Computing a search direction for large-scale linearly-constrained nonlinear optimization calculations

M. Arioli [1,2], T. F. Chan [3], I. S. Duff [1,4,5], N. I. M. Gould [1,4,5] and J. K. Reid [4,5]

CERFACS Report TR/PA/93/34

August 26, 1993

Dedicated to the memory of Leslie Fox, our friend and teacher

Abstract. We consider the computation of Newton-like search directions that are appropriate when solving large-scale linearly-constrained nonlinear optimization problems. We investigate the use of both direct and iterative methods and consider efficient ways of modifying the Newton equations in order to ensure global convergence of the underlying optimization methods.

[1] Parallel Algorithms Team, CERFACS, 42 Ave. G. Coriolis, 31057 Toulouse Cedex, France

[2] IAN-CNR, c/o Dipartimento di Matematica, 209, via Abbiategrasso 27100 Pavia, Italy

[3] Department of Mathematics, University of California, 405 Hilgard Avenue, Los Angeles, CA 90024-1555, USA

[4] Central Computing Department, Rutherford Appleton Laboratory, Chilton, Oxfordshire, OX11 0QX, England

[5] Current reports available by anonymous ftp from the directory “pub/reports” on camelot.cc.rl.ac.uk (internet 130.246.8.61)

Keywords: Large-scale problems, unconstrained optimization, linearly constrained optimization, direct methods, iterative methods, modified factorizations.

Mathematics Subject Classifications: 65F05, 65F10, 65F15, 65F50, 65K05, 90C30

Contents

Computing a search direction for large-scale linearly-constrained nonlinear optimization calculations

M. Arioli, T. F. Chan, I. S. Duff, N. I. M. Gould and J. K. Reid

August 26, 1993

Abstract

We consider the computation of Newton-like search directions that are appropriate when solving large-scale linearly-constrained nonlinear optimization problems. We investigate the use of both direct and iterative methods and consider efficient ways of modifying the Newton equations in order to ensure global convergence of the underlying optimization methods.

Dedicated to the memory of Leslie Fox, our friend and teacher.

1 Introduction

In this paper, we consider solving the problem

$$\underset{\mathbf{x}\in\Re^n}{\text{minimize}} \quad f(\mathbf{x}) \tag{1.1}$$

subject to a set of m linearly-independent, general, linear equations

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{1.2}$$

Here f is assumed to be twice continuously differentiable and its gradient and Hessian matrix will be denoted by $\mathbf{g}(\mathbf{x}) \stackrel{\text{def}}{=} \nabla_x f(\mathbf{x})$ and $\mathbf{H}(\mathbf{x}) \stackrel{\text{def}}{=} \nabla_{xx} f(\mathbf{x})$ respectively. We shall furthermore assume that $\mathbf{H}(\mathbf{x})$ is available and that we wish to exploit this curvature information. Contrary to popular belief, this last assumption holds for a wide variety of applications.

Problems of the form (1.1)–(1.2) sometimes arise in their own right (see, for instance, the collection by Bongartz *et al.*, 1993), but arise more commonly as subproblems within more general nonlinear programming calculations (see, for example, Murtagh and Saunders, 1978 and Conn *et al.*, 1993). Although these latter subproblems may at first appear to have a different form from (1.1)–(1.2), it is often possible to convert them to such a form so that algorithms discussed in this paper are appropriate. One example is that of wanting to solve (1.1)–(1.2) with the additional restriction that $\mathbf{x}$ satisfies the simple bound constraints

$$\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \tag{1.3}$$

where the inequalities should be understood componentwise and any of the components of $\mathbf{l}$ and $\mathbf{u}$ may be infinite. The problem (1.1)–(1.3) may be solved by a sequential minimization of the *barrier function*

$$\Psi(\mathbf{x},\mathbf{w},\mathbf{s}) = f(\mathbf{x}) + \sum_{i=1}^{n} \phi(x_i, l_i, u_i, w_i^l, s_i^l, w_i^u, s_i^u), \tag{1.4}$$

subject to the constraints (1.2), where the *shifts* $\mathbf{s} = (\mathbf{s}^l, \mathbf{s}^u)$ and *weights* $\mathbf{w} = (\mathbf{w}^l, \mathbf{w}^u)$ are used in the definition of *barrier terms*, ϕ. Each barrier term has the form

$$\phi(x_i, l_i, u_i, w_i^l, s_i^l, w_i^u, s_i^u) = w_i^l \psi(x_i - l_i + s_i^l) + w_i^u \psi(u_i - x_i + s_i^u), \tag{1.5}$$

where $\psi(\alpha)$ is C^2 for all $\alpha > 0$ and has a singularity at the origin — examples are the logarithmic function $\psi(\alpha) = \log(\alpha)$ and the reciprocal function $\psi(\alpha) = \alpha^{-p}$ for any $p > 0$. The shifts are all nonnegative, the weights for infinite bounds are zero while those corresponding to finite bounds are strictly positive. A variety of barrier functions have been proposed and the reader is referred to Fiacco and McCormick (1968), Wright (1992) or Conn *et al.* (1992) for details.

In Section 2 of this paper, we discuss general issues of convergence for schemes for solving (1.1)–(1.2) and lay the foundations for the linear algebraic processes we shall employ; we formally state the aims of the paper in Section 2.3. In Section 3, we consider how convergence may be ensured in the absence of constraints. We consider both direct and iterative methods and suggest some modifications to the standard conjugate gradient iteration to achieve our aims. We generalize these ideas for constrained problems in Section 4. Specific direct methods appropriate for sparse problems are considered in Sections 5 and 6 and the combination of both direct and iterative methods are considered in Section 7. We present some concluding remarks in Section 8.

2 Minimizing the objective function

We consider iterative methods for finding a local solution to the problem (1.1) subject to the linear constraints (1.2). We let $\mathbf{x}^{(k)}$ be the current iterative approximation and consider how an improved iterate $\mathbf{x}^{(k+1)}$ may be found.

2.1 General considerations

We assume that $\mathbf{x}^{(k)}$ satisfies the constraints (1.2). A typical "linesearch" type iteration would

- compute a *search direction* $\mathbf{p}^{(k)}$ for the objective function, which satisfies the constraints
$$\mathbf{A}\mathbf{p}^{(k)} = \mathbf{0}, \tag{2.1}$$
and for which $\mathbf{p}^{(k)T}\mathbf{g}(\mathbf{x}^{(k)})$ is "sufficiently" negative, and
- perform a *linesearch* for the objective function along the search direction to obtain $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}\mathbf{p}^{(k)}$, for some suitable *stepsize* $\alpha^{(k)} > 0$, such that $f(\mathbf{x}^{(k+1)})$ is "sufficiently" smaller than $f(\mathbf{x}^{(k)})$.

The linesearch should ensure that one of the classical sets of "sufficient decrease" conditions is satisfied at $\mathbf{x}^{(k+1)}$ (see, for instance, Dennis and Schnabel, 1983, Section 6.3). Furthermore, whenever possible, advantage should be taken of the "shape" of the linesearch function to derive an efficient algorithm (see, for example, Lasdon *et al.*, 1973 and Murray and Wright, 1992).

If the Hessian matrix of the objective function is positive definite, a search direction close to that given by Newton's method is desirable. However, in general the Hessian may be indefinite and we need to take precautions to ensure that the search direction is a sufficiently good descent direction.

The *(quasi-)Newton search direction*, $\mathbf{p}^{(k)}$, and the corresponding Lagrange multiplier estimates, $\boldsymbol{\lambda}^{(k)}$, satisfy the equations

$$\begin{pmatrix} \mathbf{B}^{(k)} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{p}^{(k)} \\ \boldsymbol{\lambda}^{(k)} \end{pmatrix} = - \begin{pmatrix} \mathbf{g}(\mathbf{x}^{(k)}) \\ \mathbf{0} \end{pmatrix}, \tag{2.2}$$

where $\mathbf{B}^{(k)}$ is a suitable symmetric matrix. In order to provoke rapid asymptotic convergence, it is desirable that $\mathbf{B}^{(k)}$ should be a reasonable approximation to the Hessian.

We say that a matrix $\mathbf{B}$ is *second-order sufficient* if the condition

$$\mathbf{p}^T\mathbf{B}\mathbf{p} \geq \sigma \mathbf{p}^T\mathbf{p} \quad \text{for some constant} \quad \sigma > 0 \quad \text{and all} \quad \mathbf{p} \quad \text{satisfying} \quad \mathbf{A}\mathbf{p} = \mathbf{0} \tag{2.3}$$

is satisfied. We aim to use a second-order sufficient matrix $\mathbf{B}^{(k)}$ as our Hessian approximation in (2.2). For then equations (2.2) are both necessary and sufficient for the solution $\mathbf{p}^{(k)}$ to be the global solution of the (possibly nonconvex) equality constrained quadratic programming problem

$$\underset{\mathbf{p} \in \Re^n}{\text{minimize}} \ \ \tfrac{1}{2}\mathbf{p}^T\mathbf{B}^{(k)}\mathbf{p} + \mathbf{p}^T\mathbf{g}(\mathbf{x}^{(k)}) \quad \text{subject to} \quad \mathbf{A}\mathbf{p} = \mathbf{0} \tag{2.4}$$

(see, for example, Gill *et al.*, 1981).

So long as $\mathbf{B}^{(k)}$ is second-order sufficient and the matrix

$$\mathbf{E}^{(k)} \stackrel{\text{def}}{=} \mathbf{B}^{(k)} - \mathbf{H}(\mathbf{x}^{(k)}) \tag{2.5}$$

is bounded, it is straightforward to show that the gradient of the Lagrangian function, $f(\mathbf{x}^{(k)}) + \boldsymbol{\lambda}^{(k)T}(\mathbf{A}\mathbf{x}^{(k)} - \mathbf{b})$, converges to zero, for any bounded sequence of iterates generated by the search direction obtained from (2.2) and a "sufficient" linesearch. For, picking any full-rank n by $n - m$ matrix $\mathbf{Z}$ such that $\mathbf{A}\mathbf{Z} = \mathbf{0}$ (see, for instance, Gill *et al.*, 1981), Gill and Murray (1974) show that the minimization of (1.1) subject to (1.2) is equivalent to the unconstrained minimization of $f(\mathbf{x}+\mathbf{Z}\mathbf{p}_z)$ with respect to the variables $\mathbf{p}_z$ so long as $\mathbf{x}$ satisfies (1.2). Moreover, equations (2.2) are equivalent to the (quasi-)Newton equations

$$\mathbf{Z}^T\mathbf{B}^{(k)}\mathbf{Z}\mathbf{p}_z^{(k)} = -\mathbf{Z}^T\mathbf{g}(\mathbf{x}^{(k)}). \tag{2.6}$$

We note that $\mathbf{Z}^T\mathbf{B}^{(k)}\mathbf{Z}$ is positive definite whenever $\mathbf{B}^{(k)}$ is second-order sufficient. Typical global convergence results for unconstrained optimization (see, for instance, Dennis and Schnabel, 1983, Theorem 6.3.3), then imply that

$$\lim_{k \to \infty} \mathbf{Z}^T\mathbf{g}(\mathbf{x}^{(k)}) = \mathbf{0} \tag{2.7}$$

provided the condition number of the *reduced Hessian*, $\mathbf{Z}^T\mathbf{B}^{(k)}\mathbf{Z}$, is bounded. This latter condition is satisfied whenever $\mathbf{B}^{(k)}$ is second-order sufficient and $\mathbf{Z}^T\mathbf{E}^{(k)}\mathbf{Z}$ is bounded. As $\mathbf{Z}$ is of full rank, we finally infer from (2.7) that

$$\lim_{k \to \infty} \mathbf{g}(\mathbf{x}^{(k)}) + \mathbf{A}^T\boldsymbol{\lambda}^{(k)} = \mathbf{0} \tag{2.8}$$

for some sequence of Lagrange multipliers $\{\boldsymbol{\lambda}^{(k)}\}$.

2.2 Computing a search direction

The most obvious way of determining a search direction that is close to the Newton direction is to use a matrix factorization to compute the solution to (2.2). Such a factorization should take into account the symmetric but indefinite structure of the coefficient matrix.

A second possibility is to observe that the reduced-variable (quasi-)Newton equations (2.6) provide the global solution to the unconstrained quadratic minimization problem

$$\underset{\mathbf{p}_z \in \Re^{n-m}}{\text{minimize}} \quad q(\mathbf{p}_z) \overset{\text{def}}{=} \tfrac{1}{2}\mathbf{p}_z^T \mathbf{Z}^T \mathbf{B}^{(k)} \mathbf{Z} \mathbf{p}_z + \mathbf{p}_z^T \mathbf{Z}^T \mathbf{g}(\mathbf{x}^{(k)}) \tag{2.9}$$

whenever $\mathbf{B}^{(k)}$ is second-order sufficient. While it is certainly easy to imagine solving (2.6) using a factorization of the reduced Hessian, this may be inappropriate if $n - m$ is large and $\mathbf{Z}^T\mathbf{B}^{(k)}\mathbf{Z}$ dense. In this case, it may still be possible to calculate an approximation to the Newton direction by applying an *iterative* method so long as it is feasible to form matrix-vector products of the form $\mathbf{Z}^T\mathbf{B}^{(k)}\mathbf{Z}\mathbf{v}$. This may well be the case if $\mathbf{Z}$ is carefully chosen to allow rapid computation of the subsidiary products $\mathbf{Z}\mathbf{v}$, $\mathbf{B}^{(k)}(\mathbf{Z}\mathbf{v})$ and $\mathbf{Z}^T(\mathbf{B}^{(k)}\mathbf{Z}\mathbf{v})$. Although an optimally sparse representation of $\mathbf{Z}$ may be computationally expensive (see Coleman and Pothen, 1984), good choices may often be obtained (see, for instance, Murtagh and Saunders, 1978, Coleman and Pothen, 1986 and Stern and Vavasis, 1993).

