm BAS (and ASN), we solved all the test problems with LANCELOT and an active-set method ACTSET. The results show that algorithm BAS always performs better than LANCELOT. However, it is important to note that LANCELOT has others variants [17] which can be more efficient for solving some of the test problems.

The number of iterations of the ACTSET method is quite large, since this method changes exactly one active constraint in each iteration. This is reflected in the CPU time and makes this code not too competitive in comparison with algorithm BAS. However, it is interesting to note that the efficiency of the implementation described in [16] as the difference in the number of iterations is in no way comparable with the gap in the CPU time (for instance, in problem TORSIONF, 2 iterations of ASN take 3.49 seconds, while 1748 iterations of ACTSET only need 4.46 seconds).

Since all the CUTE problems are convex and simple to solve, we decided to generate some nonconvex (M is indefinite) and concave problems. For the nonconvex

case, we chose some indefinite matrices from the HARWELL-BOEING collection [9]. The matrices M of the concave problems have been obtained by multiplying by (–1) all the entries of some positive semi-definite matrices of the same collection. We employed a quite simple technique for generating the vector q in such a way that at least a stationary point is known. Let M, the matrix of the quadratic program, be given. First we determine the largest positive definite submatrix M_{FF}. Then all the lower bounds l_i are set equal to zero. Furthermore, the upper bounds (u_i) are randomly generated real numbers uniformly distributed on (0, *Scal*], where S*cal* is a positive integer. Then a subset $\bar{F}$ of F and sets L and U are chosen such that

$$\bar{F} \cup L \cup U = \{1, \ldots, n\}$$

and

$$\bar{F} \cap L = \bar{F} \cap U = L \cap U = \emptyset.$$

Then we set

$$\begin{aligned} &\bar{z}_i = 0 && \text{if } i \in L, \\ &\bar{z}_i \in (l_i, u_i) && \text{if } i \in \bar{F}, \\ &\bar{z}_i = u_i && \text{if } i \in U. \end{aligned} \tag{43}$$

Finally, the vector q is constructed so that $\bar{z}$ is a stationary point, i.e., q has to satisfy the following conditions:

$$\begin{aligned} &(q + M\bar{z})_i \geq 0 && \text{if } i \in L, \\ &(q + M\bar{z})_i = 0 && \text{if } i \in \bar{F}, \\ &(q + M\bar{z})_i \leq 0 && \text{if } i \in U. \end{aligned} \tag{44}$$

Since the problems are not strictly convex, they can have more than one stationary point. Thus, we note that the algorithm could converge to a stationary point that is distinct from the one generated above.

We studied the effects caused by the size of the upper bounds as well as by the dimension of the active-set $(n - |\bar{F}|)$ at the generated stationary point. If we compare the results displayed in tables 2, 3 and 4, we immediately come to the conclusion that these two characteristics have no effect in algorithm BAS and a very small one in method ASN. On the other hand, the ACTSET method evidently depends on the number of active constraints of the stationary point obtained by the algorithm, but the size of the upper bound does not influence the efficiency of the method. In contrast to this, LANCELOT seems to perform worse when the size of the upper bounds is large. However, the number of active constraints at the stationary point obtained by the method does not seem to be an important factor for the efficiency of the LANCELOT code.

The results shown in tables 1–4 confirm that algorithm BAS is the most efficient procedure for solving all the test problems. The concave problems *bc*2103 and *bctm*15

Table 2

Test problems from generator with $|\overline{F}| = |F|/2$ and *Scal* = 1.

		iterations				CPU time				inner iterations		function evaluations	
	n	BAS	ASN	LANC	ACTSET	BAS	ASN	LANC	ACTSET	BAS	ASN	BAS	ASN
bc2101	3600	5	99	5	1552	3.02	112.2	27.25	14.86	8	124	6	483
bc2102	3600	8	145	8	1406	2.51	141.4	30	9.57	13	177	9	677
bc2103	3600	5	5	5	828	0.14	14.91	2.47	2.96	0	22	6	9
bcss13	2003	2	2	3	1297	2.43	8.99	67.23	3.95	7	7	3	6
bctm13	1473	3	3	4	771	0.12	6.25	81.24	1.26	10	19	4	7
bctm14	1473	3	4	4	370	0.07	6.68	10.49	0.6	5	22	4	8
bctm15	1473	3	3	3	149	0.06	5.11	0.99	0.22	0	17	4	7
oop210	3000	3	3	3	1523	0.63	5.73	1.84	9.3	3	3	4	7
pplat1	1919	3	3	3	1030	1.02	24.63	4.14	2.92	9	9	4	7
rr1031	1086	3	40	3	676	1.42	63.65	2.41	3.38	6	74	4	151

Table 3

Test problems from generator with $|\overline{F}| = |F|/2$ and *Scal* = 100.

		iterations				CPU time				inner iterations		function evaluations	
	n	BAS	ASN	LANC	ACTSET	BAS	ASN	LANC	ACTSET	BAS	ASN	BAS	ASN
bc2101	3600	5	100	10	1552	3.02	116.1	70.77	15.37	8	125	6	483
bc2102	3600	8	146	13	1406	2.5	143.4	17.28	9.6	13	178	9	676
bc2103	3600	5	5	10	828	0.15	14.67	4.05	2.95	0	22	6	8
bcss13	2003	2	2	9	1297	2.41	9.2	79.56	4.04	7	7	3	5
bctm13	1473	3	3	11	771	0.13	6.17	7.44	1.24	10	19	4	6
bctm14	1473	3	4	8	370	0.07	5.99	2.55	0.62	5	21	4	7
bctm15	1473	3	3	8	149	0.05	5.01	1.57	0.21	0	17	4	6
oop210	3000	3	3	7	1523	0.62	5.69	3.27	9.18	3	3	4	6
pplat1	1919	3	3	9	1030	1	11.69	5.83	2.99	9	9	4	6
rr1031	1086	3	40	7	676	1.42	62.49	11.73	3.39	6	74	4	150

have been solved without any inner iteration, since we have implemented the improvements presented in section 5. The gap between algorithms BAS and ASN is quite large for three problems, not only with respect to the CPU time, but also in terms of iterations. It is also interesting to notice that the behaviour of the active-set method for some test problems is better than algorithms ASN and LANCELOT. As a final conclusion of this study, we recommend algorithm BAS for computing a stationary

