Numerical methods for large-scale non-convex quadratic programming

by N.I.M. Gould[1] and Ph. L. Toint[2]

Report 01/01 7 February 2001

[1] Rutherford Appleton Laboratory,
Computational Science and Engineering Departement
Chilton, Oxfordshire, England
Email: gould@rl.ac.uk
Current reports available from
http://www.numerical.rl.ac.uk/reports/reports.html

[2] Department of Mathematics, University of Namur,
61, rue de Bruxelles, B-5000 Namur, Belgium,
Email: philippe.toint@fundp.ac.be
Current reports available by anonymous ftp from the directory
"pub/reports" on thales.math.fundp.ac.be

Numerical methods for large-scale non-convex quadratic programming

Nicholas I. M. Gould and Philippe L. Toint

07 February 2001

Abstract

We consider numerical methods for finding (weak) second-order critical points for large-scale non-convex quadratic programming problems. We describe two new methods. The first is of the active-set variety. Although convergent from any starting point, it is intended primarily for the case where a good estimate of the optimal active set can be predicted. The second is an interior-point trust-region type, and has proved capable of solving problems involving up to half a million unknowns and constraints. The solution of a key equality constrained subproblem, common to both methods, is described. The results of comparative tests on a large set of convex and non-convex quadratic programming examples are given.

1 Introduction

In this paper we consider two state-of-the-art algorithms for large-scale quadratic programming. Our aim is to illustrate the strengths and weaknesses of the two approaches, and to give the reader some indication of the sizes of problems that can now be (routinely) solved.

1.1 The generic problem

The quadratic programming (QP) problem is to

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad q(x) = g^T x + \tfrac{1}{2} x^T H x \quad \text{subject to} \quad Ax \geq b. \tag{1.1}$$

Here the Hessian matrix H is n by n symmetric, the m by n constraint Jacobian matrix A has rows a_i^T, $i = 1, \ldots, m$, $g \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and all data is real. In general, constraints may be bounded on both sides, i.e., $b^l \leq Ax \leq b^u$, they may include equalities $A_e x = b_e$, simple bounds $x^l \leq x \leq x^u$, and a variety of other interesting (and potentially exploitable) structure (such as those that arise from networks). For simplicity, we shall concentrate on problems of the generic forms (1.1) and

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad q(x) = g^T x + \tfrac{1}{2} x^T H x \quad \text{subject to} \quad Ax = b \quad \text{and} \quad x \geq 0, \tag{1.2}$$

and merely remark that modern quadratic programming codes should be (and often are) capable of coping with any of the above special structures without prompting from the user. We note that it is easy to convert a problem of the form (1.1) to that of (1.2), and vice versa, and that this is often done transparently by QP algorithms.

Our particular concern is for medium to large-scale problems, that is those involving tens or hundreds of thousands of unknowns and/or constraints. There is already a vast literature concerned with methods appropriate for small problems (those involving hundreds or low thousands of variables/constraints), and a number of excellent software packages (see, for instance, Fletcher, 1987*a*, 1987*b*, Gill, Murray and Wright, 1981, and the references given by Gould and Toint, 2000*a*). In addition, we shall make no assumption on the number of "degrees of freedom" (roughly, the number of variables minus the number of (active) constraints) encountered, although we recognize that there is again good (so-called null-space-based) software geared towards large problems for which the number of degrees of freedom stays small.

1.2 Applications

Quadratic programs arise naturally in many hundreds of applications, including portfolio analysis, support vector machines, structural analysis, VLSI design, discrete-time stabilization, optimal and fuzzy control, finite impulse response design, optimal power flow and economic dispatch. The online bibliography of Gould and Toint (2000*a*) contains over 1000 QP-related references, excluding those to the vital application of recursive/sequential/successive quadratic programming (SQP) methods, by which general constrained optimization problems are attacked by solving a sequence of suitable approximating quadratic programming models (see Boggs and Tolle, 1995, and Gould and Toint, 2000*b* for a description of SQP methods).

1.3 Convexity and non-convexity

Quadratic programming problems are normally classified as either being convex or non-convex purely on the basis of whether H is positive semi-definite or not. Convex QP problems are provably easy in the sense that there are (many) polynomial time solution algorithms (see, e.g. Vavasis, 1991), the best of these having polynomial iteration bounds of $O(\max(m,n)^{\frac{1}{2}})$ times the encoding length of the problem data, and an observed practical behaviour which appears essentially independent of m and n. Non-convex problems, on the other hand, are NP-hard (see, e.g., Vavasis, 1990). Furthermore, just finding a local minimizer of a non-convex QP is NP-hard—there are some methods that can guarantee to be within a certain fraction of local criticality in polynomial time (see, Ye, 1997)—as is even establishing that a given first-order critical (constrained stationary) point is a local minimizer (see, e.g., Murty and Kabadi, 1987, and Pardalos and Schnitger, 1988). This is particularly unfortunate since Newton-SQP models are based on the (naturally) indefinite Hessian of the Lagrangian, and thus give rise to non-convex QPs. Thus, regrettably, we shall content ourselves in this paper with methods which aim for what we shall call weak second-order critical points.

1.4 Optimality

Any point x^* that satisfies the conditions

$$\begin{array}{cl} Ax^* \geq b & (\textit{primal feasibility}) \\ Hx^* + g + A^T y^* = 0 \ \text{ and } \ y^* \leq 0 & (\textit{dual feasibility}) \\ (Ax^* - b)_i \cdot y_i^* = 0 \ \text{ for all } \ i & (\textit{complementary slackness}) \end{array} \tag{1.3}$$

for some vector of *Lagrange multipliers* y^* is a *first-order critical* (or Karush-Kuhn-Tucker) point for the QP (1.1). The *active set* at x^* is

$$\mathcal{A}(x^*) = \{ i \mid a_i^T x^* = b_i \},$$

and the critical point is *strictly complementary* if and only if $y_i^* < 0$ for all $i \in \mathcal{A}(x^*)$. Let

$$\mathcal{C}(x^*) = \left\{ s \;\middle|\; \begin{array}{l} a_i^T s = 0 \ \text{ for all } \ i \in \mathcal{A}(x^*) \ \text{ such that } \ y_i^* < 0 \ \text{ and } \\ a_i^T s \geq 0 \ \text{ for all } \ i \in \mathcal{A}(x^*) \ \text{ such that } \ y_i^* = 0 \end{array} \right\}. \tag{1.4}$$

Any first-order critical point x^* for which additionally

$$s^T H s \geq 0 \ \text{ (resp. } > 0) \ \text{ for all } \ s \in \mathcal{C}(x^*)$$

is a *second-order* (resp. *strong second-order*) critical point. The importance of the cone (1.4) becomes apparent in the following theorem.

Theorem 1.1 (Contesse, 1980, Mangasarian, 1980, Borwein, 1982). x^* is a (an isolated) local minimizer of the QP (1.1) if and only if x^* is (strong) second-order critical.

Notice that quadratic programming is highly unusual, as its necessary and sufficient optimality conditions coincide. Since, as we have said, checking a first-order critical point for (local or global) optimality is NP-hard unless the problem is convex, we cannot hope to ensure second-order criticality, so instead, we weaken our aim. Let

$$\mathcal{M}(x^*) = \{s \mid a_i^T s = 0 \text{ for all } i \in \mathcal{A}(x^*)\}$$

Any first-order critical point x^* for which additionally

$$s^T H s \geq 0 \text{ for all } s \in \mathcal{M}(x^*)$$

is a *weak* second-order critical point. Note that a weak second-order critical point may be a maximizer. However, checking for weak second-order criticality is easy, since now all we are requiring is that H be positive semi-definite over a manifold (the null-space of the active constraints), rather than the cone (1.4) required by Theorem 1.1. If, by chance, x^* turns out to be strictly complementary, the cone and manifold coincide, and x^* is a local minimizer.

1.5 Algorithms

The rest of the paper is concerned with algorithms for the solution of (1.1) or (1.2) (as appropriate). Since the fundamental subproblem we shall encounter in both of our main approaches is one in which there are only equality constraints $Ax = 0$, for which right-hand-sides are zero, we shall consider this in Section 2. Our two main algorithmic contenders for the general problem are active (or working)-set and interior-point (or barrier) methods. We shall develop an active-set method, applied to the problem (1.1), in Section 3, and follow this in Section 4 with a description of an interior point method, this time applied to the problem (1.2). The two approaches will be compared numerically in Section 5, and we conclude in Section 6.

2 Equality constrained problems

In this section, we concentrate on the equality constrained quadratic programming (EQP) problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; q(x) \stackrel{\text{def}}{=} g^T x + \tfrac{1}{2} x^T H x \;\text{ subject to }\; Ax = 0. \tag{2.1}$$

We shall assume that the m by n matrix A is of full-rank, and must be prepared to preprocess the problem to ensure that this is so—in general, this is only an issue for the first of the sequence of problems of the form (2.1) that our QP algorithms will solve, since subsequent subproblems inherit the property from their predecessors. Although the determination of rank is a tricky numerical problem, simple techniques like a sparse LU factorization with suitable pivoting usually succeed.

2.1 Optimality and consequences

The first-order criticality conditions for the EQP problem (2.1) are that there are Lagrange multipliers y for which

$$\begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -g \\ 0 \end{pmatrix}. \tag{2.2}$$

Second-order necessary optimality requires that $s^T Hs \geq 0$ for all s for which $As = 0$, while we shall say that H is *second-order sufficient* if $s^T Hs > 0$ for all $s \neq 0$ for which $As = 0$. There are four resulting possibilities.

Firstly, if (2.2) holds and H is second-order sufficient, it follows that x is the unique solution to (2.1). Secondly, if (2.2) holds, H is second-order necessary, but there is a vector s for which $Hs = 0$ and $As = 0$, then there is a family of *weak* minimizers $x + \alpha s$ for any $\alpha \in \mathbb{R}$. Thirdly, if there is an s for which $As = 0$, $Hs = 0$ and $g^T s < 0$ simultaneously hold, it follows that $q(\cdot)$ is unbounded from below along the *direction of linear infinite descent* s. Finally, if there is an s for which $As = 0$, $s^T Hs < 0$ and $s^T g \leq 0$, it follows that $q(\cdot)$ is unbounded from below along *direction of negative curvature* s. Our aim is thus to determine a solution to (2.1) if at all possible, and failing that to find a *direction of infinite descent* (i.e., negative curvature or linear infinite descent) along which $q(\cdot)$ is unbounded from below (for a discussion of these issues, see Conn and Gould, 1984).