When considering iterative methods, we choose to restrict our attention to the method of conjugate gradients. The application of the method to linearly constrained problems has been suggested by a number of authors (see, for instance Gill *et al.*, 1981, Section 5.6.2.1). Moreover, as the conjugate-gradient method produces a monotonically decreasing sequence of values of the quadratic objective function, $q(\mathbf{p}_z)$, it is possible to *truncate* the conjugate gradient iteration at a sub-optimal point for (2.9) while still maintaining a fast asymptotic convergence rate (see, Dembo *et al.*, 1982)).

2.3 Aims of the paper

We state our aims as follows. We wish to:

- determine a matrix $\mathbf{B}^{(k)} = \mathbf{H}(\mathbf{x}^{(k)}) + \mathbf{E}^{(k)}$ so that (2.3) is satisfied and such that the perturbation $\mathbf{E}^{(k)} = 0$ whenever $\mathbf{H}(\mathbf{x}^{(k)})$ satisfies (2.3);

- obtain $\mathbf{E}^{(k)}$ without incurring undue overheads above those normally considered acceptable when calculating the search direction;

- ensure that $\|\mathbf{E}^{(k)}\|$ is bounded relative to $\max(\|\mathbf{A}\|, \|\mathbf{H}(\mathbf{x}^{(k)})\|)$ — provided that $\{\mathbf{x}^{(k)}\}$ remains bounded, this will ensure that $\mathbf{B}^{(k)}$ is uniformly bounded;

- use the sparsity and structure of (2.2) to derive a sparse factorization or effective iterative procedure; and

- limit numerical growth to acceptable limits to ensure a stable algorithm.

In this paper, we shall show how, to a large extent, we may achieve these aims.

3 Hessian approximations and unconstrained optimization

In this section, we review and develop methods which have been used successfully in *unconstrained* optimization. We do this for two reasons. Firstly, such methods may be viewed as prototypes for possible extensions to the constrained case. Secondly, as we have already mentioned in Section 2.2, an implicit elimination of the constraints results in an unconstrained problem.

3.1 Direct methods

If we wish our linearly-constrained optimization algorithm to inherit the fast asymptotic convergence rate of Newton's method, we might try to ensure that $\mathbf{B}^{(k)}$ converges to the true Hessian matrix. In particular, we may choose $\mathbf{B}^{(k)}$ to be $\mathbf{H}(x^{(k)})$ whenever the latter is second-order sufficient.

We recall the precedent set from research in unconstrained optimization by Greenstadt (1967), Gill and Murray (1974) and Schnabel and Eskow (1991) amongst others, in which a modification to the exact second derivative matrix in a Newton method is only made when the exact derivatives are insufficiently positive definite. When there are no constraints present, (2.3) is equivalent to requiring that $\mathbf{B}$ be *sufficiently positive definite*, that is, that

$$\mathbf{p}^T\mathbf{B}\mathbf{p} \geq \sigma \mathbf{p}^T\mathbf{p} \quad \text{for some constant} \quad \sigma > 0 \quad \text{and all} \quad \mathbf{p}. \tag{3.1}$$

In particular, Gill and Murray (1974) and Schnabel and Eskow (1991) suggest *modified-Cholesky factorization* methods which decide whether, and by how much, the diagonals of the Hessian matrix need to be modified as the Cholesky factorization of the matrix proceeds. The factors are then used to solve the resulting modified Newton equations. Both pairs of authors are very careful to ensure that the modifications made are bounded and that no modification ensues when the Hessian is sufficiently positive definite. Extensions to large-scale unconstrained and bound constrained optimization, using sparse factorizations, have been proposed by Gill *et al.* (1992), Conn *et al.* (1991) and Schlick (1992).

We should also mention the philosophically-different but mechanically-similar class of ℓ_2 trust-region methods (see, e.g., Hebden, 1973, Moré, 1978, Sorensen, 1980 and Gay, 1981). Here perturbations of the form $\mathbf{E}^{(k)} = \mu^{(k)}\mathbf{I}$ may be made to $\mathbf{H}(\mathbf{x}^{(k)})$ for some appropriate scalar $\mu^{(k)}$ in order to make $\mathbf{B}^{(k)}$ positive semi-definite. We note the difference of approach between these methods and the modified Cholesky methods, in that perturbations to the Hessian may only be made in certain low-rank subspaces with the modification methods while any perturbation in an ℓ_2 trust-region method produces a rank n change to $\mathbf{H}(\mathbf{x}^{(k)})$.

3.2 Iterative methods

If we intend to use a conjugate-gradient method to (approximately) solve (2.9), a different matrix-modification strategy is needed. Let $\mathbf{B}_z \stackrel{\text{def}}{=} \mathbf{Z}^T\mathbf{B}^{(k)}\mathbf{Z}$ and $\mathbf{g}_z \stackrel{\text{def}}{=} \mathbf{Z}^T\mathbf{g}(\mathbf{x}^{(k)})$. Then we wish to (approximately) solve the *unconstrained* problem,

$$\underset{\mathbf{p}_z \in \Re^{n-m}}{\text{minimize}} \ \tfrac{1}{2}\mathbf{p}_z^T\mathbf{B}_z\mathbf{p}_z + \mathbf{p}_z^T\mathbf{g}_z, \tag{3.2}$$

using the method of conjugate gradients, starting with the estimate $\mathbf{p}_z = 0$.

If $\mathbf{B}_z$ is positive definite, the method of conjugate gradients may be described as follows. Let $\mathbf{p}_z^{(0)} = \mathbf{0}$, $\mathbf{g}_z^{(0)} = \mathbf{g}_z$, $\mathbf{s}_z^{(0)} = -\mathbf{g}_z^{(0)}$ and $\mathbf{B}_z^{(0)} = \mathbf{B}_z$. For $l = 0, 1, \cdots$ until convergence, perform the iteration,

$$\begin{aligned} \alpha^{(l)} &= \|\mathbf{g}_z^{(l)}\|_2^2 / \mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)} & (3.3) \\ \mathbf{p}_z^{(l+1)} &= \mathbf{p}_z^{(l)} + \alpha^{(l)}\mathbf{s}_z^{(l)} & (3.4) \end{aligned}$$

$$\begin{aligned} \mathbf{g}_z^{(l+1)} &= \mathbf{g}_z^{(l)} + \alpha^{(l)} \mathbf{B}_z^{(l)} \mathbf{s}_z^{(l)} & (3.5) \\ \beta^{(l+1)} &= \|\mathbf{g}_z^{(l+1)}\|_2^2 / \|\mathbf{g}_z^{(l)}\|_2^2 & (3.6) \\ \mathbf{s}_z^{(l+1)} &= -\mathbf{g}_z^{(l+1)} + \beta^{(l+1)} \mathbf{s}_z^{(l)} & (3.7) \\ \mathbf{B}_z^{(l+1)} &= \mathbf{B}_z^{(l)} & (3.8) \end{aligned}$$

(see Gill *et al.*, 1981, Section 4.8.3). However, as we do not know in advance whether $\mathbf{B}_z$ is positive definite, we must be prepared to allow modifications to $\mathbf{B}_z$ if necessary. We consider two alternatives.

3.2.1 Methods using estimates of the leftmost eigenvalue

Firstly, we could compute a good approximation $\tilde{\pi}_{\min}$ to the leftmost (most negative or smallest if all positive) eigenvalue, $\pi_{\min}$. If the approximation is sufficiently positive, we use $\mathbf{B}_z$ unchanged. Otherwise, before the iteration commences, we replace $\mathbf{B}_z$ by $\mathbf{B}_z - (\tilde{\pi}_{\min} - \sigma)\mathbf{I}$, where σ is chosen to ensure that $\mathbf{B}^{(k)}$ is second-order sufficient. The Lanczos algorithm (see Lanczos, 1950 or Parlett, 1980, Chapter 13) may be used to estimate $\pi_{\min}$. We observe that this method has the flavour of an ℓ_2 trust-region method (see Section 3.1).

This approach may prove to be expensive unless care is taken. In particular, it is unnecessary to estimate $\pi_{\min}$ unless a modification is necessary. Fortunately, we can use the well-known relationship between the conjugate gradient and Lanczos methods (see, for instance, Golub and Loan, 1989, Section 9.3.1) to avoid excess computation unless it is really justified. We start by assuming that $\mathbf{B}_z$ is positive definite and perform the conjugate gradient iteration (3.3)–(3.8). We also form the tridiagonal "Lanczos" matrix (see, Golub and Loan, 1989, Section 10.2.6) from the quantities involved in the conjugate gradient iteration and estimate the leftmost eigenvalue of this matrix; efficient methods for doing this have been proposed by Paige and Saunders (1975) and Parlett and Reid (1981). If the tridiagonal matrix reveals that $\mathbf{B}_z$ is insufficiently positive definite, the iteration is continued in "Lanczos" mode. That is, information from the conjugate gradient iteration is assimilated in a true "Lanczos" process which then aims to calculate an accurate approximation to the leftmost eigenvalue, $\tilde{\pi}_{\min}$, as a precursor to restarting the conjugate gradient iteration.

Obtaining the leftmost eigenvalue in this way may still prove to be too expensive. One possibility is to take the leftmost eigenvalue of the "Lanczos" matrix at the time that it is discovered that $\mathbf{B}_z$ is insufficiently positive definite as an estimate of the leftmost eigenvalue of $\mathbf{B}_z$, or perhaps to continue for a few more "Lanczos" iterations before such an estimate is made. Another possibility is to obtain an upper bound on the absolute value of the eigenvalue and to shift $\mathbf{B}_z$ by at least this amount. One such bound is given by the inequality

$$|\pi_{\min}| \leq \|\mathbf{B}_z\|_2 \leq \|\mathbf{B}_z\|_{est} \overset{\text{def}}{=} \min(\|\mathbf{B}_z\|_\infty, \|\mathbf{B}_z\|_F), \tag{3.9}$$

see Golub and Loan (1989, Section 2.3). If $\mathbf{B}_z$ were explicitly available, it would be trivial to calculate $\|\mathbf{B}_z\|_{est}$. However, in our case, $\mathbf{B}_z$ is generally only available as the product of matrices and the calculation of $\|\mathbf{B}_z\|_{est}$ is out of the question. We may nonetheless estimate $\|\mathbf{B}_z\|_{est}$ using the algorithms of Hager (1984) or Higham (1988) to estimate $\|\mathbf{B}_z\|_\infty$. We must be cautious here as such algorithms may underestimate the infinity norm.

Unless we compute an accurate approximation of $\pi_{\min}$, or a lower bound for it, we may have to perform a number of conjugate gradient cycles. Each cycle ends when a negative eigenvalue of the Lanzcos matrix is found. Then $\mathbf{B}_z$ is replaced by $\mathbf{B}_z - (\tilde{\pi}_{\min} - \sigma)\mathbf{I}$, for some appropriate $\sigma > 0$, and the next conjugate gradient cycle started.

3.2.2 Methods which modify the matrix as the iteration proceeds

Secondly, we could modify $\mathbf{B}_z$ as the conjugate gradient iteration proceeds. We have to be careful here as general perturbations to $\mathbf{B}_z$ will destroy the conjugacy properties of the algorithm and affect the convergence of the iteration. Fortunately, the conjugate gradient algorithm provides all the information we need to be able to build a second-order sufficient $\mathbf{B}_z$ if the original was insufficient while maintaining the required conjugacy. The key is to note that each iteration (3.3)–(3.8) computes the curvature $\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}$ along the current search direction $\mathbf{s}_z^{(l)}$. If this is negative, or small relative to the size of $\mathbf{s}_z^{(l)}$, the original $\mathbf{B}_z$ is indefinite and must be modified.