Table 4

Test problems from generator with $|\overline{F}| = |F|$ and $Scal = 1$.

	n	iterations				CPU time				inner iterations		function evaluations	
		BAS	ASN	LANC	ACTSET	BAS	ASN	LANC	ACTSET	BAS	ASN	BAS	ASN
bc2101	3600	5	112	5	1599	3.13	132.9	28.79	16.38	6	138	6	556
bc2102	3600	8	120	9	1380	2.57	125.3	30.63	9.9	12	176	9	536
bc2103	3600	5	5	5	828	0.14	14.84	2.6	2.96	0	22	6	9
bcss13	2003	2	2	2	1593	5.67	13.94	39.3	16.07	3	4	3	6
bctm13	1473	4	3	6	798	0.18	6.42	84.27	1.41	11	20	5	7
bctm14	1473	3	4	4	369	0.06	6.05	12.35	0.59	5	22	4	8
bctm15	1473	3	3	3	149	0.04	5.14	0.94	0.21	0	17	4	7
oop210	3000	3	3	3	1666	0.72	5.75	1.92	10.55	3	3	4	8
pplat1	1919	3	3	3	1134	2.75	14.3	5.31	8.98	8	8	4	7
rr1031	1086	3	27	3	681	1.27	43.52	2.24	3.51	5	50	4	96

point of large-scale nonconvex and concave quadratic programs with box constraints. We believe that this algorithm can be quite useful if it is incorporated in a code for finding global minima for this type of quadratic programs. This is certainly one of the main areas for our future research.

References

[1] D.P. Bertsekas, *Constrained Optimization and Lagrange Multipliers Methods*, Academic Press, New York, 1982.

[2] E.M. Bomze and G. Danninger, A finite algorithm for solving general quadratic problems, Journal of Global Optimization 4(1994)1–16.

[3] T. Coleman and L. Hulbert, A direct active set algorithm for large sparse quadratic programs with bounds, Mathematical Programming 45(1989)373–406.

[4] T. Coleman and Y. Li, A reflective Newton method for minimizing a quadratic function subject to bounds on some of the variables, Technical Report TR 92-1315, Department of Computer Science, Cornell University, USA, 1992.

[5] A. Conn, N. Gould and Ph. Toint, Global convergence of a class of trust regions algorithm for optimization with simple bounds, SIAM Journal on Numerical Analysis 25(1988)433–460.

[6] I. Bongartz, A. Conn, N. Gould and Ph. Toint, CUTE: Constrained and unconstrained testing environment, Technical Report 93/10, Department of Mathematics, Faculté Universitaires ND de la Paix, Namur, Belgium, 1993.

[7] J.W. Daniel, Stability of the solution of definite quadratic programs, Mathematical Programming 5(1973)41–53.

[8] R. Dembo and U. Tulowitzski, On the minimization of quadratic functions subject to box constraints, Technical Report, Department of Computer Science, Yale University, USA, 1983.

[9] I.S. Duff, R.G. Grimes and J.G. Lewis, Sparse matrix test problems, ACM Transactions on Mathematical Software 15(1989)1–14.

[10] A. Faustino and J. Júdice, Minimization of a concave quadratic function subject to box constraints, Investigacion Operativa 4(1994)49–68.
[11] A. Faustino and J. Júdice, Principal pivoting algorithms for a concave generalized linear complementarity problem, Investigação Operacional 14(1994)133–146.
[12] L. Fernandes, J. Júdice and J. Patricio, An investigation of the interior-point and block pivoting algorithms for large-scale symmetric monotone linear complementarity problems, Computational Optimization and Applications 5(1996)49–77.
[13] R. Fletcher and M. Jackson, Minimization of a quadratic function subject only to upper and lower bounds, Journal of Mathematics and Applications 14(1974)159–174.
[14] P. Hansen, B. Jaumard, M. Ruiz and J. Xiong, Global minimization of indefinite quadratic functions subject to box constraints, Naval Research Logistics 40(1993)373–392.
[15] J. Júdice and M. Pires, Direct methods for convex quadratic programs subject to box constraints, Investigação Operacional 9(1989)23–56.
[16] J. Júdice and F. Pires, A block principal pivoting algorithm for large-scale strictly monotone linear complementarity problems, Computers and Operations Research 21(1994)587–596.
[17] A. Conn, N. Gould and Ph. Toint, *LANCELOT – A Fortran Package for Large-Scale Nonlinear Optimization*, Springer, 1992.
[18] J. Moré and G. Toraldo, Algorithms for bound constrained quadratic programming problems, Numerische Mathematik 55(1989)377–400.
[19] K. Murty, *Linear Complementarity, Linear and Nonlinear Programming*, Heldermann, Berlin, 1988.
[20] P.M. Pardalos and G.P. Rodgers, A branch-and-bound algorithm for the maximum clique problem, Computers and Operations Research 19(1992)363–375.
[21] J. Soares, J. Júdice, and F. Facchinei, An active set Newton method for large-scale nonlinear programs with box constraints, to appear in SIAM Journal on Optimization.