2.2 An algorithm

We shall not go into a taxonomy (see Fletcher, 1987*a*, or Gill et al., 1981). of different ways to solve (2.1) here, since many of them are inappropriate when there are large number of variables We can rule out range- and null-space methods that rely on matrix factorizations as general purpose techniques, and must be suspicious of methods that aim to factorize the coefficient matrix (the "KKT" matrix)

$$\begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} \tag{2.3}$$

of (2.2) since the factors may fill-in significantly. The alternative to factorization-based approaches are iterative methods, and it is from these that we have selected our candidate. Probably the most obvious approach is to use the constraints $Ax = 0$ to eliminate variables, and to apply the conjugate-gradient (CG) method to the resulting reduced problem.

Formally this is a null-space method, that is we construct a basis N for the null-space of A (i.e., N is full rank and $AN = 0$), use the constraints to write $x = Nx_N$, and derive the unconstrained problem

$$\underset{x_N \in \mathbb{R}^{n-m}}{\text{minimize}} \quad x_N(N^T g) + \tfrac{1}{2} x_N^T N^T H N x_N. \tag{2.4}$$

The CG method is ideal for (2.4), since it is capable of finding a stationary point when that is required, and a direction of infinite descent when that is available—care has to be taken in the latter case, since the CG iteration can stall if the associated Krylov space is degenerate, but can be restarted if necessary (see, for example, Gould, Lucidi, Roma and Toint, 1999).

The disadvantage of this approach, aside from needing a null-space basis, is that it is less than obvious how to precondition the iteration (but, see, Coleman and Verma, 1998); in practice it is rare to use CG without some form of preconditioning. The approach we prefer was originally proposed by Polyak (1969), enhanced by Coleman (1994) and extended by Gould, Hribar and Nocedal (1998). The difference between this approach and that suggested by (2.4) is that the projection of the iterates into the null-space is performed *implicitly*. Moreover preconditioning is performed as part of the projection in, what we believe is, a natural way. The preconditioned projected conjugate gradient (PPCG) method is given as Algorithm 2.1.

Notice that r is nothing other than the gradient $g + Hx$ of q at x. Observant readers will recognise this as the usual (preconditioned) conjugate-gradient iteration for unconstrained quadratic minimization, with a so-far undefined preconditioning step (2.8).

Given an input vector g, the all-important preconditioning/projection operation $v = P[-r]$ requires that the output v satisfies the linear system

$$\begin{pmatrix} M & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} v \\ y \end{pmatrix} = \begin{pmatrix} -r \\ 0 \end{pmatrix}, \tag{2.12}$$

Algorithm 2.1: Preconditioned Projected Conjugate Gradients

Choose an initial point x satisfying $Ax = 0$, compute $r = Hx + g$, $v = P[-r]$ and $p = -v$. Repeat the following steps, until a convergence test is satisfied:

$$\alpha = -r^T v / p^T H p \tag{2.5}$$
$$x \leftarrow x + \alpha p \tag{2.6}$$
$$r^+ = r + \alpha H p \tag{2.7}$$
$$v^+ = P[-r^+] \tag{2.8}$$
$$\beta = (r^+)^T v^+ / r^T v \tag{2.9}$$
$$p \leftarrow v^+ + \beta p. \tag{2.10}$$
$$v \leftarrow v^+ \quad \text{and} \quad r \leftarrow r^+ \tag{2.11}$$

for some "appropriate" symmetric matrix M and some auxiliary vector y. Formally, the only restriction on M is that it should be second-order sufficient, although early proposers of Algorithm 2.1 made the simplifying assumption that M be positive definite (Coleman, 1994) or even the identity matrix (Polyak, 1969).

Notice the similarity in form between (2.2) and (2.12). If H is itself second-order sufficient, the choice $M = H$ is permissible, in which case (2.2) and (2.12) are (essentially) identical, and the PPCG method will terminate in a single iteration. There are, however, sometimes good reasons to avoid such an obvious choice. Firstly, there is no *a priori* reason why H should be second-order sufficient in general. Secondly, some form of (explicit or implicit) factorization of the coefficient matrix

$$K = \begin{pmatrix} M & A^T \\ A & 0 \end{pmatrix} \tag{2.13}$$

of (2.12) is required to obtain $P[-r]$—it is customary to use a sparse variant (see Duff, Reid, Munksgaard and Neilsen, 1979, Duff and Reid, 1983, Duff, Gould, Reid, Scott and Turner, 1991, or Vanderbei and Carpenter, 1993) of one of the symmetric-indefinite factorizations originally suggested by Bunch and Parlett (1971), Bunch and Kaufman (1977) and Fletcher (1976), for which there is good available software (for example, the HSL, 2000, codes `MA27` and `MA57`). The matrix (2.3) and its factors may have significantly more nonzeros than (2.13), and thus the cost of the operation (2.8) may be considerably reduced with an appropriate choice of M—of course, as always with preconditioning, this cost has to be balanced against an increase in the number of PPCG iterations resulting from a poor choice of M. As an extreme example, if M is nonsingular, a block elimination of (2.12) reveals that

$$Mv = -(r + A^T y), \quad \text{where} \quad AM^{-1}A^T y = -AM^{-1} r,$$

and it follows from the second-order sufficiency of M that $AM^{-1}A^T$ is non-singular and has precisely as many negative eigenvalues as has M (see Chabrillac and Crouzeix, 1984, and Gould, 1985). If, additionally, M is diagonal, and thus trivial to invert, and positive definite, $AM^{-1}A^T$ is also positive definite and thus amenable to (sparse) Cholesky factorization.

The precise form of M is an open research topic, although recent work has investigated the influence of different M on the convergence of the PPCG method (see Keller, Gould and Wathen, 2000), as well as proposing a number of possible choices for M (see Keller, 2000). Some caution must be exercised when computing the preconditioning step (2.12), since the entire PPCG method depends crucially on the relationship $Av = 0$. In floating point arithmetic, rounding can sometimes significantly violate this requirement especially when v is small (relative to y), but fortunately judicious use of iterative refinement or similar methods can control this potential drawback (see Gould et al., 1998).

It is usual to stop when $|r^T v| \equiv |v^T M v|$ is (relatively and/or absolutely) small, or when a fixed limit on the number of iterations has been exceeded. If $v^T M v = 0$, it follows that $v = 0$ since $Av = 0$ and M defines a norm on this manifold. In this case, (2.12) implies that $r + A^T y = 0 = g + Hx + A^T y$ and $Ax = 0$, and thus that the auxiliary variables y are actually Lagrange multipliers, c.f. (2.2). While this is not the case if $v^T M v \neq 0$, we may still regard y as usable Lagrange multiplier estimates so long as $v^T M v$ is small.

2.3 Negative curvature

Of course, as Section 2.1 suggests, Algorithm 2.1 will only solve (2.1) when H is second-order sufficient. In other cases—principally when H is indefinite on the manifold $Ax = 0$—the best we can hope from the algorithm as stated is that it finds a constrained stationary point. Fortunately, we can do better. The key, as before, is to recognise that the PPCG method *is* (a projected) conjugate gradient method in the null-space of A. If the denominator $p^T H p$ in (2.5) is less than (or equal to) zero, the problem is unbounded from below (or at best has a weak solution) since, by construction, the same p satisfies $Ap = 0$. Once negative curvature has been detected, we can do one of two things: stop, or try to find an even "better" direction of negative curvature, possibly even one which gives the "most-negative" curvature, which corresponds to the smallest eigenvalue of H restricted to the null-space of A. To obtain "good" negative curvature, we simply recognise that the (preconditioned) conjugate-gradient and Lanczos methods are actually one-and-the-same (see, for instance, Golub and Van Loan, 1989, and, particularly, Conn, Gould and Toint, 2000*a*, for further details). When the problem is unbounded from below, we shall require that the "solution" obtained is just such a direction of infinite descent.

2.4 Trust-region constraints

A related, and (for us) vitally important, variant of problem (2.1) is the constrained *trust-region* subproblem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; q(x) \stackrel{\text{def}}{=} g^T x + \tfrac{1}{2} x^T H x \;\; \text{subject to} \;\; Ax = 0 \;\; \textit{and} \;\; \|x\|_M \leq \Delta, \tag{2.14}$$

where $\|x\|_M^2 = x^T M x$, for some given "radius" $\Delta > 0$ and second-order sufficient symmetric matrix M. Notice that although $\|\cdot\|_M$ is not a norm—it is a semi-norm—it is a norm on the manifold $Ax = 0$. If we explicitly transform into the null-space of A via $x = N x_N$, the trust-region subproblem (2.14) becomes

$$\underset{x_N \in \mathbb{R}^{n-m}}{\text{minimize}} \;\; x_N (N^T g) + \tfrac{1}{2} x_N^T N^T H N x_N \;\; \text{subject to} \;\; \|x_N\|_{N^T M N} \leq \Delta. \tag{2.15}$$

We can (approximately) solve (2.15) using either the Steihaug (1983)–Toint (1981) truncated conjugate-gradient method (this stops on the boundary of the trust-region $\|x_N\|_{N^T M N} \leq \Delta$ if it gets that far) or the generalized Lanczos trust-region (GLTR) method of Gould et al. (1999) (which continues around the trust-region boundary if necessary). It will come as no surprise that precisely the same methods can be performed implicitly, exactly as in Algorithm 2.1: the PPCG iteration is performed so long as the iterates lie within the trust-region $\|x\|_M \leq \Delta$, but once an iterate encounters the trust-region boundary, the method either halts (Steihaug-Toint) or moves around the boundary while continuing to reduce $q(x)$ (GLTR). Notice here that it is the same matrix M which appears both in the definition of the trust region, and in form of the preconditioner. Since (at least in finite-dimension), it matters little from a theoretical viewpoint precisely which norm defines the trust-region, it is usual to first choose the preconditioner, and then assign the shape of the trust-region on this basis. The GLTR algorithm (which includes that of Steihaug-Toint as special case) is available as the package `HSL_VF05` in the HSL (2000).