We will make use of the identities

$$\begin{aligned} \mathbf{s}_z^{(l+1)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(j)} &= 0 \quad \text{for} \quad j = 0, \cdots, l, & (3.10)\\ \mathbf{g}_z^{(l+1)T}\mathbf{g}_z^{(j)} &= 0 \quad \text{for} \quad j = 0, \cdots, l, & (3.11)\\ \mathbf{g}_z^{(l+1)T}\mathbf{s}_z^{(j)} &= 0 \quad \text{for} \quad j = 0, \cdots, l, \quad \text{and} & (3.12)\\ \mathbf{s}_z^{(l+1)T}\mathbf{g}_z^{(l+1)} &= -\mathbf{g}_z^{(l+1)T}\mathbf{g}_z^{(l+1)} & (3.13) \end{aligned}$$

(see, for instance, Fletcher, 1987, Theorem 4.1.1, and Gill *et al.*, 1981, Section 4.8.3.1). We first note that it is trivial to embed the identity

$$\|\mathbf{s}_z^{(l+1)}\|_2^2 = \|\mathbf{g}_z^{(l+1)}\|_2^2 + \beta^{(l+1)2}\|\mathbf{s}_z^{(l)}\|_2^2 \tag{3.14}$$

within the conjugate gradient iteration (3.3)–(3.8) and, as we have already computed all quantities on the right-hand side of (3.14), we may recur $\|\mathbf{s}_z^{(l)}\|_2^2$ at almost no cost. The identity (3.14) follows from the definition of $\mathbf{s}_z^{(l+1)}$ in (3.7) and the linesearch condition (3.12). Thus we may compute the Rayleigh quotient

$$\rho_{\text{old}}^{(l+1)} = \frac{\mathbf{s}_z^{(l+1)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l+1)}}{\|\mathbf{s}_z^{(l+1)}\|^2}. \tag{3.15}$$

If $\rho_{\text{old}}^{(l+1)} < \bar{\sigma}$ for some small $\bar{\sigma} \stackrel{\text{def}}{=} (n+1)\sigma > 0$ (and where σ is the required tolerence in (2.3)), $\mathbf{B}_z^{(l)}$ is judged to be insufficiently positive definite and we replace (3.8) by

$$\mathbf{B}_z^{(l+1)} = \mathbf{B}_z^{(l)} + \theta^{(l+1)}\mathbf{v}_z^{(l+1)}\mathbf{v}_z^{(l+1)T}, \tag{3.16}$$

where $\theta^{(l+1)}$ is given by

$$\theta^{(l+1)} = \frac{\sigma^{(l+1)}\|\mathbf{s}_z^{(l+1)}\|_2^2 - \mathbf{s}_z^{(l+1)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l+1)}}{(\mathbf{v}_z^{(l+1)T}\mathbf{s}_z^{(l+1)})^2}, \tag{3.17}$$

the scalar $\sigma^{(l+1)} \geq \sigma$ and $\mathbf{v}_z^{(l+1)}$ is any vector for which

$$\mathbf{v}_z^{(l+1)T}\mathbf{s}_z^{(j)} = 0 \quad \text{for} \quad j = 0, \cdots, l. \tag{3.18}$$

The definition (3.17) ensures that

$$\rho_{\text{new}}^{(l+1)} = \frac{\mathbf{s}_z^{(l+1)T}\mathbf{B}_z^{(l+1)}\mathbf{s}_z^{(l+1)}}{\|\mathbf{s}_z^{(l+1)}\|_2^2} = \sigma^{(l+1)} \geq \bar{\sigma}. \tag{3.19}$$

Moreover, as (3.16) and (3.18) ensure that

$$\mathbf{B}_z^{(l+1)}\mathbf{s}_z^{(j)} = (\mathbf{B}_z^{(l)} + \theta^{(l+1)}\mathbf{v}_z^{(l+1)}\mathbf{v}_z^{(l+1)T})\mathbf{s}_z^{(j)} = \mathbf{B}_z^{(l)}\mathbf{s}_z^{(j)} \tag{3.20}$$

for $j = 0, \cdots, l$, the conjugate gradient method generates the same sequence of l iterates if we replace $\mathbf{B}_z^{(l)}$ by $\mathbf{B}_z^{(l+1)}$.

Our strategy may be summarized as follows. Implement the conjugate gradient iteration (3.3)–(3.7) and (3.14). At each iteration, evaluate the Rayleigh quotient (3.15). Then set

$$\mathbf{B}_z^{(l+1)} = \begin{cases} \mathbf{B}_z^{(l)} + \theta^{(l+1)} \mathbf{v}_z^{(l+1)} \mathbf{v}_z^{(l+1)T} & \text{if } \rho_{\text{old}}^{(l+1)} < \bar{\sigma} \\ \mathbf{B}_z^{(l)} & \text{otherwise} \end{cases}, \tag{3.21}$$

and continue the iteration. Notice that each time $\mathbf{B}_z^{(l)}$ is updated, we need to store an extra vector $\sqrt{\theta^{(l+1)}}\mathbf{v}_z^{(l+1)}$ and thus it would seem that we can only continue this strategy so long as we have sufficient room. We also note that, since the cost of a matrix-vector product increases as we add extra rank-one terms, we may also wish to end the iteration on efficiency grounds if many modifications are required. Actually, in at least one important case, we will shortly show that it is theoretically possible to avoid the storage of the vectors $\sqrt{\theta^{(l+1)}}\mathbf{v}_z^{(l+1)}$.

A nice consequence of the above modification is that

$$\mathbf{s}_z^{(i)T} \mathbf{B}_z^{(l)} \mathbf{s}_z^{(i)} \geq \mathbf{s}_z^{(i)T} \mathbf{B}_z^{(i)} \mathbf{s}_z^{(i)} \geq \bar{\sigma} \|\mathbf{s}_z^{(i)}\|^2 \tag{3.22}$$

for all $i \leq l$. Hence, if l iterations are performed, and $\mathbf{s}_z = \sum_{i=0}^{l} \upsilon_i \mathbf{s}_z^{(i)}$ is any vector lying in the (Krylov) space searched by the conjugate gradient iteration, (3.10), (3.20) and (3.22) imply

$$\begin{array}{rcl} \mathbf{s}_z^T \mathbf{B}_z^{(l)} \mathbf{s}_z = & \sum_{i=0}^{l} \sum_{j=0}^{l} \upsilon_i \upsilon_j \mathbf{s}_z^{(i)T} \mathbf{B}_z^{(l)} \mathbf{s}_z^{(j)} & = \sum_{i=0}^{l} \upsilon_i^2 \mathbf{s}_z^{(i)T} \mathbf{B}_z^{(l)} \mathbf{s}_z^{(i)} \\ \geq & \bar{\sigma} \sum_{i=0}^{l} \upsilon_i^2 \mathbf{s}_z^{(i)T} \mathbf{s}_z^{(i)} & = \bar{\sigma} \sum_{i=0}^{l} \|\upsilon_i \mathbf{s}_z^{(i)}\|_2^2 \\ \geq & \dfrac{\bar{\sigma}}{l+1} \left\| \sum_{i=0}^{l} \upsilon_i \mathbf{s}_z^{(i)} \right\|_2^2 & = \dfrac{\bar{\sigma}}{l+1} \|\mathbf{s}_z\|_2^2 \geq \sigma \|\mathbf{s}_z\|_2^2. \end{array} \tag{3.23}$$

Thus the Rayleigh quotient, and hence the smallest eigenvalue, of $\mathbf{B}_z^{(l)}$ restricted to the (Krylov) subspace investigated is bounded away from zero.

We would also like to say something about the size of any perturbations made. If the Hessian is modified on the l-th conjugate gradient iteration, (3.16) and (3.17) combine to show the perturbation satisfies the bound

$$\|\theta^{(l+1)} \mathbf{v}_z^{(l+1)} \mathbf{v}_z^{(l+1)T})\| = (\sigma^{(l+1)} \|\mathbf{s}_z^{(l+1)}\|_2^2 - \mathbf{s}_z^{(l+1)T} \mathbf{B}_z^{(l)} \mathbf{s}_z^{(l+1)}) \frac{\|\mathbf{v}_z^{(l+1)}\|_2^2}{(\mathbf{v}_z^{(l+1)T} \mathbf{s}_z^{(l+1)})^2}. \tag{3.24}$$

If the initial $\mathbf{B}_z$ has smallest eigenvalue π_{min}, we may use the boundedness property of the Rayleigh quotient (see Parlett, 1980, Section 1.5) to replace (3.24) by

$$\|\theta^{(l+1)} \mathbf{v}_z^{(l+1)} \mathbf{v}_z^{(l+1)T})\| \leq (\sigma^{(l+1)} - \pi_{\text{min}}) \frac{\|\mathbf{s}_z^{(l+1)}\|_2^2 \|\mathbf{v}_z^{(l+1)}\|_2^2}{(\mathbf{v}_z^{(l+1)T} \mathbf{s}_z^{(l+1)})^2}. \tag{3.25}$$

Thus, in order to bound the size of the perturbation, we need to choose $\mathbf{v}_z^{(l+1)}$ so that the ratio

$$\omega(\mathbf{v}_z^{(l+1)}) \overset{\text{def}}{=} \frac{\|\mathbf{s}_z^{(l+1)}\|_2^2 \|\mathbf{v}_z^{(l+1)}\|_2^2}{(\mathbf{v}_z^{(l+1)T} \mathbf{s}_z^{(l+1)})^2} \tag{3.26}$$

is bounded.

There are two "obvious" — obvious in the sense that, following (3.10) and (3.12), they automatically satisfy (3.18) — choices for $\mathbf{v}_z^{(l+1)}$, namely

$$\mathbf{v}_z^{(l+1)} = \mathbf{B}_z^{(l)} s_z^{(l+1)} \quad \text{and} \tag{3.27}$$

$$\mathbf{v}_z^{(l+1)} = \mathbf{g}_z^{(l+1)}. \tag{3.28}$$

The first choice is unlikely to be useful as there is then no guarantee that the denominator of (3.26) is nonzero. Fortunately, as we shall now show, the second choice (3.28) provides a satisfactory perturbation.

When $\mathbf{v}_z^{(l+1)}$ is given by (3.28), (3.13) implies that

$$\omega(\mathbf{g}_z^{(l+1)}) = \|\mathbf{s}_z^{(l+1)}\|_2^2 / \|\mathbf{g}_z^{(l+1)}\|_2^2. \tag{3.29}$$

Using (3.14) and the definition (3.6), it is straightforward to show that

$$\omega(\mathbf{g}_z^{(l+1)}) = 1 + \beta^{(l+1)} \omega(\mathbf{g}_z^{(l)}) \geq 1. \tag{3.30}$$

But, as

$$\mathbf{g}_z^{(l+1)} = \mathbf{g}_z^{(l)} + \frac{\|\mathbf{g}_z^{(l)}\|_2^2}{\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}} \mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}, \tag{3.31}$$

from (3.3) and (3.5), the identity (3.11) implies that

$$0 = \mathbf{g}_z^{(l+1)T}\mathbf{g}_z^{(l)} = \|\mathbf{g}_z^{(l)}\|_2^2 + \frac{\|\mathbf{g}_z^{(l)}\|_2^2}{\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}} \mathbf{g}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)} \tag{3.32}$$

and hence that

$$\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)} = -\mathbf{g}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}. \tag{3.33}$$

But then, the relationships (3.31) and (3.33) give that

$$\begin{aligned} \|\mathbf{g}_z^{(l+1)}\|_2^2 = & \|\mathbf{g}_z^{(l)}\|_2^2 + 2\frac{\|\mathbf{g}_z^{(l)}\|_2^2}{\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}} \mathbf{g}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)} + \frac{\|\mathbf{g}_z^{(l)}\|_2^4}{(\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)})^2} \|\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}\|_2^2 \\ = & \|\mathbf{g}_z^{(l)}\|_2^2 - 2\frac{\|\mathbf{g}_z^{(l)}\|_2^2}{\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}} \mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)} + \frac{\|\mathbf{g}_z^{(l)}\|_2^4}{(\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)})^2} \|\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}\|_2^2 \\ = & \|\mathbf{g}_z^{(l)}\|_2^2 \left(\frac{\|\mathbf{g}_z^{(l)}\|_2^2}{(\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)})^2} \|\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}\|_2^2 - 1 \right). \end{aligned} \tag{3.34}$$

Combining (3.6), (3.19), (3.29) and (3.34), we obtain the bound

$$\begin{aligned} \beta^{(l+1)} = & \frac{\|\mathbf{g}_z^{(l)}\|_2^2}{(\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)})^2} \|\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}\|_2^2 - 1 \\ \leq & \frac{\|\mathbf{g}_z^{(l)}\|_2^2}{\|\mathbf{s}_z^{(l)}\|_2^2} \left(\frac{\|\mathbf{s}_z^{(l)}\|_2^2}{(\mathbf{s}_z^{(l)T}\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)})} \right)^2 \|\mathbf{B}_z^{(l)}\|_2^2 - 1 = \frac{\|\mathbf{B}_z^{(l)}\|_2^2}{\rho_{\text{new}}^{(l)2}\omega(\mathbf{g}_z^{(l)})} - 1. \end{aligned} \tag{3.35}$$

Finally, (3.30) and (3.35) combine to give

$$\omega(\mathbf{g}_z^{(l+1)}) = 1 + \frac{\|\mathbf{B}_z^{(l)}\|_2^2}{\rho_{\text{new}}^{(l)2}} - \omega(\mathbf{g}_z^{(l)}) \leq \frac{\|\mathbf{B}_z^{(l)}\|_2^2}{\rho_{\text{new}}^{(l)2}} \tag{3.36}$$

and thus

$$\|\theta^{(l+1)}\mathbf{g}_z^{(l+1)}\mathbf{g}_z^{(l+1)T})\| \leq (\sigma^{(l+1)} - \pi_{\min}) \frac{\|\mathbf{B}_z^{(l)}\|_2^2}{\rho_{\text{new}}^{(l)2}}. \tag{3.37}$$

The inequality (3.37) thus furnishes us with an (albeit large) upper bound on the possible perturbation to $\mathbf{B}_z$ in terms of the initial data $\mathbf{B}_z$ and the tolerance $\rho_{\text{new}}^{(l)} \geq \bar{\sigma}$.