3 An active-set method

Active (or as they are perhaps more correctly called, working) set methods aim to find a critical point of the general problem (1.1) by solving a (potentially long) sequence of related EQPs

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; q(x) \stackrel{\text{def}}{=} g^T x + \tfrac{1}{2} x^T H x \;\; \text{subject to} \;\; A_{\mathcal{W}} x = b_{\mathcal{W}}, \tag{3.1}$$

where the working set $\mathcal{W}$ is a prediction of the indices of those constraints which will be active at the desired critical point—the set subscript $_{\mathcal{W}}$ here indicates the submatrix/vector whose rows/components are indexed by the set $\mathcal{W}$. The defining features of an active set method are: (i) the *working* set is a subset of the indices of constraints which are active at the current estimate x^C for which $A_{\mathcal{W}}$ is of full rank, (ii) the subproblem (3.1) is actually solved to obtain a *search direction* s^C from x^C, and (iii) the next estimate x^+ is obtained as $x^C + \alpha s^C$ where α is chosen to reduce $q(x)$ or the current infeasibilty (or both).

Properties (i) and (ii) together imply that the search direction subproblem may be rewritten as

$$\underset{s \in \mathbb{R}^n}{\text{minimize}} \;\; f^C + g^{C^T} s + \tfrac{1}{2} s^T H s \;\; \text{subject to} \;\; A_{\mathcal{W}} s = 0 \tag{3.2}$$

where $g^C = g + Hx^C$ is the gradient of q at x^C, and the constant term $f^C = g^T x^C + \frac{1}{2} x^{C^T} H x^C$ may be ignored. Thus (3.2) is of the form (2.1) (for the given data), and the methods we discussed in Section 2 may be applied. Recall that by "solving" (3.2), we mean finding a direction of infinite descent (if there is one) or a weak second-order critical point.

3.1 Generic active-set methods

For simplicity, we suppose that x^C is feasible, and that we wish successive iterates to inherit this property. In practice, feasibility can be achieved by solving a "phase-1" problem, or (as actually happens in our implementation) feasibility may be relaxed and a composite penalty problem (trading infeasibility against objective decrease) solved instead (see Section 3.3).

If $s^C = 0$ is the "solution" of (3.2), the associated vector y^C of Lagrange multipliers (c.f., (2.2)) may be used to determine whether q may be further reduced by removing an element from the current working set (and thereby allowing a step to be taken into the interior of the feasible region for the corresponding constraint). In particular, if $y_i^C > 0$, the objective function may be further reduced by removing the ith element of $\mathcal{W}$, while if $y^C \leq 0$ (and all the constraints feasible), x^C is a weak second-order critical point (c.f. (1.3)), since the fact that $s^C = 0$ is the "solution" rules out the possibility of infinite-descent directions from x^C.

If $s^C \neq 0$, the objective function may be decreased as α increases from 0, and progress will only be interupted when an initially feasible constraint, say the jth, becomes active (and thus infeasible for any larger α). Since, for this constraint $a_j^T s^C \neq 0$ while $A_{\mathcal{W}} s^C = 0$, a_j^T must be linearly independent of the constraints indexed by the current working set. Thus it is safe (at least in theory—caution is needed in practice to guard against "nearly" dependent working constraints) to set $\mathcal{W}^+ = \mathcal{W} \bigcup \{j\}$. If s^C points to a weak second-order critical point of (3.2), and α reaches 1 without interference, $x^C + s^C$ is optimal for the current working set; we have reached the position described in the previous paragraph, and thus will need to remove an index from the working set at the next iteration if we are to make further progress. If s^C is a direction of infinite descent for (3.2), and no constraint interferes as α increases, the original problem is unbounded from below, and we terminate with this information.

In summary, a working set method is simply a mechanism by which a sequence of EQPs are solved depending on a given working set; the working set at one iteration differs from its predecessor by the introduction or removal of a single element. Progress is always possible if nonzero steps α are taken. If this is the case, termination is finite (but not necessarily rapid or polynomially bounded), since there are only a finite number of active sets, and each is investigated at most once. The possibility of zero steps arises when constraints are *degenerate*, that is active but not in the working set. Many so-called anti-degeneracy rules have been proposed to deal with this worrying possibility, and by far the easiest in our experience is to randomly perturb the right-hand-sides of the constraints, and only restore (and refine) the solution when optimal for the perturbed version—a randomly perturbed problem can "never" be degenerate.

Active set methods for both convex and non-convex QP are considered in more detail by Fletcher (1987*a*, 1987*b*), Gill et al. (1981) and Nocedal and Wright (1999), and the papers contained therein. The vast majority of existing methods use explicit or implicit (updated) factors of (2.3) to find a stationary point of EQP. As we have mentioned, this may not be appropriate for large problems, nor is it immediately obvious how to find directions of infinite descent when they exist. An important class of *inertia controlling* methods avoid the second of these potential defects by careful choice of the initial working set (important examples are those given by Fletcher, 1971 and Gill and Murray, 1978; see the review by Gill, Murray, Saunders and Wright, 1991), but this can be inconvenient when a user wishes to prescribe this set rather than letting the algorithm generate it. We believe that the method outlined in Section 2 avoids both of the above disadvantages.

3.2 Solving sequences of closely-related EQPs

The main novelty in our approach is in how to solve the sequence of closely related EQP subproblems that arise in the above generic active-set method when using the method we outlined in Sections 2.2–2.3. The crucial aspect is that the only significant difference between successive subproblems is that the *preconditioning step* (2.12) depends upon different but closely-related $A_{\mathcal{W}}$ for each subproblem.

3.2.1 The Schur complement update method

The overall algorithm we propose is divided into a sequence of major iterations. At the start of each major iteration, a factorization of the preconditioning matrix

$$K_k = \begin{pmatrix} M_k & A_k^T \\ A_k & 0 \end{pmatrix},$$

involving the set of constraints in the current working set, is found—we shall call the set of these constraints at the start of a major iteration the *reference* set. As we have already said, the symmetric matrix M_k is chosen so that it is second-order sufficient, but is otherwise arbitrary. We stress that although it is desirable to choose a good approximation of H, the overriding concern is that M_k be second-order sufficient. If M_k is second-order sufficient, we shall say that the augmented matrix K_k is *standard*. Otherwise, it is *nonstandard*.

Having determined the factors of K_k, all subsequent linear systems during the current major iteration are solved using the Schur complement method. That is to say, if we require the solution of a system

$$\begin{pmatrix} M_\ell & A_\ell^T \\ A_\ell & 0 \end{pmatrix} \begin{pmatrix} s_\ell \\ t_\ell \end{pmatrix} = - \begin{pmatrix} g_\ell \\ c_\ell \end{pmatrix}, \tag{3.3}$$

for $\ell \geq k$, and if

$$M_k = M_\ell,$$

the solution is obtained using the factors of K_k and an appropriate Schur complement involving M_k, A_k and A_ℓ—notice here that thus far we do not allow M_ℓ to change during the course of a major iteration. To be specific, suppose without loss of generality, that

$$A_k = \begin{pmatrix} A_C \\ A_D \end{pmatrix}, \;\; A_\ell = \begin{pmatrix} A_C \\ A_A \end{pmatrix} \;\text{ and }\; c_\ell = \begin{pmatrix} c_C \\ c_A \end{pmatrix}, \tag{3.4}$$

that is that the rows A_C are common to A_k and A_ℓ, but that the rows A_D in A_k are replaced by the rows

A_A in A_ℓ. In this case, the solution to (3.3) also satisfies the expanded system

$$\left(\begin{array}{ccc|cc} M_k & A_C^T & A_D^T & A_A^T & 0 \\ A_C & 0 & 0 & 0 & 0 \\ A_D & 0 & 0 & 0 & I \\ \hline A_A & 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \end{array}\right) \begin{pmatrix} s_\ell \\ t_C \\ t_D \\ t_A \\ u_\ell \end{pmatrix} = - \begin{pmatrix} g_\ell \\ c_C \\ 0 \\ c_A \\ 0 \end{pmatrix}, \tag{3.5}$$

where we recover

$$t_\ell = \begin{pmatrix} t_C \\ t_A \end{pmatrix}.$$

Notice that the leading 3 by 3 block of the coefficient matrix of (3.5) is simply K_k, and thus that the system may be written as

$$\begin{pmatrix} K_k & B_\ell^T \\ B_\ell & 0 \end{pmatrix} \begin{pmatrix} v_\ell \\ w_\ell \end{pmatrix} = - \begin{pmatrix} h_\ell \\ d_\ell \end{pmatrix}, \tag{3.6}$$

for the appropriately repartitioned data

$$B_\ell = \begin{pmatrix} A_A & 0 & 0 \\ 0 & 0 & I \end{pmatrix}, \quad v_\ell = \begin{pmatrix} s_\ell \\ t_C \\ t_D \end{pmatrix} \quad \text{and} \quad w_\ell = \begin{pmatrix} t_A \\ u_\ell \end{pmatrix},$$

and right-hand side

$$h_\ell = \begin{pmatrix} g_\ell \\ c_C \\ 0 \end{pmatrix} \quad \text{and} \quad d_\ell = \begin{pmatrix} c_A \\ 0 \end{pmatrix}.$$

Thus (3.6) can be solved in the standard way using the factors of K_k and those of the Schur complement $S_l = -B_l K_k^{-1} B_\ell^T$. Crucially, the factors of S_ℓ may be *updated* rather than recomputed every time a constraint is added to or removed from the working set. It is usual to store the growing matrix S_ℓ and its factors as dense matrices; as a consequence each major iteration is concluded when the dimension of S_ℓ exceeds a given upper limit (or when the cost of continuing to enlarge the Schur complement method is believed to exceed that of refactorizing K_ℓ).

This method was first suggested by Bisschop and Meeraus (1977), and championed by Gill, Murray, Saunders and Wright (1990, 1991). We have implemented such a method as part of the package `MA39` in the HSL (2000)—the package is actually designed to handle updates in the unsymmetric case, but is capable of exploiting both symmetry and even *a priori* knowledge that S_ℓ is definite. In principle, a symmetric indefinite factorization of S_ℓ is both possible, and possible to update. However, the details are complicated (see Sorensen, 1977), and we have chosen instead to use a nonsymmetric (QR) factorization since updates are then relatively straightforward.

3.2.2 Inertia control

It is important to be able to check that K_ℓ is standard at every iteration. Fortunately this is easy. Specifically, a very minor modification of Gill et al. (1991, Lemma 7.2) gives the following:

Theorem 3.1 Suppose that the most recent reference iteration is k, and that both K_k and K_ℓ ($\ell \geq k$) are standard. Then

$$\text{In}(S_\ell) = (\sigma_-, \sigma_+, 0), \tag{3.7}$$

where the inertia $\text{In}(S_\ell)$ gives the number of positive, negative and zero eigenvalues of S_ℓ respectively, σ_+ constraints have been added since the start of the major iteration, and σ_- have been deleted.

Since we require that K_k is standard, it follows that if, at any stage, the inertia of S_ℓ does not agree with (3.7), it must be because K_ℓ is nonstandard. It is easy to check this condition since the inertia of S_ℓ may be recurred as its factors are updated (in our case, since we are using the non-symmetric QR factors, we record the determinants S_ℓ on subsequent iterations—a change in sign when a single row and column are added or deleted indicates an extra negative eigenvalue, while a repeated sign indicates an extra positive one—directly from the products of those of Q and R. We ensure by construction that $\det Q = 1$, while the eigenvalues of R are merely its diagonal entries.). We now consider the implication of adding and deleting constraints for the inertia of S_ℓ.

3.2.3 Adding a constraint

If K_ℓ is standard, and we add a constraint to the working set, $K_{\ell+1}$ is also standard provided that $M_{\ell+1} = M_\ell$. This follows immediately, since as we have already said K_ℓ being standard is equivalent to $N_\ell^T M_\ell N_\ell$ being positive definite, where the columns of N_ℓ form am orthonormal basis for the null-space of the full-rank matrix A_ℓ, the fact that

$$N_\ell = \begin{pmatrix} N_{\ell+1} \\ n^T \end{pmatrix} V$$

for some vector n and orthonormal matrix V (see Gill, Golub, Murray and Saunders, 1974), and the observation that $N_{\ell+1}^T M_{\ell+1} N_{\ell+1}$ is then a principal submatrix of the positive definite matrix $V N_\ell^T M_{\ell+1} N_\ell V^T = V N_\ell^T M_\ell N_\ell V^T$, and hence is itself positive definite.

3.2.4 Deleting a constraint

Complications arise when we delete a constraint, since then it does not automatically follow that $K_{\ell+1}$ is standard even if K_ℓ was. Fortunately, provided we are prepared to modify M_k when necessary, we can avoid this potential defect.

Suppose the columns of N form an orthonormal basis for the null-space of the full-rank matrix A, i.e., $AN = 0$. Suppose furthermore that $N^T M N$ is positive definite. Let

$$A = \begin{pmatrix} \bar{A} \\ a^T \end{pmatrix}$$

in which case

$$\bar{A}N = 0 \;\text{ and }\; a^T N = 0. \tag{3.8}$$

Then there is a vector n for which the columns of $(N \;\; n)$ form an orthonormal basis for the null-space of $\bar{A}$, i.e.,

$$\bar{A}N = 0, \;\; N^T n = 0 \;\text{ and }\; \bar{A}n = 0. \tag{3.9}$$

Now consider the matrix $M + \sigma a a^T$ for some scalar σ. Then

$$\begin{pmatrix} N^T \\ n \end{pmatrix} (M + \sigma a a^T)(N \;\; n) = \begin{pmatrix} N^T M N & N^T M n \\ n^T M N & n^T M n + \sigma (a^T n)^2 \end{pmatrix}, \tag{3.10}$$

where we have used the fact that $N^T a = 0$ from (3.8). Since the columns of $(A^T \;\; N)$ form a basis for $\mathbb{R}^n$, we may write

$$n = \bar{A}^T w + \alpha a + N v$$

for some vectors finite w and v and scalar α. Premultiplying by n^T, and using (3.9) and the orthonormality of $(N \;\; n)$ yields that $1 = \alpha a^T n$, from which we deduce that $a^T n \neq 0$. Thus we can ensure that the matrix (3.10) is positive definite by, if necessary, picking σ sufficiently large.

Of course, we are not basing our method on (3.10), but rather on being able to solve augmented systems like (3.5). In order to accommodate changes to M_k of the type suggested above, we actually need to solve systems of the form

$$\begin{pmatrix} M_k & A_C^T & A_D^T & A_A^T & 0 \\ A_C & 0 & 0 & 0 & 0 \\ A_D & 0 & 0 & 0 & I \\ A_A & 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 & S_D \end{pmatrix} \begin{pmatrix} s_\ell \\ t_C \\ t_D \\ t_A \\ u_\ell \end{pmatrix} = - \begin{pmatrix} g_\ell \\ c_C \\ 0 \\ c_A \\ 0 \end{pmatrix}, \tag{3.11}$$

where S_D is a diagonal matrix. To see why this is appropriate, on eliminating t_D and u_ℓ and using (3.4), we obtain

$$\begin{pmatrix} M_k + A_D^T S_D A_D & A_\ell^T \\ A_\ell & 0 \end{pmatrix} \begin{pmatrix} s_\ell \\ t_\ell \end{pmatrix} = - \begin{pmatrix} g_\ell \\ c_\ell \end{pmatrix}, \tag{3.12}$$

which is (3.3) with

$$M_\ell = M_k + A_D^T S A_D.$$

A diagonal entry in S_D need only be nonzero if the resulting K_ℓ would otherwise be nonstandard. Crucially, as before, the leading 3 by 3 block of (3.11) is simply K_k, and thus that the system may be written as

$$\begin{pmatrix} K_k & B_\ell^T \\ B_\ell & D_\ell \end{pmatrix} \begin{pmatrix} v_\ell \\ w_\ell \end{pmatrix} = - \begin{pmatrix} h_\ell \\ d_\ell \end{pmatrix}, \tag{3.13}$$

where

$$D_\ell = \begin{pmatrix} 0 & 0 \\ 0 & S_D \end{pmatrix}.$$

Thus (3.13) can be solved in the standard way using the factors of K_k and those of the Schur complement $S_l = D_\ell - B_l K_k^{-1} B_\ell^T$, and the factors of the latter can be updated as the working set changes.

To see this, suppose (without loss of generality) that we have added constraint terms whose Jacobian is A_A, and now intend to remove the first row from A_D. The resulting Schur complement is then

$$\begin{aligned} S(\sigma) = & \begin{pmatrix} 0 & 0 \\ 0 & \sigma \end{pmatrix} - \begin{pmatrix} -BK^{-1}B^T & -BK^{-1}b \\ -b^T K^{-1} B^T & -b^T K^{-1} b \end{pmatrix} \\ = & \begin{pmatrix} QR & v \\ v^T & \sigma + \beta \end{pmatrix} \\ = & \begin{pmatrix} Q & 0 \\ 0 & 1 \end{pmatrix} H^T H \begin{pmatrix} R & w \\ v^T & \sigma + \beta \end{pmatrix} = \bar{Q} \bar{R}(\sigma), \end{aligned}$$

where

$$B = \begin{pmatrix} A_A & 0 & 0 \end{pmatrix}, \; b^T = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix},$$
$$QR = -BK^{-1}B^T, \;\; v = -BK^{-1}b, \;\; \beta = -b^T K^{-1} b \text{ and } w = Q^T v,$$

and the orthonormal matrix

$$H = \begin{pmatrix} \bar{H} & h \end{pmatrix}$$

is a product of plain rotations chosen to eliminate the spike v^T (see Gill et al., 1974). But then

$$\begin{aligned} \bar{R}(\sigma) = & \; H \begin{pmatrix} R & w \\ v^T & \sigma + \beta \end{pmatrix} = \begin{pmatrix} \bar{H} R + h v^T & \bar{H} w + \beta h \end{pmatrix} + \sigma \begin{pmatrix} 0 & h \end{pmatrix} \\ = & \; \bar{R}(0) + \sigma \begin{pmatrix} 0 & h \end{pmatrix} \end{aligned}$$

and the introduction of σ simply adds σh to the last row of the updated upper triangular factor $\bar{R}(0)$. Fortunately, the updated orthonormal matrix is

$$\bar{Q} = \begin{pmatrix} Q & 0 \\ 0 & 1 \end{pmatrix} H^T = \begin{pmatrix} Q \bar{H}^T \\ h^T \end{pmatrix},$$

and hence h is available. We can evaluate the sign of the determinant of $\bar{R}(0)$, and if this indicates that the new K is nonstandard, add a sufficiently large σ to change the sign of the last diagonal of $\bar{R}$.

3.2.5 Further improvements

As described above, the dimension of the Schur complement S_l increases every time a change is made to the working set. It is also possible, and desirable, to note that when a non-reference constraint is chosen to leave the Schur complement, or when a previously-deleted reference constraint wishes to re-enter, the same effect may be achieved by *removing* an appropriate row from S_l. The advantage is that a major iteration can then span a large number of minor iterations, and this prolongs the usefulness of the current reference matrix K_k. Further details are given by Gould and Toint (2001*a*).

3.3 Other details

As we briefly mentioned, rather than use a two-phase method to solve (1.1), our implemented version is based on a minimization of the ℓ_1 exact penalty function

$$g^T x + \tfrac{1}{2} x^T H x + \rho \|\min(Ax - b, 0)\|_1, \tag{3.14}$$

for some appropriately large value of the penalty parameter $\rho > 0$. Such an approach was first considered for QP by Conn and Sinclair (1975) and Han (1981), and aside from the step-length selection, is essentially based on the preceeding algebraic manipulations. Our Fortran 90 package `HSL_VE19` is close to completion and will shortly be available as part of HSL (2000); a more basic version, `qpa` (which differs simply in that the older-but-widely available HSL matrix factorization routine `MA27` is used instead of its more up-to-date-but-proprietary companion `MA57`), will ultimately be part of our evolving nonlinear programming library *GALAHAD*. The implemented algorithm uses a number of "tricks" not mentioned here, but is based on that described above. See Gould and Toint (2001*a*) for more details.