The scalar $\sigma^{(l+1)}$ should be chosen so that (3.37) is not too large, but may also be used to control the size of the step

$$\alpha^{(l+1)}\|\mathbf{s}_z^{(l+1)}\|_2 = \frac{\|\mathbf{g}_z^{(l+1)}\|_2^2}{\mathbf{s}_z^{(l+1)T}\mathbf{B}_z^{(l+1)}\mathbf{s}_z^{(l+1)}} \|\mathbf{s}_z^{(l+1)}\|_2 = \frac{\omega(\mathbf{g}_z^{(l+1)})\|\mathbf{s}_z^{(l+1)}\|_2}{\sigma^{(l+1)}} \tag{3.38}$$

in (3.4). Notice how the curvature, $\rho_{\mathrm{new}}^{(l)}$, from iteration l may affect the size of any perturbation (3.37) in iteration $l+1$. In particular, if modifications are made on two successive iterations, a small choice for $\sigma^{(l)}$ can lead to large pertubations in (3.37) since, when a perturbation is made, $\rho_{\mathrm{new}}^{(l)} = \sigma^{(l)}$. Thus it may be worth picking $\sigma^{(l)} = O(\|\mathbf{B}_z^{(l)}\|)$; preliminary numerical experience indicates that this is so (see Section 3.2.4).

Perhaps more seriously, if $\rho_{\mathrm{new}}^{(l)}$ is small, but no perturbation is made during iteration l, while $\rho_{\mathrm{old}}^{(l+1)} < \bar{\sigma}$, the perturbation at iteration $l+1$ is likely to be large. In order to prevent this, it may be worth restarting the computation from the last iteration, say iteration k, for which $\rho_{\mathrm{new}}^{(k)}$ is large, but replacing $\bar{\sigma}$ by $\max(\bar{\sigma}, \rho_{\mathrm{old}}^{(k+1)})$. Then, a small perturbation will be made at iteration $k+1$ — as the test at iteration $l+1$ has already determined that $\mathbf{B}_z$ is insufficiently positive definite, we may as well perturb it in a controlled fashion.

We should also remark that it is possible to choose

$$\mathbf{v}_z^{(l+1)} = \mathbf{g}_z^{(l+1)} + \xi^{(l)}\mathbf{B}_z^{(l)} s_z^{(l+1)} \tag{3.39}$$

for some scalar $\xi^{(l)}$. A simple, but tedious, calculation reveals that the perturbation bound (3.25) is minimized for all such $\mathbf{v}_z^{(l+1)}$ when $\xi^{(l)} = 0$. It is not known if there is a choice for $\mathbf{v}_z^{(l+1)}$ which gives a better perturbation bound than (3.28).

Finally, we mentioned earlier that although it would appear that we need to store all the update vectors $\sqrt{\theta^{(l+1)}}\mathbf{g}_z^{(l+1)}$, this is not the case. Recall, we only need to access $\mathbf{B}_z$ and the update vectors in order to form the matrix-vector product $\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)}$. From (3.21), we may write

$$\mathbf{B}_z^{(l)} = \mathbf{B}_z + \sum_{i=1}^{l} \theta^{(i)}\mathbf{g}_z^{(i)}\mathbf{g}_z^{(i)T} \tag{3.40}$$

where $\theta^{(i)} = 0$ on iterations for which $\rho_{\mathrm{old}}^{(i)} \geq \bar{\sigma}$.

Thus, we need to be able to calculate

$$\mathbf{B}_z^{(l)}\mathbf{s}_z^{(l)} = \mathbf{B}_z\mathbf{s}_z^{(l)} + \sum_{i=1}^{l} \theta^{(i)}\mathbf{g}_z^{(i)}\mathbf{g}_z^{(i)T}\mathbf{s}_z^{(l)} = \mathbf{B}_z\mathbf{s}_z^{(l)} + \mathbf{w}_z^{(l)}, \tag{3.41}$$

where

$$\mathbf{w}_z^{(l)} \stackrel{\mathrm{def}}{=} \sum_{i=1}^{l} \theta^{(i)}(\mathbf{g}_z^{(i)T}\mathbf{s}_z^{(l)})\mathbf{g}_z^{(i)}. \tag{3.42}$$

But, we may combine (3.7), (3.11) , (3.13) and (3.42) to obtain the relationship

$$\begin{aligned}
\mathbf{w}_z^{(l+1)} &= \textstyle\sum_{i=1}^{l+1} \theta^{(i)}(\mathbf{g}_z^{(i)T}\mathbf{s}_z^{(l+1)})\mathbf{g}_z^{(i)} \\
&= \theta^{(l+1)}(\mathbf{g}_z^{(l+1)T}\mathbf{s}_z^{(l+1)})\mathbf{g}_z^{(l+1)} + \textstyle\sum_{i=1}^{l} \theta^{(i)}(\mathbf{g}_z^{(i)T}\mathbf{s}_z^{(l+1)})\mathbf{g}_z^{(i)} \\
&= \theta^{(l+1)}(\mathbf{g}_z^{(l+1)T}\mathbf{s}_z^{(l+1)})\mathbf{g}_z^{(l+1)} + \textstyle\sum_{i=1}^{l} \theta^{(i)}(-\mathbf{g}_z^{(i)T}\mathbf{g}_z^{(l+1)} + \beta^{(l+1)}\mathbf{g}_z^{(i)T}\mathbf{s}_z^{(l)})\mathbf{g}_z^{(i)} \\
&= \theta^{(l+1)}(\mathbf{g}_z^{(l+1)T}\mathbf{s}_z^{(l+1)})\mathbf{g}_z^{(l+1)} + \beta^{(l+1)}\textstyle\sum_{i=1}^{l} \theta^{(i)}(\mathbf{g}_z^{(i)T}\mathbf{s}_z^{(l)})\mathbf{g}_z^{(i)} \\
&= -\theta^{(l+1)}\|\mathbf{g}_z^{(l+1)}\|_2^2\mathbf{g}_z^{(l+1)} + \beta^{(l+1)}\mathbf{w}_z^{(l)}.
\end{aligned} \tag{3.43}$$

Thus, rather than storing all the vectors $\sqrt{\theta^{(l)}}\mathbf{g}_z^{(l)}$, we merely need to recur the vector $\mathbf{w}_z^{(l)}$ from iteration to iteration, using the relationship (3.43). Alternatively, if we define $\mathbf{u}_z^{(l)}$ by the relationship

$$\mathbf{w}_z^{(l)} = -\|\mathbf{g}_z^{(l)}\|_2^2\mathbf{u}_z^{(l)}, \tag{3.44}$$

(3.43) reveals that

$$\mathbf{u}_z^{(l+1)} = \mathbf{u}_z^{(l)} + \theta^{(l+1)}\mathbf{g}_z^{(l+1)} \tag{3.45}$$

and we may thus calculate (3.41) using (3.44) and the recurrence (3.45). We should caution the reader that significant numerical cancellation may occur in the recurrences (3.43) and (3.45) and numerical experience suggests that the relationships should only be used with caution.

3.2.3 Other methods

The only other similar method we are are aware of is that by Nash (1984). Here, the equivalence between the conjugate gradient and Lanczos methods is exploited in a method which calculates a LDL^T factorization of the tridiagonal Lanczos matrix. Whenever this matrix is indefinite, it is modified in a numerically stable way and Nash shows that any perturbation so induced is bounded. The main drawback appears to be that the method needs all of the Lanczos vectors to calculate the solution to (3.2) and it not clear that economies may be made to prevent this.

3.2.4 Numerical experience

We now investigate the performance of the two distinct approaches described in Sections 3.2.1 and 3.2.2.

We constructed 51 problems of the form (3.2), where $n - m$ is 100, each $\mathbf{B}_z$ is of the form $\mathbf{Q}^T\mathbf{D}\mathbf{Q}$, $\mathbf{Q}$ is a random Housholder matrix, — the entries in the Housholder vector are uniformly distributed between -1 and 1 — and $\mathbf{g}_z$ is a vector with entries 100. The diagonal matrix $\mathbf{D}$ for the i-th problem has $2i$ negative eigenvalues; the eigenvalues themselves are randomly selected from a uniform distribution across the interval $[-100\min(1, i/25), 100\min(1, 2 - i/25)]$. We report results for tests on 11 of these 51 problems. Similar behaviour was observed on the remaining 40 problems and, for brevity, we omit the details from this paper. The salient features of the 11 considered problems are given in Table 3.1.

Table 3.1: Characteristics of test problems

problem	-ve evals	spectrum	problem	-ve evals	spectrum
0	0	$[0.0007, 100]$	30	60	$[-100, -0.02] \cup [1, 80]$
5	10	$[-20, -0.4] \cup [0.004, 100]$	35	70	$[-100, -0.4] \cup [0.2, 60]$
10	20	$[-40, -1] \cup [1, 100]$	40	80	$[-100, -0.05] \cup [2, 40]$
15	30	$[-60, -0.3] \cup [1, 100]$	45	90	$[-100, -1] \cup [1, 20]$
20	40	$[-80, -0.02] \cup [1, 100]$	50	100	$[-100, -0.7]$
25	50	$[-100, -0.5] \cup [0.6, 100]$			

In Table 3.2, we illustrate the behaviour of the methods outlined in Section 3.2.1. We first report, in the column headed "Lan", the number of iterations (matrix-vector products) required by the Harwell Subroutine Library Lanczos code `EA15` (see Parlett and Reid, 1981) to locate the leftmost eigenvalue of $\mathbf{B}_z$. We note that the number of products required invariably lies between $\frac{1}{2}(n - m)$ and $n - m$ and indicates that the first method suggested in Section 3.2.1 is likely to be impractical.

In the other columns of Table 3.2, we report on the method suggested at the end of Section 3.2.1 in which we perform a number of conjugate gradient cycles. Each cycle ends when either the norm of the gradient (residual) is smaller than 10^{-6}, in which case the method terminates, or when negative curvature is encountered, at which point an estimate $\tilde{\pi}_{\min}$ of the leftmost eigenvalue, $\pi_{\min}$, of $\mathbf{B}_z$ is obtained using the inequality (3.9) and the Harwell Subroutine Library `MC41` that uses the method of Hager (1984) to estimate the one

		$\sigma = 10^3$				$\sigma = 10^2$				$\sigma = 10$				$\sigma = 1$			
pr	Lan	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	ne	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	ne	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	ne	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	ne	it
0	74	-	5	0	73	-	5	0	73	-	5	0	73	-	5	0	73
5	57	3	0	1	14	2	0	1	18	2	0	1	19	2	0	1	19
10	56	3	0	1	13	2	0	1	16	2	0	1	18	2	0	1	18
15	73	3	0	1	16	2	0	1	19	2	0	1	20	2	0	1	21
20	60	3	0	1	14	2	0	1	17	2	0	1	18	2	0	1	18
25	75	3	0	1	14	2	0	1	18	2	0	1	19	2	0	1	20
30	93	3	0	1	13	2	0	1	16	2	0	1	18	2	0	1	18
35	89	3	0	1	15	2	0	1	18	2	0	1	19	2	0	1	19
40	81	3	0	1	12	2	0	1	17	2	1	1	19	2	1	1	20
45	86	3	0	1	12	2	0	1	16	2	1	1	18	2	1	1	19
50	66	3	0	1	12	2	0	1	15	2	1	1	17	2	1	1	17

Table 3.2: Methods which estimate the leftmost eigenvalue.
Key: *pr* = problem, *Lan* = number of matrix-vector products required by `EA15` to find $\pi_{\min}$, $||\mathbf{E}||$ = exponent of norm of perturbation, $||\mathbf{p}||$ = exponent of norm of solution estimate, *ne* = number of times $\pi_{\min}$ was estimated, *it* = number of matrix-vector products required by conjugate gradients.

or infinity norm of a matrix. Then $\mathbf{B}_z$ is replaced by $\mathbf{B}_z - (\hat{\pi}_{\min} - \sigma)\mathbf{I}$, for some appropriate $\sigma > 0$, and the next conjugate gradient cycle is started. We report the exponent (logarithm to the base 10) of the norms of the perturbation matrix $\mathbf{E}$ and the solution estimate $\mathbf{p}_z$ — we recall that the main purpose of perturbing $\mathbf{B}_z$ is to guarantee a reasonable bound on $\mathbf{p}_z$ — together with the number of times that $\pi_{\min}$ was estimated and the total number of matrix-vector products required, in the columns headed "$||\mathbf{E}||$", "$||\mathbf{p}||$", "ne" and "it", respectively. We give these statistics for a range of values of σ and note that results identical to those reported for $\sigma = 1$ were also obtained for smaller values of σ. We note that the solution is obtained, in all cases, in an extremely modest number of matrix-vector products. Unfortunately, the one-norm estimator typically overestimated the two-norm by a factor of two, and consequently the perturbation used was larger than strictly necessary. This has the pleasant property that the resulting perturbed matrix $\mathbf{B}_z$ is well conditioned but tends to bias the solution $\mathbf{p}_z$ towards the steepest descent direction, $-\mathbf{g}_z$. We also note that, in all examples, a single estimate of the leftmost eigenvalue proved adequate.