4 An interior-point method

A second class of methods is based upon the sequential minimization of logarithmic barrier functions as advocated by Fiacco and McCormick (1968). Although maligned in the 1970s and early 1980s, these interior-point methods have subsequently proved to be probably the most successful means of solving large-scale convex optimization problems. The state-of-the-art for convex QP is represented by the methods proposed by Carpenter, Lustig, Mulvey and Shanno (1993), Vanderbei (1994), Zhang (1994), Wright and Zhang (1996), Ye (1997) and Altman and Gondzio (1998), most of which have resulted in software packages—the LOQO package of Vanderbei and others is capable of dealing with non-convex (and even non-quadratic) problems.

Since we are interested in non-convex problems, we aim to extract the best aspects from the above, while including precautions to ensure global convergence to a weak second-order critical point. To simplify matters, we shall consider the alternative version (1.2) of our problem; we could have used the original (1.1) version, but there are implementational advantages with the alternative.

4.1 The barrier problem and optimality

The barrier function approach replaces (1.2) by a sequential solution of the *logarithmic barrier* problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; \phi(x, \mu) = g^T x + \tfrac{1}{2} x^T H x - \mu e^T \log x \;\text{ subject to }\; Ax = b, \tag{4.1}$$

where e is the vector of ones, $\log x$ is the vector with components $\log x_i$, and μ is a strictly positive barrier parameter. We shall insist that there is a strictly feasible *interior point* x^0, that is that $Ax^0 = b$ and $x^0 > 0$, and that all iterates remain strictly feasible—the logarithmic singularity for the barrier function $\phi(x, \mu)$ and the act of minimization makes it impossible for iterates to approach the boundary too closely. We note that many of the interior-point methods we mentioned above do not require an interior point, but it is not clear to us then which are the best ways to balance infeasibility against optimality in the non-convex case.

A first-order critical point $x(\mu)$ for (4.1) satisfies

$$\begin{array}{cl} Ax(\mu) = b \ \text{ and } \ x(\mu) > 0 & (\textit{strict primal feasibility}) \\ Hx(\mu) + g + A^T y(\mu) - z(\mu) = 0 \ \text{ and } \ z(\mu) > 0 & (\textit{strict dual feasibility}) \\ x_i(\mu) \cdot z_i(\mu) = \mu \ \text{ for all } \ i & (\textit{perturbed complementary slackness}), \end{array} \tag{4.2}$$

where $y(\mu)$ are Lagrange multipliers for the equality constraints, $z(\mu) \stackrel{\text{def}}{=} \mu X(\mu)^{-1} e$, and (generically) X is the diagonal matrix whose entries are the x_i. Comparing this with the first-order criticality conditions for (1.2),

$$\begin{array}{cl} Ax^* = b \ \text{ and } \ x^* \geq 0 & (\textit{primal feasibility}) \\ Hx^* + g + A^T y^* - z^* = 0 \ \text{ and } \ z^* \geq 0 & (\textit{dual feasibility}) \\ x_i^* \cdot z_i^* = 0 \ \text{ for all } \ i & (\textit{complementary slackness}), \end{array} \tag{4.3}$$

it is clear that

$$\lim_{\mu \to 0_+} x(\mu) = x^*, \ \lim_{\mu \to 0_+} y(\mu) = y^* \ \text{ and } \ \lim_{\mu \to 0_+} z(\mu) = z^*$$

under an appropriate non-singularity condition. It is also possible to draw an equivalence between the weak second-order criticality conditions for the two problems (see Auslender, 1979), although the strong second-order conditions do not coincide since there are problems for which the limit of a strong-second order critical point for (4.1) turns out to be a *maximizer* of (1.2) (see Gould and Toint, 1999). Thus the most we can expect of an interior-point method based on a (logarithmic) barrier function is that we achieve a weak second-order critical point.

4.2 A barrier algorithm

The obvious barrier algorithm comprises an inner iteration, in which an approximate (second-order) critical point of the barrier (4.1) is determined, and an outer iteration, in which the barrier parameter, and attendant inner-iteration convergence tolerances, are adjusted (and ultimately reduced to zero).

In order to state our preferred methods, we shall use the following terminology. A scalar valued function $\epsilon(\alpha) \geq 0$ is a *forcing* function if $\epsilon(\alpha) = 0$ if and only if $\alpha = 0$. Given a second-order sufficient matrix M and a null-space basis matrix N for A, we define the semi-norm $\|\cdot\|_{[M,N]}$ so that

$$\|r\|_{[M,N]} = \|N^T r\|_{(N^T M N)^{-1}};$$

it is straightforward to show that this may be more conveniently calculated as $\|r\|_{[M,N]} = -r^T v$, where v satisfies (2.12). We shall use this semi-norm to measure distance from the null-space of A. In addition, for any symmetric matrix B, we let

$$\lambda^{\min}_{M,N}(B) = \lambda^{\min}((N^T M N)^{-\frac{1}{2}} N^T B N (N^T M N)^{-\frac{1}{2}})$$

where $\lambda^{\min}$ denotes the smallest (leftmost) eigenvalue of a given matrix.

4.2.1 The inner iteration

In principle, the inner iteration appears straightforward. The subproblem (4.1) is a (non-convex) linearly constrained optimization problem for which we are given a feasible interior point. Thus, enforcing feasibility for the linear constraints $Ax = b$ is simply a matter of choosing corrections s for which $As = 0$. Although linesearch-based methods for (4.1) are possible, we are attracted by the alternative, trust-region-based approaches, since their convergence properties are easy to derive—similar properties for linesearch methods are possible with some care.

Given a strictly interior estimate x^C, an iteration of a typical second-order-model trust-region approach centers on the approximate solution of the model problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; q^C(s) \stackrel{\text{def}}{=} g^{C^T}x + \tfrac{1}{2}s^T B^C s \;\text{ subject to }\; As = 0 \;\text{ and }\; \|s\|_{M^C} \le \Delta^C, \tag{4.4}$$

for some appropriate symmetric matrix B^C, gradient $g^C = g + Hx^C - \mu(X^C)^{-1}e$, radius Δ^C and semi-norm $\|\cdot\|_{M^C}$. A pure primal-Newton method would choose B^C to be the Hessian of the barrier function $H+\mu(X^C)^{-2}$, but an alternative Newton derivation based on the nonlinear system (4.2) (see, for instance, Conn et al., 2000*a*) and significant numerical experience suggest that the choice

$$B^C = H + (X^C)^{-1}Z^C$$

is usually far superior.

Of course, (4.4) is precisely of the form (2.14) (for the given data), and thus the methods developed in Section 2.4 and its predecessors are appropriate. The obvious choice for the preconditioner is of the form

$$M = G + (X^C)^{-1}Z^C \tag{4.5}$$

for some approximation G of H, the term $(X^C)^{-1}Z^C$ being essential to cope with the increasing ill-conditioning that results when one or more components x_i converges to zero.

Having computed a suitable approximation s^C to the solution of (4.4) using the PPCG/GLTR method discussed earlier—a single iteration suffices since this gives a Cauchy point for the model (for this and other trust-region technicalities, see Conn et al., 2000*a*) although further iterations are essential for rapid convergence of the overall iteration—the remaining trust-region method is essentially standard. In particular x^+ is set to $x^C + s^C$ so long as the ratio of actual to predicted reduction,

$$\rho = \frac{\phi(x^C + s^C, \mu) - \phi(x^C, \mu)}{q^C(s^C)},$$

is sufficiently positive, and left at x^C otherwise, while the trust-region radius is increased if the step s^C has encountered the current trust region boundary and ρ is close to one, and always decreased when $x^+ = x^C$. There are only two exceptional details. Firstly, any predicted step that lies outside the feasible region is automatically rejected—a variant in which extra constraints $x^C + s \ge \eta$, for some small $\eta > 0$ are imposed on the model (4.4) is possible (see, for example, Byrd, Hribar and Nocedal, 1999, and Byrd, Gilbert and Nocedal, 2000), but the model problem is then far harder to solve. Secondly, dual variables are updated according to the rather complicated-looking procedure of projecting the primal-dual estimates $\mu(X^C)^{-1}e - (X^C)^{-1}Z^C s^C$ into the interval

$$\left[\tfrac{1}{2}\min\left(e, z^C, \mu(X^+)^{-1}e\right), \left(2e, z^C, 2\mu^{-1}e, 2\mu(X^+)^{-1}e\right)\right].$$

This procedure is designed to allow considerable freedom, and to encourage primal-dual-like steps, while still allowing a rigorous convergence analysis—the values $\frac{1}{2}$ and 2 are not critical, but representative of numbers smaller and larger than one respectively.

The standard first-order trust-region convergence theory is easily adapted to cope with this slight modification. In particular Conn, Gould, Orban and Toint (2000*b*) show that if (i) A is of full rank, (ii) $\phi(x,\mu)$ is bounded from below on the feasible region, (iii) B is bounded, (iv) no z_i grows faster than a (possibly huge) constant times the reciprocal of its corresponding x_i, (v) the preconditioner M is uniformly second-order sufficient, and (vi) at least one iteration of the PPCG/GLTR method is taken, then any limit of the trust-region method sketched above is first-order critical for (4.1). If, in addition, (vii) B converges to $H + X^{-1}Z$, (viii) z converges to $\mu X^{-1}e$ and (ix) PPCG/GLTR is allowed to generate a suitable eigenpoint, at least one of the limit points satisfies the weak second-order optimality conditions for (4.1).

Algorithm 4.1: An iteration for the outer minimization

Input. A barrier parameter $\mu^C > 0$ and the forcing functions $\epsilon^{\mathrm{C}}(\mu)$, $\epsilon^{\mathrm{D}}(\mu)$ and $\epsilon^{\mathrm{E}}(\mu)$ are given.