In Tables 3.3 and 3.4, we illustrate the behaviour of the methods outlined in Section 3.2.2. Once again we consider a range of values of the smallest allowable perturbed eigenvalue, $\sigma^{(l)}$, and report the exponent of the norms of the perturbation matrix $\mathbf{E}$ and the solution estimate $\mathbf{p}_z$ together with the number of rank-one modifications made to the initial $\mathbf{B}_z$ and the total number of matrix-vector products required, in the columns headed "$||\mathbf{E}||$", "$||\mathbf{p}||$", "nm" and "it", respectively. We impose an upper bound of $2(n-m)$ iterations per test and regard the method to have failed if this bound is reached.

As we have already suggested in Section 3.2.2 the choice of $\sigma^{(l)}$ significantly influences the size of the perturbation made and, as predicted, a choice $\sigma^{(l)} = O(||\mathbf{B}_z^{(l)}||)$ gives the smallest perturbations. Nonetheless, the perturbations reported are quite large in some cases and invariably lead to larger $\mathbf{p}_z$ than those obtained from the methods of Section 3.2.1. Moreover, the condition numbers for the final matrices $\mathbf{B}_z$ obtained using the methods which made rank-one changes were often significantly larger than those for the full-rank change methods which results in a higher average number of matrix-vector products for the rank-one methods.

In Table 3.4, we illustrate the behaviour of the methods outlined in Section 3.2.2. in

	$\sigma^{(l)} = 10^3$				$\sigma^{(l)} = 10^2$				$\sigma^{(l)} = 10$				$\sigma^{(l)} = 1$			
pr	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	nm	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	nm	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	nm	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	nm	it
0	-	5	0	73	-	5	0	73	-	5	0	73	-	5	0	73
5	5	2	4	15	4	2	7	28	5	2	12	82	4	4	14	115
10	6	1	6	14	6	1	9	23	7	3	17	65	5	4	29	182
15	4	1	6	14	3	2	11	26	6	3	35	102	10	5	43	200
20	6	2	5	12	6	2	9	24	5	3	60	136	9	4	47	200
25	7	3	4	9	6	3	16	25	7	3	61	147	6	4	62	200
30	3	0	7	7	2	1	22	22	8	3	68	134	10	4	69	200
35	3	0	7	7	2	1	20	20	5	3	65	124	8	4	68	200
40	3	0	7	7	2	1	19	19	5	2	75	137	6	4	63	200
45	3	0	6	6	2	1	17	17	6	2	68	113	8	4	66	200
50	3	0	6	6	2	1	15	15	6	2	75	131	6	4	77	200

Table 3.3: Methods which modify the matrix as the iteration proceeds.
Key: *pr* = problem, $\|\mathbf{E}\|$ = exponent of norm of perturbation, $\|\mathbf{p}\|$ = exponent of norm of solution estimate, *nm* = number of rank-one modifications made, *it* = number of matrix-vector products required.

	$\sigma^{(l)} = 10^3$				$\sigma^{(l)} = 10^2$				$\sigma^{(l)} = 10$				$\sigma^{(l)} = 1$			
pr	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	nm	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	nm	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	nm	it	$\|\mathbf{E}\|$	$\|\mathbf{p}\|$	nm	it
0	-	5	0	73	-	5	0	73	-	5	0	73	-	5	0	73
5	5	2	4	13	4	2	7	23	4	2	12	54	4	3	19	200
10	3	1	6	12	6	1	9	20	4	3	26	58	3	4	34	200
15	4	1	6	12	4	2	11	22	4	3	48	68	19	7	36	200
20	5	2	5	10	5	2	9	18	7	3	79	200	4	8	50	200
25	7	3	4	7	6	3	16	18	21	4	85	117	21	12	68	200
30	3	0	7	7	2	1	22	22	5	3	78	200	4	7	57	200
35	3	0	7	7	2	1	20	20	18	5	95	200	21	8	59	94
40	3	0	7	7	2	1	19	19	2	2	100	102	3	6	55	200
45	3	0	6	6	2	1	17	17	2	2	100	101	20	11	63	200
50	3	0	6	6	2	1	15	15	2	2	85	85	3	8	63	200

Table 3.4: Methods which modify the matrix using the recurrence.
Key: *pr* = problem, $\|\mathbf{E}\|$ = exponent of norm of perturbation, $\|\mathbf{p}\|$ = exponent of norm of solution estimate, *nm* = number of rank-one modifications made, *it* = number of matrix-vector products required.

which the matrix-vector products are formed from the relationship (3.41) and the recurrence (3.43). We observe extremely large perturbations for $\sigma^{(l)} = 10$ and $\sigma^{(l)} = 1$. On closer examination, we found that this was due to the instability of the recurrence and would thus rule this version out.

		Lanczos estimate						
name	n	its	fe	mv	nm	time	**EA15** nm	**EA15** time
BROYDN7D	500	34	71	1041	25	7.34	2965	14.28
BROYDN7D	1000	35	79	1657	26	21.62	5408	52.88
CHNROSNB	50	46	66	598	2	0.51	58	0.04
DIXMAANA	300	6	14	12	1	0.16	4	0.02
DIXMAANE	300	7	14	144	1	0.63	154	0.63
DIXMAANI	300	7	14	971	1	3.92	269	1.07
ERRINROS	50	69	127	842	6	0.73	247	0.16
GENROSE	100	80	162	3479	64	3.44	3030	2.99
GENROSE	500	432	861	22178	414	121.18	21743	106.09
MANCINO	100	12	24	17	1	52.36	26	1.02
SPMSQRT	100	20	38	539	14	1.28	663	1.55
SPMSQRT	1000	59	123	7450	53	175.18	9664	216.90

		ℓ_∞ estimate					low rank				
name	n	its	fe	mv	nm	time	its	fe	mv	nm	time
BROYDN7D	500	251	544	2374	242	27.92	40	76	463	29	4.78
BROYDN7D	1000	452	985	4218	443	101.89	63	121	777	51	15.67
CHNROSNB	50	993	1996	11634	968	8.95	45	65	543	2	0.50
DIXMAANA	300	5	12	13	1	0.12	6	13	10	1	0.14
DIXMAANE	300	7	14	120	1	0.52	15	30	263	5	1.19
DIXMAANI	300	7	14	770	1	3.11	7	15	912	1	3.68
ERRINROS	50	62	102	587	4	0.57	63	107	602	5	0.60
GENROSE	100	>1001	-	-	-	> 17.62	71	133	1318	41	1.48
GENROSE	500	>1001	-	-	-	> 107.57	274	548	5125	170	32.87
MANCINO	100	12	24	22	1	52.57	12	20	18	1	49.60
SPMSQRT	100	12	24	165	5	0.45	13	26	161	4	0.44
SPMSQRT	1000	398	864	42282	110	994.29	36	72	636	25	17.64

Table 3.5: Embedding the methods within an unconstrained optimization code.
Key: *its* = number of iterations performed, *fv* = numbers of function evaluations required, *mv* = number of matrix-vector products required (not including EA15 if used), *nm* = number of iterations on which a modification was necessary, *time* = cpu time (IBM RS/6000 320H seconds) required (not including EA15 if used).

So far it is unclear whether it is desirable to allow larger perturbations in a relatively small subspace, as constructed by the methods of Section 3.2.2, or to settle for smaller, but full-rank, perturbations as suggested in Section 3.2.1. In Table 3.5, we illustrate the consequences of embedding the matrix modification techniques within an unconstrained optimization code. At each iteration, a search direction is obtained by solving a subproblem of the form (3.2) using a (non-preconditioned) conjugate gradient method — of course, as the problem is unconstrained, $m = 0$ and $\mathbf{Z} = \mathbf{I}$. The conjugate gradient iteration is terminated when the norm of the gradient of the quadratic model is smaller than $||\mathbf{g}_z|| \min(0.1, ||\mathbf{g}_z||^{0.5})$, with a safety-net termination if more than $n + 10$ conjugate gradient steps have been taken. The matrix $\mathbf{B}_z$ is modified whenever it is found to be insufficiently positive definite. We consider modifying the matrix by:

- the method of Section 3.2.1 in which the leftmost eigenvalue is calculated by the Lanczos process and $\sigma = 10^{-4}$;

- the method from the same section in which the leftmost eigenvalue is estimated using an approximation to the one-norm with the same value of σ; and
- the method of Section 3.2.2 in which all the required rank-one terms are stored and $\sigma^{(l)} = 1.0$.

We stress that the only difference between the three illustrated algorithms is in the approximate solution of (2.9) and the consequent modification (when necessary) of $\mathbf{B}_z$. We use the test problem set given by Bongartz *et al.* (1993), selecting all unconstrained problem classes with 50 or more unknowns and reporting on those problems for which negative curvature was encountered. For each problem and each method, we give the number of iterations performed (its), the numbers of function evaluations and matrix-vector products required (fe and mv), the number of iterations on which a modification was necessary (nm) and the cpu time in seconds on an IBM RS/6000 320H workstation required to solve the problem (time). For the method in which the leftmost eigenvalue is calculated by the Lanczos process, we report separately the total number of matrix-vector products and the time required for by the calls to the Lanczos algorithm, `EA15` (`EA15` nm and `EA15` time, respectively).

Note that, although the full-rank modifications appear to be efficient in solving the linear system (Tables 3.2 to 3.4), the number of matrix-vector products (and hence, normally, the time) required by the nonlinear solver greatly increases when such a technique is used. This is undoubtedly because of the bias towards steepest descent and indicates that, although the perturbations from the low-rank method may be larger, the modifications for such methods do not tend to upset the positive-definite subspaces of $\mathbf{B}$ and retain the Newton-like convergence properties in these subspaces. Closer scrutiny of the runs indicates that the low-rank method appears to move more rapidly into regions where there are relatively few negative eigenvalues than the other methods. It is also clear from Table 3.5 that if a full-rank method is used, it is important to get good estimates of the leftmost eigenvalues - the method in which estimates are used is significantly less efficient than the Lanczos-based method where exact values are calculated. We also observe that, generally, the run-times of the Lanczos-based method, even discounting the `EA15` times is inferior to that of the low-rank method. Thus, any of the other methods suggested in Section 3.2.1, which attempt to estimate the smallest eigenvalue more efficiently than `EA15`, are unlikely to be competative with the low-rank technique. Thus, we feel that the low-rank modification methods hold considerable promise for the iterative solution of (large-scale) nonlinear optimization problems.

4 Hessian approximations and constrained optimization

As far as we are aware, the only serious attempt to generalize the methods of Section (3) to solve general, large-scale, linearly-constrained optimization problems is that by Forsgren and Murray (1993). All other methods we are aware of are either only really appropriate for small-scale calculations as they disregard problem structure (see Fletcher, 1987, Section 11.1, or Gill *et al.*, 1981, Section 5.1, for example) or implicitly assume that $n - m$ is sufficiently small that coping with dense matrices of order $n - m$ is practicable (see, for instance, Murtagh and Saunders, 1978).

4.1 Matrix factorizations

Firstly, note that the coefficient matrix,

$$\mathbf{K}^{(k)} \overset{\text{def}}{=} \begin{pmatrix} \mathbf{B}^{(k)} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{pmatrix}, \tag{4.1}$$

of (2.2) is inevitably indefinite — it must have at least m positive and m negative eigenvalues. If $\mathbf{B}^{(k)}$ is a second-order sufficient matrix, Gould (1985) showed that $\mathbf{K}^{(k)}$ must have precisely m negative and n positive eigenvalues. Thus any matrix factorization of (4.1) must be capable of handling indefinite matrices. Moreover, in order to be efficient, one would normally try to exploit the symmetry of $\mathbf{K}^{(k)}$ in the factorization. The natural generalization of the Cholesky (or more precisely $\mathbf{LDL}^T$) factorization in the symmetric, indefinite case is that first proposed by Bunch and Parlett (1971) and later improved by Bunch and Kaufman (1977) and Fletcher (1976) in the dense case and Duff *et al.* (1979) and Duff and Reid (1983) in the sparse case. Here a symmetric matrix $\mathbf{K}$ is decomposed as

$$\mathbf{K} = \mathbf{PLDL}^T\mathbf{P}^T, \tag{4.2}$$

where $\mathbf{P}$ is a permutation matrix, $\mathbf{L}$ unit lower triangular and $\mathbf{D}$ block diagonal, with blocks of size at most two. Each diagonal block corresponds to a pivoting operation. We shall refer to the blocks as 1 by 1 and 2 by 2 pivots. Notice that the inertia of $\mathbf{K}$ — the numbers of positive, negative and zero eigenvalues of $\mathbf{K}$ — is trivially obtained by summing the inertia of the pivots.

As we are particularly concerned with the large-scale case in this paper, it is the Duff-Reid variant that is of special interest. We note that the permutation matrices are used extensively in the factorization of sparse matrices to keep the fill-in at an acceptable level. Unfortunately, the Harwell Subroutine Library (1990) implementation, MA27, (Duff and Reid, 1982) of the Duff-Reid variant sometimes proves inefficient when applied to matrices of the form (4.1) as the analysis phase treats the whole diagonal of $\mathbf{K}$ as entries. Thus a good predicted ordering supplied by the analyse phase is often replaced, for stability reasons, by a less satisfactory ordering when the factorization is performed, resulting in considerable extra work and fill-in. Ways of avoiding these difficulties, and of taking further advantage of the zero block in $\mathbf{K}$, have been suggested by Duff *et al.* (1991), and form the basis for a recent Harwell Subroutine Library code MA47 (Duff and Reid, 1993).