Inner Minimization. Approximately minimize the log-barrier function $\phi(x, \mu^C)$. Stop this inner algorithm as soon as an inner iterate (x^+, z^+) is found for which

$$\begin{aligned} Ax^+ &= b, && (4.6)\\ (x^+, z^+) &> 0, && (4.7)\\ \|X^+z^+ - \mu^C e\|_2 &\leq \epsilon^{\mathrm{C}}(\mu^C), && (4.8)\\ \|Hx^+ + g - z^+\|_{[M^+,N]} &\leq \epsilon^{\mathrm{D}}(\mu^C), \quad \text{and} && (4.9)\\ \lambda^{\min}_{M^+,N}\left[H + (X^+)^{-1}Z^+\right] &\geq -\epsilon^{\mathrm{E}}(\mu^C), && (4.10) \end{aligned}$$

where M^+ is a second-order sufficient preconditioning matrix. Choose $\mu^+ < \mu^C$, and perform the next inner minimization.

4.2.2 The outer iteration

We now turn to the following outer iteration.

The aforementioned convergence theory of the inner iteration ensures that the given trust-region method will provide a suitable (x^+, z^+) after a finite number of iterations. Measuring convergence in terms of the semi-norm (4.9) is convenient, since precisely this norm is used as a measure in the PPCG/GLTR iteration. The Lanczos aspect of this iteration naturally records the eigenvalues of $(N^TMN)^{-\frac{1}{2}}N^TBN(N^TMN)^{-\frac{1}{2}}$, which justifies the rather-strange termination test (4.10).

The reader might be concerned that the scaling used here might cause difficulties since the matrix (4.5) will naturally diverge if any component of x approaches the boundary of the feasible region. Fortunately these difficulties do not arise since this scaling is designed to counteract the same divergent effects as B approaches $H + X^{-1}Z$. Indeed, the test (4.8) is weaker than measuring $N^T(Hx^+ + g - z^+)$ in an unscaled norm, and thus the inner iteration may terminate earlier than with such an alternative.

Conn et al. (2000*b*) show that, under slightly more restrictive assumptions on the forcing functions $\epsilon^{\mathrm{C}}(\mu)$, $\epsilon^{\mathrm{D}}(\mu)$ and $\epsilon^{\mathrm{E}}(\mu)$, there is at least one subsequence generated by the iteration given in Algorithm 4.1 that converges to a weak second-order critical point for (1.2).

4.2.3 Asymptotic convergence

One further aspect of interest is the ultimate convergence rate of the method we have just summarised. Gould, Orban, Sartenaer and Toint (2000) consider the case where the convergence forcing functions in (4.8)–(4.9) have the form $\epsilon^{\mathrm{C,D}}(\mu) = \mu^{1+\alpha^{\mathrm{C,D}}}$, and the barrier parameter parameter update is of the form $\mu^+ = (\mu^C)^{1+\alpha}$. They show that it is possible to choose $\alpha^{\mathrm{C,D}}$ and α so that the primal-dual iterates converge Q-superlinearly—the rate may be made arbitrarily close to Q-quadratic—to (x^*, z^*) under suitable regularity and strict complementarity conditions. Perhaps more surprising, the same rate is achieved componentwise in most cases.

4.3 Other aspects

The method we have described above has been implemented as the HSL (2000) Fortran 90 package `HSL_VE12`; as before, a more basic version, `qpb`, will shortly be available as part of our nonlinear programming library *GALAHAD*. Further details are given by Conn et al. (2000*b*).

Our method requires a strictly feasible initial starting point, and we use another HSL package HSL_VE13 (itself a primal-dual infeasible interior-point method for *convex* quadratic programming with separable objectives, based on Zhang's method), to find an approximation to the analytic centre of the feasible region. In the event that the size of the iterate exceeds some prescribed upper bound (as may happen if there is no analytic centre), the last point with a norm smaller than this bound is taken for the initial point. In principle, any good interior-point method would suffice, but in any event, this part of the calculation is usually very efficient.

5 Numerical comparisons

We now compare the two approaches. We should warn the reader that although the interior-point code HSL_VE12 has been released, its active-set competitor HSL_VE19 is still under development, so the results (and any conclusions drawn) should be considered provisional. That said, we do not expect HSL_VE19 to change very dramatically.

There are essentially two uses for a QP code, namely to solve a new problem from scratch without any knowledge of the final active set (a so-called *cold* start), and to solve a small perturbation of an existing problem with full knowledge of the solution of the latter (a *warm* start). Cold-start problems normally arise from specific one-off applications, while SQP methods for nonlinear programming are a rich source of warm-start problems. In this section we shall consider both.

5.1 Cold-start problems

The CUTE test set (see Bongartz, Conn, Gould and Toint, 1995) contains a large number of QP test examples. We have selected all of the larger, and variable-dimensioned, examples, and applied our two contending QP packages to them. All of our experiments were performed on a single processor of a Compaq AlphaServer DS20 with 3.5 Gbytes of RAM.

The interior-point code HSL_VE12 is used with supplied default settings. Since defaults have not yet been finalized for the active-set package HSL_VE19, our runs simply select M as the diagonal of H, with the remaining modifications of (and simplifications to) K being exactly as described for HSL_VE12. An upper bound of 75 is set on the maximum permitted dimension for the Schur complement S (some experimentation indicates that this is a reasonable value), the initial working set is simply the set of equality constraints (with dependencies automatically removed), and a variety of tests are performed to guarantee that the residuals of active constraints really are small. For full details, see Gould and Toint (2001*a*).

In Table 5.1, we report results for what are, by today's standards, small problems. We report the number of iterations (its) and CPU times (rounded to the nearest tenth of a second) for both methods. For the active-set method, we also include the number of factorizations (facts) of (2.13) required; the interior-point approach requires one factorization per iteration. The better of the two approaches for each problem (if it is significant) is indicated in bold typeface. As one has been lead to expect, the interior-point approach is already starting to show its superiority over the active-set one, although the results are far from completely one sided. Qualitatively similar results were obtained on the older HSL active-set QP code VE09 (see Gould, 1991).

				HSL_VE19 active-set			HSL_VE12 barrier	
Problem	n	m	type	its	facts	time	its	time
AUG2DCQP	3280	1600	C	454	9	3.3	22	**1.4**
AUG2DQP	3280	1600	C	573	21	20.4	20	**1.3**
AUG3DCQP	3873	1000	C	437	8	3.8	22	**1.8**
AUG3DQP	3873	1000	C	608	19	16.7	21	**1.7**
BLOCKQP1	2005	1001	NC	4	5	0.6	9	0.5
BLOCKQP2	2005	1001	NC	1005	30	8.9	18	**0.9**
BLOCKQP3	2005	1001	NC	4	5	0.6	9	0.5
BLOWEYA	2002	1002	NC	22	4	1.5	7	**1.3**

BLOWEYB	2002	1002	NC	20	4	1.3	1	**0.2**
BLOWEYC	2002	1002	NC	23	4	2.2	7	**1.1**
CVXQP1	1000	500	C	802	13	19.0	28	**4.8**
CVXQP2	1000	250	C	568	10	7.2	19	**2.1**
CVXQP3	1000	750	C	562	9	24.0	29	**6.7**
DEGENQP	20	8010	C	110	3	8.0	6	**1.8**
DUALC1	9	215	C	62	5	**0.0**	29	0.4
DUALC2	7	229	C	22	3	**0.0**	27	0.3
DUALC5	8	278	C	14	3	**0.0**	15	0.3
DUALC8	8	503	C	38	3	**0.0**	19	0.6
GOULDQP2	5	2	C	9	4	0.0	2	0.0
GOULDQP3	699	349	C	211	5	0.6	5	**0.1**
KSIP	20	1001	C	1692	21	**1.0**	13	1.9
MOSARQP1	900	600	C	1681	34	10.1	17	**1.0**
MOSARQP2	900	600	C	442	17	1.2	14	**0.9**
NCVXQP1	1000	500	NC	686	18	1.8	56	**0.9**
NCVXQP2	1000	500	NC	1208	28	3.3	46	**0.8**
NCVXQP3	1000	500	NC	908	21	**4.0**	150	7.1
NCVXQP4	1000	250	NC	768	22	1.3	48	**0.5**
NCVXQP5	1000	250	NC	775	24	1.4	57	**0.7**
NCVXQP6	1000	250	NC	777	21	**1.8**	99	2.3
NCVXQP7	1000	750	NC	497	10	**2.4**	101	23.2
NCVXQP8	1000	750	NC	800	21	**4.5**	113	5.5
NCVXQP9	1000	750	NC	802	15	8.4	70	**5.4**
POWELL20	1000	1000	C	1319	22	2.2	83	**0.9**
PRIMALC1	230	9	C	221	8	0.2	33	0.2
PRIMALC2	231	7	C	240	7	**0.2**	27	0.4
PRIMALC5	287	8	C	290	7	0.2	36	0.2
PRIMALC8	520	8	C	560	11	0.6	48	0.6
PRIMAL1	325	85	C	81	4	**0.2**	23	0.5
PRIMAL2	649	96	C	97	5	**0.4**	13	1.0
PRIMAL3	745	111	C	110	5	**0.7**	17	2.0
PRIMAL4	1489	75	C	70	4	**0.6**	18	2.9
QPBAND	1000	500	C	1977	20	5.2	9	**0.3**
QPNBAND	1000	500	NC	28	4	**0.0**	10	0.2
QPCBOEI1	384	351	C	1559	18	1.6	71	1.4
QPCBOEI2	143	166	C	788	21	0.4	63	0.5
QPCSTAIR	467	356	C	653	15	**1.1**	72	2.5
QPNBOEI1	384	351	NC	7277	109	8.9	92	**2.2**
QPNBOEI2	143	166	NC	509	13	**0.4**	75	0.8
QPNSTAIR	467	356	NC	950	12	**2.4**	78	3.0
SOSQP1	2000	1001	SOS	9	5	0.6	5	**0.3**
STCQP1	4097	2052	C	355	7	**13.0**	24	29.3
STCQP2	4097	2052	C	117	4	**2.9**	17	13.6
STNQP1	4097	2052	NC	603	16	16.9	15	**11.4**
STNQP2	4097	2052	NC	770	21	10.0	16	**1.3**
UBH1	909	600	C	6	3	0.1	3	0.1
YAO	2002	2000	C	2258	51	8.6	76	**2.7**

Table 5.1: Numerical results: small problems. The parameters n and m are the numbers of variables and general constraints (not including simple bounds). The types C and NC refer to convex and non-convex problems, while SOS indicates a problem whose Hessian is second-order sufficient with respect to all feasible active sets.