If $\mathbf{B}^{(k)}$ is known *a priori* to be second-order sufficient, as for instance would be the case if $f(\mathbf{x})$ were convex, we wholeheartedly recommend the use of MA27 or MA47 to solve (2.2). When there is a chance that $\mathbf{B}^{(k)}$ may not be second-order sufficient, alternatives to solving (2.2) must be sought.

4.2 Forsgren and Murray's sufficient pivoting conditions

We say that the first n rows of $\mathbf{K}^{(k)}$ are $\mathbf{B}^{(k)}$-rows, and the remaining m rows are $\mathbf{A}$-rows. Forsgren and Murray (1993) show that, if the pivots are restricted to be of certain types until all of the $\mathbf{A}$-rows of $\mathbf{K}^{(k)}$ have been eliminated, the remaining un-eliminated (Schur-complement) matrix, $\mathbf{S}^{(k)}$, is sufficiently positive definite if and only if $\mathbf{B}^{(k)}$ is second-order sufficient.

Until all $\mathbf{A}$-rows of $\mathbf{K}^{(k)}$ have been exhausted, Forsgren and Murray only allow the following types of pivots:

$\mathbf{b}_+$ pivots: strictly positive 1 by 1 pivots occurring in $\mathbf{B}^{(k)}$-rows of $\mathbf{K}^{(k)}$.

$\mathbf{a}_-$ pivots: strictly negative 1 by 1 pivots occurring in $\mathbf{A}$-rows of $\mathbf{K}^{(k)}$.

ba pivots: 2 by 2 pivots with a strictly negative determinant, one of whose rows is an $\mathbf{B}^{(k)}$-row and the other of whose rows is an $\mathbf{A}$-row of $\mathbf{K}^{(k)}$.

They further restrict the pivot so that its determinant is greater than a small positive constant so as to limit any growth in $\mathbf{S}^{(k)}$. The motivation behind this choice of pivot is simply that if i $\mathbf{A}$-rows have been eliminated, the factorized matrix has exactly i negative eigenvalues. Thus, when all $\mathbf{A}$-rows have been eliminated, the factorized matrix has precisely m negative eigenvalues and hence any further negative eigenvalues in $\mathbf{S}^{(k)}$ can only occur because $\mathbf{B}^{(k)}$ is not second-order sufficient.

Once $\mathbf{S}^{(k)}$ has been determined, Forsgren and Murray form a partial Cholesky factorization of it, stopping if a pivot is insufficiently positive. If the factorization runs to completion, $\mathbf{B}^{(k)}$ must be second-order sufficient. The (quasi-)Newton equations (2.2) are subsequently solved using the factorization. If an insufficiently positive pivot is encountered, a search arc is obtained as a nonlinear combination of a search direction derived from the partial factorization and a direction of negative curvature from the remaining unfactorized part.

An obvious variation is, instead, to form a modified Cholesky factorization of $\mathbf{S}^{(k)}$. If no modification is performed, the true Hessian $\mathbf{H}(x^{(k)})$ must be second-order sufficient. Otherwise, a suitable perturbation $\mathbf{E}^{(k)}$ will have been produced. In either case, the Newton equations (2.2) are solved using the complete factorization.

The main difficulty with Forsgren and Murray's approach is that any restriction on the pivot order can disqualify potentially advantageous sparsity orderings. While it is always possible to choose a pivot according to the Forsgren-Murray recipe, the available choices may all lead to considerable fill-in. Nonetheless, we shall consider a number of variations of this scheme.

5 Methods using ba pivots

In this section, we consider a scheme which uses a restricted version of Forsgren and Murray's (1993) pivoting rules. Specifically, we consider what happens if we choose the first m pivots to be *ba* pivots.

In what follows, we shall drop the superscript (k) unless it is absolutely needed for clarity.

5.1 Algebraic considerations

Let us first suppose that we have chosen a pivot sequence so that the first m pivots are *ba* pivots. Algebraically, this is equivalent to constructing a permutation $\mathbf{P}$ for which

$$\mathbf{P}^T\mathbf{K}\mathbf{P} = \left(\begin{array}{cc|c} \mathbf{B}_{11} & \mathbf{A}_1^T & \mathbf{B}_{12} \\ \mathbf{A}_1 & \mathbf{0} & \mathbf{A}_2 \\ \hline \mathbf{B}_{12}^T & \mathbf{A}_2^T & \mathbf{B}_{22} \end{array}\right), \tag{5.1}$$

where $\mathbf{B}_{11}$ and $\mathbf{A}_1$ are both square matrices of order m, $\mathbf{A}_1$ is non-singular and $\mathbf{P}$ may be partitioned as

$$\mathbf{P} = \begin{pmatrix} \mathbf{P}_{11} & \mathbf{0} & \mathbf{P}_{13} \\ \mathbf{P}_{21} & \mathbf{0} & \mathbf{P}_{23} \\ \mathbf{0} & \mathbf{P}_{32} & \mathbf{0} \end{pmatrix}. \tag{5.2}$$

(The actual pivot sequence would interlace the i-th and $m+i$-th rows of $\mathbf{P}$ for $i = 1, \cdots, m$). This permutation implies a corresponding partitioning

$$\begin{pmatrix} \mathbf{p} \\ \boldsymbol{\lambda} \end{pmatrix} = \mathbf{P} \begin{pmatrix} \mathbf{p}_1 \\ \boldsymbol{\lambda} \\ \hline \mathbf{p}_2 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} \mathbf{g} \\ \mathbf{0} \end{pmatrix} = \mathbf{P} \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{0} \\ \hline \mathbf{g}_2 \end{pmatrix} \tag{5.3}$$

of the solution and right-hand side of (2.2). Thus, to solve (2.2), we obtain auxiliary variables $\mathbf{q}_1$ and $\boldsymbol{\lambda}_2$ from the equation

$$\begin{pmatrix} \mathbf{B}_{11} & \mathbf{A}_1^T \\ \mathbf{A}_1 & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{q}_1 \\ \boldsymbol{\lambda}_2 \end{pmatrix} = - \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{0} \end{pmatrix}, \tag{5.4}$$

and subsequently solve the equations

$$\left(\mathbf{B}_{22} - (\mathbf{B}_{12}^T \;\; \mathbf{A}_2^T) \begin{pmatrix} \mathbf{B}_{11} & \mathbf{A}_1^T \\ \mathbf{A}_1 & \mathbf{0} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{B}_{12} \\ \mathbf{A}_2 \end{pmatrix} \right) \mathbf{p}_2 = -\mathbf{g}_2 - (\mathbf{B}_{12}^T \;\; \mathbf{A}_2^T) \begin{pmatrix} \mathbf{q}_1 \\ \boldsymbol{\lambda}_2 \end{pmatrix} \tag{5.5}$$

and

$$\begin{pmatrix} \mathbf{B}_{11} & \mathbf{A}_1^T \\ \mathbf{A}_1 & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{p}_1 \\ \boldsymbol{\lambda} \end{pmatrix} = - \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{0} \end{pmatrix} - \begin{pmatrix} \mathbf{B}_{12} \\ \mathbf{A}_2 \end{pmatrix} \mathbf{p}_2 \tag{5.6}$$

and therefore require decompositions of the matrix

$$\begin{pmatrix} \mathbf{B}_{11} & \mathbf{A}_1^T \\ \mathbf{A}_1 & \mathbf{0} \end{pmatrix}, \tag{5.7}$$

and its Schur-complement,

$$\mathbf{S} = \mathbf{B}_{22} - (\mathbf{B}_{12}^T \;\; \mathbf{A}_2^T) \begin{pmatrix} \mathbf{B}_{11} & \mathbf{A}_1^T \\ \mathbf{A}_1 & \mathbf{0} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{B}_{12} \\ \mathbf{A}_2 \end{pmatrix}, \tag{5.8}$$

in $\mathbf{P}^T\mathbf{K}\mathbf{P}$. These decompositions would be performed implicitly if we factorized $\mathbf{K}$ as (4.2) with the pivot sequence defined by $\mathbf{P}$, but a number of salient points may be made if we consider (5.7) and (5.8) explicitly.

5.2 An equivalence to reduced-variable methods

Since $\mathbf{A}_1$ is non-singular, we have an explicit form for the inverse of (5.7),

$$\begin{pmatrix} \mathbf{B}_{11} & \mathbf{A}_1^T \\ \mathbf{A}_1 & \mathbf{0} \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{0} & \mathbf{A}_1^{-1} \\ \mathbf{A}_1^{-T} & -\mathbf{A}_1^{-T}\mathbf{B}_{11}\mathbf{A}_1^{-1} \end{pmatrix}. \tag{5.9}$$

Gill *et al.* (1990, Theorem 7.2) observe that this enables us to rewrite (5.8) as

$$\mathbf{S} = \mathbf{Z}^T\mathbf{B}\mathbf{Z}, \quad \text{where} \quad \mathbf{Z} = \begin{pmatrix} \mathbf{P}_{11}^T & \mathbf{P}_{21}^T \\ \mathbf{P}_{13}^T & \mathbf{P}_{23}^T \end{pmatrix} \begin{pmatrix} \mathbf{I} \\ -\mathbf{A}_1^{-1}\mathbf{A}_2 \end{pmatrix}, \tag{5.10}$$

and the matrix $\mathbf{Z}$ satisfies $\mathbf{A}\mathbf{Z} = \mathbf{0}$. The equations (5.4)–(5.6) are then equivalent to the reduced (quasi-)Newton equations (2.6), together with the extra equation

$$\mathbf{A}_1^T\boldsymbol{\lambda} = -\mathbf{g}_1 - \mathbf{B}_{11}\mathbf{p}_1 - \mathbf{B}_{12}\mathbf{p}_2 \tag{5.11}$$

for the Lagrange multipliers. Thus forcing *ba* pivots until we have exhausted all the $\mathbf{A}$-rows of $\mathbf{K}$ is equivalent to finding a representation of the null-space of $\mathbf{A}$ and using the reduced-variable method described in Section 2.1. In particular, the Schur-complement matrix, $\mathbf{S}$, is a reduced Hessian matrix.

5.3 Boundedness of perturbations

Because of the relationship (5.10), the norm of $\mathbf{S}$ satisfies

$$\|\mathbf{S}\| \leq (1 + \|\mathbf{A}_1^{-1}\mathbf{A}_2\|)^2 \|\mathbf{B}\| \tag{5.12}$$

and thus element growth may be controlled by using an appropriate threshold-pivoting tolerance when factorizing $\mathbf{A}_1$. Therefore, if one of the modified Cholesky methods cited in Section 3.1 is subsequently employed to factorize $\mathbf{S}$, the perturbation matrix $\mathbf{E}$ will remain bounded.

5.4 Appropriate orderings

As the same permutation may be used at *every* iteration of the nonlinear programming algorithm, it may be worth investing considerable effort in producing a good ordering. As we are primarily concerned with large problems, it is essential to try to ensure that the chosen permutation $\mathbf{P}$ introduces as little fill-in within the Schur complement and the factorization of $\mathbf{A}_1$ as possible. Notice that each *ba* pivot requires that we select a row and column of $\mathbf{A}$ and that the selected column of $\mathbf{A}$ defines the row of $\mathbf{B}$ used.

Without loss of generality, we describe how the first *ba* pivot is determined. The same procedure may then be applied recursively to the Schur-complement of this pivot in $\mathbf{K}$ to determine *ba* pivots $2, \cdots, m$. Suppose that we consider the permutation

$$\mathbf{P}_1^T \mathbf{K} \mathbf{P}_1 = \left(\begin{array}{cc|cc} \beta & \alpha & \mathbf{b}_c^T & \mathbf{a}_c^T \\ \alpha & 0 & \mathbf{a}_r^T & \mathbf{0} \\ \hline \mathbf{b}_c & \mathbf{a}_r & \mathbf{B}_R & \mathbf{A}_R^T \\ \mathbf{a}_c & \mathbf{0} & \mathbf{A}_R & \mathbf{0} \end{array} \right), \tag{5.13}$$

where $\alpha \neq 0$ and β are scalars, $\mathbf{b}_c$ and $\mathbf{a}_r$ are $n-1$-vectors, $\mathbf{a}_c$ is an $m-1$-vector, and $\mathbf{B}_R$ and $\mathbf{A}_R$ are $n-1$ by $n-1$ and $m-1$ by $n-1$ matrices, respectively. Then, a simple calculation reveals that the Schur-complement of the *ba* pivot in $\mathbf{P}_1^T \mathbf{K} \mathbf{P}_1$ is

$$\mathbf{S}_1 = \begin{pmatrix} \mathbf{B}_R & \mathbf{A}_R^T \\ \mathbf{A}_R & \mathbf{0} \end{pmatrix} - \frac{1}{\alpha^2} \left[\alpha \begin{pmatrix} \mathbf{b}_c \\ \mathbf{a}_c \end{pmatrix} (\mathbf{a}_r^T \;\; \mathbf{0}) + \alpha \begin{pmatrix} \mathbf{a}_r \\ \mathbf{0} \end{pmatrix} (\mathbf{b}_c^T \;\; \mathbf{a}_c^T) - \beta \begin{pmatrix} \mathbf{a}_r \\ \mathbf{0} \end{pmatrix} (\mathbf{a}_r^T \;\; \mathbf{0}) \right]. \tag{5.14}$$

Notice that no fill-in occurs in the zero, bottom block of $\mathbf{S}_1$. We now follow Markowitz (1957) by picking the *ba* pivot to modify the least number of coefficients in the remaining $n+m-2$ order block of $\mathbf{P}_1^T \mathbf{K} \mathbf{P}_1$ as the Schur complement is formed. Thus we aim to minimize the number of nonzeros, n_s in the matrix

$$\alpha \begin{pmatrix} \mathbf{b}_c \\ \mathbf{a}_c \end{pmatrix} (\mathbf{a}_r^T \;\; \mathbf{0}) + \alpha \begin{pmatrix} \mathbf{a}_r \\ \mathbf{0} \end{pmatrix} (\mathbf{b}_c^T \;\; \mathbf{a}_c^T) - \beta \begin{pmatrix} \mathbf{a}_r \\ \mathbf{0} \end{pmatrix} (\mathbf{a}_r^T \;\; \mathbf{0}). \tag{5.15}$$

There are two cases to consider.