In Tables 5.2 and 5.3, we exhibit specimen results for medium and large-scale instances of the variable-dimensional problems. We include these simply to show that the advantages of interior-point methods over conventional active-set approaches are now clear. Indeed, no results for the active-set method are given for the largest problems, simply because the CPU times required are excessive. These results simply reinforce the impressions given by the preliminary ones presented by Conn et al. (2000*b*). In addition, we are now able to give results for non-convex problems involving up to half a million unknowns (and a similar magnitude of constraints), suggesting that our approach is capable of solving at least some large problems.

Problem	n	m	type	HSL_VE19 active-set its	facts	time	HSL_VE12 barrier its	time
AUG2DCQP	20200	10000	C	15389	210	837.7	23	**10.3**
AUG2DQP	20200	10000	C	-	-	> 1800.0	23	**10.3**
AUG3DCQP	27543	8000	C	5368	70	794.6	17	**73.7**
AUG3DQP	27543	8000	C	-	-	> 1800.0	18	**75.9**
BLOCKQP1	20005	10001	NC	4	5	**2.8**	9	5.2
BLOWEYA	20002	10002	NC	12	5	**6.9**	8	32.5
DEGENQP	50	1250251	C	356	3	525.6	8	**59.6**
GOULDQP2	19999	9999	C	8002	101	622.9	2	**0.6**
KSIP	20	10001	C	346	3	**2.8**	15	10.6
MOSARQP1	20000	10000	C	15743	252	1727.7	17	**25.6**
NCVXQP1	10000	5000	NC	18492	323	1165.9	98	**68.8**
NCVXQP4	10000	2500	NC	7824	207	155.4	78	**5.6**
NCVXQP7	10000	7500	NC	-	-	> 1800.0	67	**348.7**
POWELL20	10000	10000	C	13055	175	237.3	185	**21.3**
QPBAND	10000	5000	C	10435	135	344.2	12	**2.3**
QPNBAND	10000	5000	NC	10147	202	177.1	14	**1.7**
SOSQP1	100000	50001	SOS	11	3	85.4	6	**6.8**
STCQP1	8193	4095	C	881	14	230.4	26	**84.1**
STCQP2	8193	4095	C	174	6	**30.1**	15	84.1
STNQP1	8193	4095	NC	1174	47	111.3	18	**64.2**
STNQP2	8193	4095	NC	1491	57	39.2	19	**2.6**
UBH1	18009	12000	C	6	3	**0.9**	6	2.3
YAO	10002	10000	C	46019	592	1169.5	132	**19.0**

Table 5.2: Numerical results: specimen medium problems. The captions are as for Tables 5.1.

Problem	n	m	type	HSL_VE12 barrier its	time
GOULDQP2	200001	100000	C	17	**52**
GOULDQP3	200001	100000	C	46	**154**
POWELL20	100000	100000	C	148	**234**
QPBAND	100000	50000	C	13	**157**
QPBAND	200000	100000	C	17	**1138**
QPBAND	400000	200000	C	17	**2304**
QPBAND	500000	250000	C	17	**2909**
QPNBAND	100000	50000	NC	12	**32**
QPNBAND	200000	100000	NC	13	**71**
QPNBAND	400000	200000	NC	14	**156**
QPNBAND	500000	250000	NC	13	**181**

Table 5.3: Numerical results: specimen large problems. All runs for HSL_VE19 exceeded the time limit of 7200 seconds. The captions are as for Tables 5.1.

5.2 Warm-start problems

We now turn our attention to problems which are small perturbations of those whose solutions we already know. To study this case, we consider the same examples we compared in Table 5.2, and use the solution (and its active set) obtained from the HSL_VE12 run. We now perturb all the problem data very slightly (by uniformly distributed random numbers in the range 0 to 10^{-8}), and resolve the problems starting at the previously obtained "solution". This "solution" is easily exploited by the active set method, but ignored by the interior-point one since HSL_VE12 picks its own interior starting point—warm-starting interior-point methods is notoriously hard (see, however, Gondzio, 1998, and Yildirim and Wright, 2000, for some progress in this area).

					HSL_VE19 active-set		HSL_VE12 barrier	
Problem	n	m	type	misc	its	time	its	time
AUG2DCQP	20200	10000	C	1	5	**1.2**	30	10.1
AUG2DQP	20200	10000	C	8	14	**6.6**	26	9.7
AUG3DCQP	27543	8000	C	66	78	**16.7**	20	77.8
AUG3DQP	27543	8000	C	0	-	$>$ 1800.0	18	**71.9**
BLOCKQP1	20005	10001	NC	0	-	$>$ 1800.0	43	**16.3**
BLOWEYA	20002	10002	NC	0	12	**2.8**	-	$>$ 1800.0
DEGENQP	50	1250251	C	23694	201	669.3	8	**57.7**
GOULDQP2	19999	9999	C	0	-	$>$ 1800.0	2	**0.5**
KSIP	20	10001	C	9400	350	**2.8**	14	8.6
MOSARQP1	20000	10000	C	328	284	81.3	21	**31.3**
NCVXQP1	10000	5000	NC	3905	4971	239.6	38	**32.5**
NCVXQP4	10000	2500	NC	196	300	**4.8**	104	7.0
NCVXQP7	10000	7500	NC	3182	6026	547.6	23	**124.2**
POWELL20	10000	10000	C	0	30	**1.6**	1364	173.8
QPBAND	10000	5000	C	130	77	**1.0**	12	2.2
QPNBAND	10000	5000	NC	0	4	**0.1**	14	1.7
SOSQP1	100000	50001	SOS	0	-	$>$ 1800.0	8	11.4
STCQP1	8193	4095	C	0	20	**65.1**	15	80.2
STCQP2	8193	4095	C	0	10	**3.8**	12	76.4
STNQP1	8193	4095	NC	0	27	63.0	11	**61.7**
STNQP2	8193	4095	NC	0	6	2.3	13	**1.8**
UBH1	18009	12000	C	0	6	**0.8**	6	2.3
YAO	10002	10000	C	0	8	**1.0**	166	25.8

Table 5.4: Warm-started active-set versus cold-started interior point methods: specimen medium problems. The column headed misc gives the number of constraints which have been misclassified as being active from the original problem, but inactive in the perturbed version (and vice versa). The remaining captions are as for Tables 5.1.

In some cases the benefits of warm-starting the active set method are clear to see in Table 5.4. In particular significant savings are made, both compared to the interior-point approach, and relative to the cold-started results (admittedly for the original not the perturbed problem) given in Table 5.2. However the results are far from uniformly better.

For problems AUG3DQP, BLOCKQP1 and SOSQP1, the perturbation turned the convex or second-order sufficient problem into a non-convex one with a radically different solution. Similarly for NCVXQP1 and NCVXQP7, perturbations of the already non-convex and ill-conditioned problem lead to different solutions. For DEGENQP and GOULDQP2, perturbing problems with highly-degenerate optimal solutions leads to completely incorrect guesses for the resulting active set. Thus it appears that while it is tempting to believe that a warm-started active set method is the ideal choice when solving a sequence of closely related problems, this may not be the case when the problem is either degenerate or nonconvex, and actually the cold-started interior-point approach is preferable. With hindsight, perhaps we should not be surprised, but we are certainly disappointed as we had hoped that the active-set method would be the obvious choice for “warm-started” applications like ”asymptotic” iterations in SQP methods.

Particularly poor behaviour for the barrier approach can be seen for BLOWEYA and POWELL20, where the constraint regions for the perturbed problems are tiny.

6 Conclusions

We have described two very different methods for nonconvex quadratic programming. The interior-point approach, exemplified by HSL_VE12, proves to be superior in most, but not all, cases when the problems are reasonably large. When a sequence of closely-related problems are solved, the active-set approach is sometimes to be preferred, although this is not the case when the problem is degenerate or ill-conditioned.

As we have stressed, these conclusions are somewhat provisional since the active-set code `HSL_VE19` is still under development. In particular, sensible ways to update the penalty parameter associated with the merit function (3.14), and schemes to refine cold-started working sets are still under investigation.

One further aspect that we are currently considering is the idea of presolving the QP (see, Gould and Toint, 2001*b*). The idea is to apply inexpensive elementary techniques to the problem, with the aims of tightening the feasible region, eliminating obviously inactive constraints, removing fixed (or implied fixed) variables, etc. The QP algorithm is then applied to the presolved problem, and after a solution is found, the transformations made in the presolve are reversed. Such techniques have proved to be most successful for linear programming (see Andersen and Andersen, 1995, Andersen, Gondzio, Mészáros and Xu, 1996, and Gondzio, 1997) and our expectation is that the same will be true for the quadratic case. The resulting presolve code, along with slightly less-sophisticated versions of `HSL_VE19` and `HSL_VE12`, named `qpa` and `qpb` (respectively), will ultimately be part of our evolving nonlinear programming library *GALAHAD*.

Acknowledgments

Nick Gould would like to thank Oxford University Computing Laboratory for providing a tranquil atmosphere in which to write. We both express our gratitude to our collaborators, Andy Conn, Mary Beth Hribar, Jorge Nocedal, Dominique Orban and Annick Sartenaer, for their priceless contributions to the work described here, and again to Andy and Annick for their useful comments on this manuscript.

References

A. Altman and J. Gondzio. Regularized symmetric indefinite systems in interior point methods for linear and quadratic optimization. Logilab Technical Report 1998.6, Department of Management Sciences, University of Geneva, Geneva, Switzerland, 1998.

E. D. Andersen and K. D. Andersen. Presolving in linear-programming. *Mathematical Programming, Series A*, **71**(2), 221–245, 1995.

E. D. Andersen, J. Gondzio, C. Mészáros, and X. Xu. Implementation of interior point methods for large scale linear programming. *In* T. Terlaky, ed., 'Interior Point Methods in Mathematical Programming', pp. 189–252, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

A. Auslender. Penalty methods for computing points that satisfy second order necessary conditions. *Mathematical Programming*, **17**(2), 229–238, 1979.

J. Bisschop and A. Meeraus. Matrix augmentation and partitioning in the updating of the basis inverse. *Mathematical Programming*, **13**(3), 241–254, 1977.