Following Duff *et al.* (1991), we call a *ba* pivot a *tile* pivot if $\beta \neq 0$ and an *oxo* pivot when $\beta = 0$. We let $n_z(\mathbf{v})$ denote the number of nonzeros in the vector $\mathbf{v}$ and $n_o(\mathbf{v}, \mathbf{w})$ give the number of overlaps (the number of indices i for which both v_i and w_i are nonzero) between the vectors $\mathbf{v}$ and $\mathbf{w}$.

A simple computation reveals that, if we choose an oxo pivot, the number of nonzeros in the matrix (5.15) is

$$n_s = 2n_z(\mathbf{a}_r)[n_z(\mathbf{a}_c) + n_z(\mathbf{b}_c)] - n_o(\mathbf{a}_r, \mathbf{b}_c)^2, \tag{5.16}$$

while a tile pivot yields

$$n_s \leq 2n_z(\mathbf{a}_r)[n_z(\mathbf{a}_c) + n_z(\mathbf{b}_c)] - n_o(\mathbf{a}_r, \mathbf{b}_c)^2 + [n_z(\mathbf{a}_r) - n_o(\mathbf{a}_r, \mathbf{b}_c)]^2 \tag{5.17}$$

(the inequality in (5.17) accounts for the possibility of exact cancellation between the terms in (5.15)). Thus, if $\mathbf{A}$ has rows $\mathbf{a}_{r_i}$, $i = 1, \cdots, m$ and columns $\mathbf{a}_{c_j}$, $j = 1, \cdots, n$ and $\mathbf{B}$ has columns $\mathbf{b}_{c_j}$, $j = 1, \cdots, n$, one possibility is to pick the *ba* pivot for which

$$|a_{i,j}| \geq v \max_{1 \leq l \leq n} |a_{i,l}| \tag{5.18}$$

for some pivot tolerance $0 < v \leq 1$ and for which

$$\sigma^e_{i,j} = \begin{cases} 2(n_z(\mathbf{a}_{r_i}) - 1)(n_z(\mathbf{a}_{c_j}) + n_z(\mathbf{b}_{c_j}) - 1) - n_o(\mathbf{a}_{r_i}, \mathbf{b}_{c_i})^2 & \text{when} \quad b_{j,j} = 0 \\ 2(n_z(\mathbf{a}_{r_i}) - 1)(n_z(\mathbf{a}_{c_j}) + n_z(\mathbf{b}_{c_j}) - 2) - & \\ (n_o(\mathbf{a}_{r_i}, \mathbf{b}_{c_i}) - 1)^2 + (n_z(\mathbf{a}_{r_i}) - n_o(\mathbf{a}_{r_i}, \mathbf{h}_{c_j}) - 2)^2 & \text{when} \quad b_{j,j} \neq 0 \end{cases} \tag{5.19}$$

is smallest. However, as computing $n_o(\mathbf{a}_{r_i}, \mathbf{b}_{c_j})$ may prove to be unacceptably expensive, we follow Duff *et al.* (1991) and overestimate (5.16) and (5.17) by assuming that, except in the pivot rows, there are no overlaps and thus pick the pivot for which

$$\sigma^a_{i,j} = \begin{cases} 2(n_z(\mathbf{a}_{r_i}) - 1)(n_z(\mathbf{a}_{c_j}) + n_z(\mathbf{b}_{c_j}) - 1) & \text{when} \quad b_{j,j} = 0 \\ 2(n_z(\mathbf{a}_{r_i}) - 1)(n_z(\mathbf{a}_{c_j}) + n_z(\mathbf{b}_{c_j}) - 2) + (n_z(\mathbf{a}_{r_i}) - 1)^2 & \text{when} \quad b_{j,j} \neq 0 \end{cases} \tag{5.20}$$

is smallest. It is relatively straightforward to compute and update the nonzero counts required to use (5.20). Indeed, as $n_z(\mathbf{a}_{r_i})$ and $n_z(\mathbf{a}_{c_j}) + n_z(\mathbf{b}_{c_j})$ are, respectively, the row and column counts for the matrix

$$\begin{pmatrix} \mathbf{B}^{(k)} \\ \mathbf{A} \end{pmatrix}, \tag{5.21}$$

the schemes described by Duff *et al.* (1986, Section 9.2) are appropriate.

Although this section has been concerned with direct methods of solution, we observed in subsection 5.2 that the use of m *ba* pivots is equivalent to calculating a change of basis so that a reduced variable method can be used. If, after performing such pivots, our aim is subsequently to use an *iterative* method to (approximately) solve the resulting unconstrained problem (2.9), a different strategy for selecting the *ba* pivots is appropriate. For then, our aim should be to obtain as sparse a factorization of $\mathbf{A}_1$ as possible — so that the matrix-vector product $\mathbf{Z}^T\mathbf{B}^{(k)}\mathbf{Z}$ can be formed as economically as possible (see Section 2.2) — and the interaction between $\mathbf{A}$ and $\mathbf{B}^{(k)}$ is mostly irrelevant. A good ordering in this case may be obtained by minimizing the Markowitz count

$$\sigma^o_{i,j} = (n_z(\mathbf{a}_{r_i}) - 1)(n_z(\mathbf{a}_{c_j}) - 1) \tag{5.22}$$

over all indices $i = 1, \cdots, m$ and $j = 1, \cdots, n$ which satisfy (5.18). Alternatively, one might select the column index j to minimize $n_z(\mathbf{a}_{c_j})$ and then pick any i which satisfies (5.18).

5.5 Dense rows

The main disadvantage of the schemes described in this section is that, by restricting the pivot order, the fill-in within the Schur complement may prove unacceptable. This will be the case if $\mathbf{A}$ contains dense rows since then the Schur complement will almost certainly be completely dense.

A possible way of alleviating this difficulty is to allow all of the pivot types suggested by Forsgren and Murray (1993) (see Section 4.2). A drawback is that, by allowing b_+

and a_- pivots, we may introduce fill-ins into the "zero" block of (4.1) and, thereafter the Markowitz costs (5.19) and (5.20) are inappropriate. Appropriate Markowitz costs in this case have been suggested by Duff *et al.* (1991). Preference should still be given to pivots involving **A**-rows if at all possible.

However, even if we allow all types of pivots suggested by Forsgren and Murray, there are still cases where the Schur complement becomes unacceptably dense. In the next two sections, we consider methods which aim to avoid such difficulties.

6 Delayed pivoting Methods

Suppose that **A** contains m_d rows with a large number of nonzeros and that the remaining $m_e \equiv m - m_d$ rows are sparse. Then it is likely that if any of the dense **A**-rows is included in an early pivot, the remaining Schur complement will substantially fill in. It therefore makes sense to avoid pivoting on these rows until the closing stages of the elimination when the Schur complement may be treated as a dense matrix. However, the Forsgren-Murray pivoting rules may conspire to make this impossible.

Let us suppose that we have eliminated the sparse m_e rows of **A** and n_e rows of $\mathbf{B}^{(k)}$ using Forsgren and Murray's pivoting rules and that the remaining Schur complement **S** is relatively sparse excepting the m_d **A**-rows. Thus, we may no longer use ba or a_- pivots and are restricted to b_+ ones. Let us further assume that there are no longer any acceptable b_+ pivots, possibly because all the diagonals in $\mathbf{B}^{(k)}$-rows are negative or more likely because the remaining b_+ pivot will cause unacceptable fill-in when eliminated. At this stage, we are no longer able to take Forsgren-Murray pivots.

Now assume that a b_- pivot — a negative 1 by 1 pivot occurring in a $\mathbf{B}^{(k)}$-row of **S** — would be acceptable from the point of view of fill-in and stability. We aim to investigate the consequences of using this pivot. Remember that our goal is only to modify $\mathbf{B}^{(k)}$ if it fails to be second-order sufficient.

6.1 The condemned submatrix

We pick a nonsingular, square submatrix, the *condemned* submatrix **C**, of **S** which contains all the **A**-rows and perhaps some of the $\mathbf{B}^{(k)}$-rows (but not the b_- pivot row) of **S** and has precisely m_d negative eigenvalues. The condemned submatrix will be eliminated last of all and thus any $\mathbf{B}^{(k)}$-rows included in **C** will not be generally available as pivots. The aim is that, when only the rows which make up **C** remain to be eliminated, **S** will have precisely m_d negative eigenvalues and hence **K** will have exactly m negative eigenvalues.

The Schur complement **S** has at least m_d negative eigenvalues. A suitable **C** may be obtained, for instance, by picking $\max(n_e - m_e, 0)$ a_- followed by $\max(m - n_e, 0)$ ba pivots. A factorization of the condemned submatrix should be obtained. As **C** is invariably dense, a full matrix factorization is appropriate and, because we may subsequently need to modify the factors, we recommend the QR or LQ factorizations. This of course limits the scope of the current proposal because of the size of **C** which can be accommodated. We note that the dimension of **C** cannot exceed $2m_d$.

6.2 The consequences of pivoting

With this choice of **C**, **S** is a permutation of the matrix

$$\left(\begin{array}{cc|c} \beta & \mathbf{s}_1^T & \mathbf{s}_2^T \\ \mathbf{s}_1 & \mathbf{C} & \mathbf{S}_{12} \\ \hline \mathbf{s}_2 & \mathbf{S}_{21} & \mathbf{S}_{22} \end{array} \right), \tag{6.1}$$

where β is the candidate b_- pivot. If we were now to pivot on $\mathbf{C}$ instead of β, we would have eliminated all m $\mathbf{A}$-rows of $\mathbf{K}^{(k)}$ and, because of the choice of $\mathbf{C}$, the factorized matrix (the submatrix of $\mathbf{K}^{(k)}$ corresponding to eliminated rows) would have exactly m negative eigenvalues. Thus $\mathbf{B}^{(k)}$ is second-order sufficient if and only if the matrix

$$\begin{pmatrix} \beta & \mathbf{s}_2^T \\ \mathbf{s}_2 & \mathbf{S}_{22} \end{pmatrix} - \begin{pmatrix} \mathbf{s}_1^T \\ \mathbf{S}_{21} \end{pmatrix} \mathbf{C}^{-1} \left(\mathbf{s}_1 \quad \mathbf{S}_{12} \right) \tag{6.2}$$

is sufficiently positive definite. In particular, if $\beta - \mathbf{s}_1^T\mathbf{C}^{-1}\mathbf{s}_1$ is insufficiently positive, $\mathbf{B}^{(k)}$ is not second-order sufficient and should be modified.

If

$$\beta - \mathbf{s}_1^T\mathbf{C}^{-1}\mathbf{s}_1 < \sigma, \tag{6.3}$$

we modify $\mathbf{B}^{(k)}$ by replacing β by at least $\mathbf{s}_1^T\mathbf{C}^{-1}\mathbf{s}_1 + \sigma$. Conversely, if

$$\beta - \mathbf{s}_1^T\mathbf{C}^{-1}\mathbf{s}_1 \geq \sigma > 0, \tag{6.4}$$

it is safe to pivot on β as the reduction in the number of negative eigenvalues available through using β is reflected in a reduction in the number of available negative eigenvalues in the Schur complement $\mathbf{C} - \mathbf{s}_1\mathbf{s}_1^T/\beta$. This follows directly from the inertial identity

$$In(\mathbf{C} - \mathbf{s}_1\mathbf{s}_1^T/\beta) = In(\mathbf{C}) + In(\beta - \mathbf{s}_1^T\mathbf{C}^{-1}\mathbf{s}_1) - In(\beta), \tag{6.5}$$

where $In(\mathbf{M})$ is the inertia of a given matrix $\mathbf{M}$ (see, e.g. Cottle, 1974). We then pivot on the possibly modified value of β and replace $\mathbf{C}$ by $\mathbf{C} - \mathbf{s}_1\mathbf{s}_1^T/\beta$ — we update the matrix factorization to account for this (see, Gill *et al.*, 1974). We repeat this procedure until we have eliminated the remaining $\mathbf{S}_{22}$ rows, at which point, the only non eliminated portion of $\mathbf{K}^{(k)}$ is the (updated) matrix $\mathbf{C}$.