P. T. Boggs and J. W. Tolle. Sequential quadratic programming. *Acta Numerica*, **4**, 1–51, 1995.

I. Bongartz, A. R. Conn, N. I. M. Gould, and Ph. L. Toint. CUTE: Constrained and unconstrained testing environment. *ACM Transactions on Mathematical Software*, **21**(1), 123–160, 1995.

J. M. Borwein. Necessary and sufficient conditions for quadratic minimality. *Numerical Functional Analysis and Optimization*, **5**, 127–140, 1982.

J. R. Bunch and L. C. Kaufman. Some stable methods for calculating inertia and solving symmetric linear equations. *Mathematics of Computation*, **31**, 163–179, 1977.

J. R. Bunch and B. N. Parlett. Direct methods for solving symmetric indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, **8**(4), 639–655, 1971.

R. H. Byrd, J. Ch. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, **89**(1), 149–185, 2000.

R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large scale nonlinear programming. *SIAM Journal on Optimization*, **9**(4), 877–900, 1999.

T. J. Carpenter, I. J. Lustig, J. M. Mulvey, and D. F. Shanno. Higher-order predictor-corrector interior point methods with application to quadratic objectives. *SIAM Journal on Optimization*, **3**(4), 696–725, 1993.

Y. Chabrillac and J.-P. Crouzeix. Definiteness and semidefiniteness of quadratic forms revisited. *Linear Algebra and its Applications*, **63**, 283–292, 1984.

T. F. Coleman. Linearly constrained optimization and projected preconditioned conjugate gradients. *In* J. Lewis, ed., 'Proceedings of the Fifth SIAM Conference on Applied Linear Algebra', pp. 118–122, SIAM, Philadelphia, USA, 1994.

T. F. Coleman and A. Verma. A preconditioned conjugate gradient approach to linear equality constrained minimization. Technical report, Department of Computer Sciences, Cornell University, Ithaca, New York, USA, July 1998.

A. R. Conn and N. I. M. Gould. On the location of directions of infinite descent for nonlinear programming algorithms. *SIAM Journal on Numerical Analysis*, **21**(6), 302–325, 1984.

A. R. Conn and J. W. Sinclair. Quadratic programming via a non-differentiable penalty function. Technical Report CORR 75/15, Faculty of Mathematics, University of Waterloo, 1975.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-region methods*. SIAM, Philadelphia, 2000*a*.

A. R. Conn, N. I. M. Gould, D. Orban, and Ph. L. Toint. A primal-dual trust-region algorithm for non-convex nonlinear programming. *Mathematical Programming*, **87**(2), 215–249, 2000*b*.

B. L. Contesse. Une caractérisation complète des minima locaux en programmation quadratique. *Numerische Mathematik*, **34**(3), 315–332, 1980.

I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM Transactions on Mathematical Software*, **9**(3), 302–325, 1983.

I. S. Duff, N. I. M. Gould, J. K. Reid, J. A. Scott, and K. Turner. The factorization of sparse symmetric indefinite matrices. *IMA Journal of Numerical Analysis*, **11**, 181–204, 1991.

I. S. Duff, J. K. Reid, N. Munksgaard, and H. B. Neilsen. Direct solution of sets of linear equations whose matrix is sparse, symmetric and indefinite. *Journal of the Institute of Mathematics and its Applications*, **23**, 235–250, 1979.

A. V. Fiacco and G. P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. J. Wiley and Sons, Chichester, England, 1968. Reprinted as *Classics in Applied Mathematics 4*, SIAM, Philadelphia, USA, 1990.

R. Fletcher. A general quadratic programming algorithm. *Journal of the Institute of Mathematics and its Applications*, **7**, 76–91, 1971.

R. Fletcher. Factorizing symmetric indefinite matrices. *Linear Algebra and its Applications*, **14**, 257–272, 1976.

R. Fletcher. Quadratic programming. *In* 'Practical Methods of Optimization', chapter 10, pp. 229–258. J. Wiley and Sons, Chichester, England, second edn, 1987*a*.

R. Fletcher. Recent developments in linear and quadratic programming. *In* A. Iserles and M. J. D. Powell, eds, 'State of the Art in Numerical Analysis. Proceedings of the Joint IMA/SIAM Conference', pp. 213–243. Oxford University Press, Oxford, England, 1987*b*.

P. E. Gill and W. Murray. Numerically stable methods for quadratic programming. *Mathematical Programming*, **14**(3), 349–372, 1978.

P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, **28**, 505–535, 1974.

P. E. Gill, W. Murray, and M. H. Wright. Quadratic programming. *In* 'Practical Optimization', chapter 5.3.2–5.4.1, pp. 177–184. Academic Press, London, England, 1981.

P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. A Schur-complement method for sparse quadratic programming. *In* M. G. Cox and S. J. Hammarling, eds, 'Reliable Scientific Computation', pp. 113–138, Oxford University Press, Oxford, England, 1990.

P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Inertia-controlling methods for general quadratic programming. *SIAM Review*, **33**(1), 1–36, 1991.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, second edn, 1989.

J. Gondzio. Presolve analysis of linear programs prior to applying an interior point method. *INFORMS Journal on Computing*, **9**(1), 73–91, 1997.

J. Gondzio. Warm start of the primal-dual method applied in the cutting plane scheme. *Mathematical Programming*, **83**(1), 125–143, 1998.

N. I. M. Gould. On practical conditions for the existence and uniqueness of solutions to the general equality quadratic-programming problem. *Mathematical Programming*, **32**(1), 90–99, 1985.

N. I. M. Gould. An algorithm for large-scale quadratic programming. *IMA Journal of Numerical Analysis*, **11**(3), 299–324, 1991.

N. I. M. Gould and Ph. L. Toint. A note on the convergence of barrier algorithms to second-order necessary points. *Mathematical Programming*, **85**(2), 433–438, 1999.

N. I. M. Gould and Ph. L. Toint. A quadratic programming bibliography. Numerical Analysis Group Internal Report 2000-1, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2000*a*.

N. I. M. Gould and Ph. L. Toint. SQP methods for large-scale nonlinear programming. *In* M. J. D. Powell and S. Scholtes, eds, 'System Modelling and Optimization, Methods, Theory and Applications', pp. 149–178, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000*b*.

N. I. M. Gould and Ph. L. Toint. An iterative active-set method for large-scale quadratic programming. Technical Report in preparation, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2001*a*.

N. I. M. Gould and Ph. L. Toint. Preprocessing for quadratic programming. Technical Report in preparation, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2001*b*.

N. I. M. Gould, M. E. Hribar, and J. Nocedal. On the solution of equality constrained quadratic problems arising in optimization. Technical Report RAL-TR-98-069, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 1998.

N. I. M. Gould, S. Lucidi, M. Roma, and Ph. L. Toint. Solving the trust-region subproblem using the Lanczos method. *SIAM Journal on Optimization*, **9**(2), 504–525, 1999.

N. I. M. Gould, D. Orban, A. Sartenaer, and Ph. L. Toint. Superlinear convergence of primal-dual interior point algorithms for nonlinear programming. Technical Report RAL-TR-2000-014, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2000. To appear in *SIAM Journal on Optimization*.

S. P. Han. Solving quadratic programs with an exact penalty function. *In* O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds, 'Nonlinear Programming, 4', pp. 25–55, Academic Press, London and New York, 1981.

HSL. A collection of Fortran codes for large scale scientific computation, 2000.

C. Keller. *Constraint preconditioning for indefinite linear systems.* D. Phil. thesis, Oxford University, England, 2000.

C. Keller, N. I. M. Gould, and A. J. Wathen. Constraint preconditioning for indefinite linear systems. *SIAM Journal on Matrix Analysis and Applications*, **21**(4), 1300–1317, 2000.

O. L. Mangasarian. Locally unique solutions of quadratic programs, linear and non-linear complementarity problems. *Mathematical Programming*, **19**(2), 200–212, 1980.

K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, **39**(2), 117–129, 1987.

J. Nocedal and S. J. Wright. Quadratic programming. *In* 'Numerical Optimization', Series in Operations Research, chapter 16, pp. 441–488. Springer Verlag, Heidelberg, Berlin, New York, 1999.

P. M. Pardalos and G. Schnitger. Checking local optimality in constrained quadratic programming is NP-hard. *Operations Research Letters*, **7**(1), 33–35, 1988.

B. T. Polyak. The conjugate gradient method in extremal problems. *U.S.S.R. Computational Mathematics and Mathematical Physics*, **9**, 94–112, 1969.

D. C. Sorensen. Updating the symmetric indefinite factorization with applications in a modified Newton method. Technical Report ANL-77-49, Argonne National Laboratory, Illinois, USA, 1977.

T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, **20**(3), 626–637, 1983.

Ph. L. Toint. Towards an efficient sparsity exploiting Newton method for minimization. *In* I. S. Duff, ed., 'Sparse Matrices and Their Uses', pp. 57–88, Academic Press, London, 1981.

R. J. Vanderbei. LOQO: an interior point code for quadratic programming. Technical Report SOR 94–15, Program in Statistics and Operations,Research, Princeton University, New Jersey, USA, 1994.

R. J. Vanderbei and T. J. Carpenter. Symmetrical indefinite systems for interior point methods. *Mathematical Programming*, **58**(1), 1–32, 1993.

S. A. Vavasis. Quadratic programming is in NP. *Information Processing Letters*, **36**(2), 73–77, 1990.

S. A. Vavasis. Convex quadratic programming. *In* 'Nonlinear Optimization: Complexity Issues', pp. 36–75, Oxford University Press, Oxford, England, 1991.

S. Wright and Y. Zhang. A superquadratic infeasible-interior-point method for linear complementarity problems. *Mathematical Programming, Series A*, **73**(3), 269–289, 1996.

Y. Ye. Indefinite quadratic programming. *In* 'Interior-Point Algorithm: Theory and Analysis', chapter 9.4–9.5, pp. 310–331. J. Wiley and Sons, New York, USA, 1997.

E. A. Yildirim and S. J. Wright. Warm-start strategies in interior-point methods for linear programming. Technical Report MCS-P799-0300, Argonne National Laboratory, Illinois, USA, 2000.

Y. Zhang. On the convergence of a class of infeasible interior-point methods for the horizontal linear complementarity problem. *SIAM Journal on Optimization*, **4**(1), 208–227, 1994.