Alternatively, once it has been determined that $\mathbf{B}^{(k)}$ is not second-order sufficient, we might modify all remaining $\mathbf{B}^{(k)}$ pivots. One possibility, in the same vein as Schnabel and Eskow (1991), is to insist that all diagonals are larger than the sum of the absolute values of the (remaining) off diagonal terms in $\mathbf{B}^{(k)}$-rows.

6.3 Other pivot types

If the only possible pivots in $\mathbf{B}^{(k)}$-rows are zero or small, we may again test them one at a time to see if they might be modified and then used as pivots. If the test reveals that the matrix is not second-order sufficient, we may modify the tested pivot and pivot on it. But, if the test is inconclusive, we must reject the potential pivot and pass to the next.

It may be better to consider 2 by 2 pivots,

$$\begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix}, \tag{6.6}$$

arising from the $\mathbf{B}^{(k)}$-rows of $\mathbf{S}$, especially when the only possible 1 by 1 pivots are small or zero. Then $\mathbf{S}$ is a permutation of the matrix

$$\left(\begin{array}{ccc|c} \beta_{11} & \beta_{12} & \mathbf{s}_{11}^T & \mathbf{s}_{12}^T \\ \beta_{21} & \beta_{22} & \mathbf{s}_{21}^T & \mathbf{s}_{22}^T \\ \mathbf{s}_{11} & \mathbf{s}_{21} & \mathbf{C} & \mathbf{S}_{12} \\ \hline \mathbf{s}_{12} & \mathbf{s}_{22} & \mathbf{S}_{21} & \mathbf{S}_{22} \end{array} \right), \tag{6.7}$$

and $B^{(k)}$ is second-order sufficient only if the matrix

$$\begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix} - \begin{pmatrix} \mathbf{s}_{11}^T \\ \mathbf{s}_{21}^T \end{pmatrix} \mathbf{C}^{-1} (\mathbf{s}_{11} \quad \mathbf{s}_{21}) \tag{6.8}$$

is sufficiently positive definite. As before, if (6.8) is indefinite, the potential pivot (6.6) should be modified before use. The inertial result

$$\begin{aligned} & In\left(\mathbf{C} - (\mathbf{s}_{11}^T \quad \mathbf{s}_{12}^T) \begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{s}_{11} \\ \mathbf{s}_{12} \end{pmatrix}\right) = \\ & In(\mathbf{C}) + In\left(\begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix} - \begin{pmatrix} \mathbf{s}_{11}^T \\ \mathbf{s}_{21}^T \end{pmatrix} \mathbf{C}^{-1} (\mathbf{s}_{11} \quad \mathbf{s}_{21})\right) - In\begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix} \end{aligned} \tag{6.9}$$

once again indicates that the updated $\mathbf{C}$ after the pivot inherits the correct number of negative eigenvalues.

7 Mixed direct-iterative methods

Another possibility is to combine the iterative methods of Section 3.2 with a variation on the direct methods of Sections 5 and 6 so as to counteract the latter's difficulties with unacceptable fill-in.

We suppose we have started a factorization of $\mathbf{K}$ using Forsgren and Murray's (1993) pivot scheme, have eliminated all the $\mathbf{A}$-rows and n_e $\mathbf{B}^{(k)}$-rows, perhaps modifying pivots as we proceed, but that the remaining Schur complement matrix is too large and dense to proceed further. This is equivalent to permuting the matrix $\mathbf{K}$ to produce a partition (5.1), where $\mathbf{B}_{11}$ and $\mathbf{A}_1$ are now n_e by n_e and m by n_e matrices, respectively. As before, a similar permutation and partition of the solution and right-hand side vectors enables us to decompose the solution process into successively solving the three systems of equations (5.4)–(5.6). We have been careful to arrange that the matrix (5.7) has a sparse factorization but unfortunately the matrix (5.8) is too large and dense for us to factorize. However, the relationship (5.8) indicates that matrix-vector products between $\mathbf{S}$ and a given vector $\mathbf{v}$ are possible, each product requiring the solution of an intermediate linear system, whose coefficient matrix is (5.7), sandwiched between a pair of sparse-matrix-vector products. Thus we may contemplate solving (5.4)–(5.6) by solving (5.4) and (5.6), using the factorization of (5.7), and (5.5), using an appropriate iterative method.

It is convenient to view the solution of (5.5) as the solution of the related problem

$$\underset{\mathbf{p}_2 \in \Re^{n-n_e}}{\text{minimize}} \quad \tfrac{1}{2}\mathbf{p}_2^T \mathbf{B}_z \mathbf{p}_2 + \mathbf{p}_2^T \mathbf{g}_z, \tag{7.1}$$

where $\mathbf{B}_z = \mathbf{S}$ and $\mathbf{g}_z = \mathbf{g}_2 + \mathbf{B}_{12}^T \mathbf{q}_1 + \mathbf{A}_2^T \boldsymbol{\lambda}_2$. Then we may use any of the iterative methods discussed in Section 3.2 to compute an appropriate, approximate solution to (7.1), modifying $\mathbf{B}_z$ if necessary.

8 Conclusions

We have examined several approaches to the solution of structured linear systems of the form (2.2) which arise as subproblems in optimization calculations. We have indicated a method for modifying the matrix during a conjugate gradient iteration that maintains second-order sufficiency with a bounded error. We have shown in practice that this technique chooses good search directions. We have also explored direct factorization techniques and have described a method for a stable factorization with modifications to satisfy the second-order sufficiency condition while preserving sparsity. For a general purpose robust solver, it is likely that a combination of these methods will be required. A software package incorporating many of these ideas is currently under development (see, Gould, 1993).

References

[Bongartz *et al.*, 1993] I. Bongartz, A. R. Conn, N. I. M. Gould, and Ph. L. Toint. CUTE: Constrained and Unconstrained Testing Environment. Technical Report TR/PA/93/10, CERFACS, Toulouse, France, 1993.

[Bunch and Kaufman, 1977] 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.

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

[Coleman and Pothen, 1984] T. F. Coleman and A. Pothen. The sparse null space basis problem. Technical Report CS-84-12, Department of Computer Science, The Pennsylvania State University, University Park, USA, 1984.

[Coleman and Pothen, 1986] T. F. Coleman and A. Pothen. The null space problem II: Algorithms. Technical Report CS-86-09, Department of Computer Science, The Pennsylvania State University, University Park, USA, 1986.

[Conn *et al.*, 1991] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. A comprehensive description of LANCELOT. Technical Report 91/10, Department of Mathematics, FUNDP, Namur, Belgium, 1991.

[Conn *et al.*, 1992] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. A globally convergent Lagrangian barrier algorithm for optimization with general inequality constraints and simple bounds. Technical Report 92-067, Rutherford Appleton Laboratory, Chilton, England, 1992.

[Conn *et al.*, 1993] A. R. Conn, N. I. M. Gould, A. Sartenaer, and Ph. L. Toint. Global convergence properties of two augmented Lagrangian algorithms for optimization with a combination of general equality and linear constraints. Technical Report TR/PA/93/26, CERFACS, Toulouse, France, 1993.

[Cottle, 1974] R. W. Cottle. Manifestations of the Schur complement. *Linear Algebra and Applications*, 8:189–211, 1974.

[Dembo *et al.*, 1982] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact-Newton methods. *SIAM Journal on Numerical Analysis*, 19(2):400–408, 1982.

[Dennis and Schnabel, 1983] J. E. Dennis and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations.* Prentice-Hall, Englewood Cliffs, USA, 1983.

[Duff and Reid, 1982] I. S. Duff and J. K. Reid. MA27: A set of Fortran subroutines for solving sparse symmetric sets of linear equations. Report R-10533, AERE Harwell Laboratory, Harwell, UK, 1982.

[Duff and Reid, 1983] 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.

[Duff and Reid, 1993] I. S. Duff and J. K. Reid. MA47: A set of Fortran subroutines for solving sparse symmetric sets of linear equations. Report (to appear), Rutherford Appleton Laboratory, Didcot, UK, 1993.

[Duff *et al.*, 1979] 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.

[Duff *et al.*, 1986] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct methods for sparse matrices.* Clarendon Press, Oxford, UK, 1986.

[Duff *et al.*, 1991] 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.

[Fiacco and McCormick, 1968] A. V. Fiacco and G. P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* J. Wiley and Sons, New-York, 1968.

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

[Fletcher, 1987] R. Fletcher. *Practical Methods of Optimization.* J. Wiley and Sons, Chichester, second edition, 1987.

[Forsgren and Murray, 1993] A. L. Forsgren and W. Murray. Newton methods for large-scale linear equality-constrained minimization. *SIAM Journal on Matrix Analysis and Applications*, 14(2):560–587, 1993.

[Gay, 1981] D. M. Gay. Computing optimal locally constrained steps. *SIAM Journal on Scientific and Statistical Computing*, 2:186–197, 1981.

[Gill and Murray, 1974] P. E. Gill and W. Murray. Newton-type methods for unconstrained and linearly constrained optimization. *Mathematical Programming*, 28:311–350, 1974.

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

[Gill *et al.*, 1981] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press, London and New York, 1981.

[Gill *et al.*, 1990] 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. Hammarling, editors, *Reliable Numerical Computation*, pages 113–138, Oxford, 1990. Clarendon Press.

[Gill *et al.*, 1992] P. E. Gill, W. Murray, D. B. Poncelèon, and M. A. Saunders. Preconditioners for indefinite systems arising in optimization. *SIAM Journal on Matrix Analysis and Applications*, 13:292–311, 1992.

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

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

[Gould, 1993] N. I. M. Gould. KKTSOL/MA49: A set of Fortran subroutines for solving sparse symmetric sets of linear equations which arise in constrained optimization calculations. Technical Report (in preparation), CERFACS, Toulouse, France, 1993.

[Greenstadt, 1967] J. Greenstadt. On the relative efficiencies of gradient methods. *Mathematics of Computation*, 21:360–367, 1967.

[Hager, 1984] W. W. Hager. Condition estimates. *SIAM Journal on Scientific and Statistical Computing*, 5:311–316, 1984.

[Harwell Subroutine Library, 1990] Harwell Subroutine Library. *A catalogue of subroutines (release 10).* Advanced Computing Department, Harwell Laboratory, Harwell, UK, 1990.

[Hebden, 1973] M. D. Hebden. An algorithm for minimization using exact second derivatives. Technical Report T. P. 515, AERE Harwell Laboratory, Harwell, UK, 1973.

[Higham, 1988] N. J. Higham. FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation. *ACM Transactions on Mathematical Software*, 14:381–396, 1988.

[Lanczos, 1950] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of research of the National Bureau of Standards B*, 45:225–280, 1950.

[Lasdon *et al.*, 1973] L. S. Lasdon, R. L. Fox, and M. W. Ratner. An efficient one-dimensional search procedure for barrier functions. *Mathematical Programming*, 4(3):279–296, 1973.

[Markowitz, 1957] H. M. Markowitz. The elimination form of the inverse and its application to linear programming. *Management Science*, 3:255–269, 1957.

[Moré, 1978] J. J. Moré. The Levenberg-Marquardt algorithm: implementation and theory. In G. A. Watson, editor, *Proceedings Dundee 1977*, Berlin, 1978. Springer Verlag. Lecture Notes in Mathematics.

[Murray and Wright, 1992] W. Murray and M. H. Wright. Line search procedures for the logarithmic barrier function. Numerical Analysis Manuscript 92-01, AT&T Bell Laboratories, Murray Hill, USA, 1992.

[Murtagh and Saunders, 1978] B. A. Murtagh and M. A. Saunders. Large-scale linearly constrained optimization. *Mathematical Programming*, 14:41–72, 1978.

[Nash, 1984] S. G. Nash. Newton-type minimization via the Lanczos method. *SIAM Journal on Numerical Analysis*, 21(4), 1984.

[Paige and Saunders, 1975] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 12(4):617–629, 1975.

[Parlett and Reid, 1981] B. N. Parlett and J. K. Reid. Tracking the progress of the Lanczos algorithm for large symmetric eigenproblems. *Journal of the Institute of Mathematics and its Applications*, 1:135–155, 1981.

[Parlett, 1980] B. N. Parlett. *The Symmetric Eigenvalue Problem.* Prentice-Hall, Englewood Cliffs, USA, 1980.

[Schlick, 1992] T. Schlick. Modified Cholesky factorizations for sparse preconditioners. *SIAM Journal on Scientific and Statistical Computing*, (to appear), 1992.

[Schnabel and Eskow, 1991] R. B. Schnabel and E. Eskow. A new modified Cholesky factorization. *SIAM Journal on Scientific and Statistical Computing*, 11:1136–1158, 1991.

[Sorensen, 1980] D. C. Sorensen. Newton's method with a model trust-region modification. Technical Report ANL-80-106, Argonne National Laboratory, Argonne, USA, 1980.

[Stern and Vavasis, 1993] J. M. Stern and S. A. Vavasis. Nested dissection for sparse nullspace bases. *SIAM Journal on Matrix Analysis and Applications*, 14:766–775, 1993.

[Wright, 1992] M. H. Wright. Interior methods for constrained optimization. *Acta Numerica*, 1:341–407, 1992.