An Overview of Unconstrained Optimization *

by

R. Fletcher†

Numerical Analysis Report NA/149, June 1993

Abstract

Developments in the theory and practice of unconstrained optimization are described. Both line search and trust region prototypes are explained and various algorithms based on the use of a quadratic model are outlined. The properties and implementation of the BFGS method are described in some detail and the current preference for this approach is discussed. Various conjugate gradient methods for large unstructured systems are given, but it is argued that limited memory methods are considerably more effective without a great increase in storage or time. Further developments in this area are described. For structured problems the possibilities for updates that retain sparsity are described, including a recent proposal which maintains positive definite matrices and reduces to the BFGS update in the dense case. The alternative use of structure in partially separable optimization is also discussed.

1. Background and Introduction

The development of modern numerical methods for unconstrained optimization has taken place over the last thirty or forty years and considerable progress has been made. We now have good methods for many classes of problem and many insights into why these methods are successful. There is a wealth of literature available. Nevertheless there are a number of open questions of interest, and new ideas continue to enrich the field.

This paper reviews the progress made in *practical* methods of unconstrained optimization, and the relevant theory. Relevant considerations include the following.

- Ease of implementation, reliability, insensitivity to roundoff errors.
- Practical experience on actual problems and test problems.
  Relative performance measures. Ease of use.
- Global convergence to a solution from a remote starting point.

---

* Presented at the NATO ASI on Algorithms for Continuous Optimization, Il Ciocco, Italy, 5th-18th Sept., 1993

† Dept. of Mathematics and Computer Science, University of Dundee, Dundee DD1 4HN, Scotland, U.K., e-mail: fletcher@uk.ac.dundee.mcs

- Local convergence. Rate of convergence.
- Idealized behaviour. Worst case performance.
- Effects of size and/or structure.

The presentation reflects the author's view of the field, but many other interesting contributions are omitted due to limitations of space.

The problem of interest can be stated as that of finding a local solution x^* to the problem

$$\text{minimize} \quad f(x) \qquad x \in \mathbb{R}^n. \tag{1.1}$$

Usually x^* exists and is locally unique, although there are some pathological exceptions. There are two particular caveats that must be made. One is that these methods do not guarantee to find a global solution to (1.1), but one which is optimal only in some neighbourhood of x^*. Global optimization is an order of magnitude more difficult and is reviewed in the paper of Evtushenko at this conference. Another caveat is that the objective function $f(x)$ must be sufficiently smooth in some sense. Nonsmooth functions, such as arise when max or modulus operations are used to define $f(x)$, require a different approach more related to constrained optimization. The minimization of nonsmooth functions is reviewed in the paper of Zowe at this conference. In what follows it is assumed that the *gradient vector* denoted by $g(x) = \nabla f(x)$ and also the *Hessian matrix* denoted by $G(x) = \nabla^2 f(x) = [\partial^2 f/\partial x_i \partial x_j]$ are continuously differentiable functions of x.

A necessary condition for x^* to solve (1.1) is that x^* is a *stationary point*, that is $g^* = 0$. (The notation $g^* = g(x^*)$ etc. is used.) This is equivalent to having zero slope $s^T g* = 0$ in any direction s from x^*. Another necessary condition is that the matrix G^* is positive semi-definite, which is equivalent to nonnegative curvature $s^T G^* s \geq 0$ in any direction. It is then convenient to refer to x^* as a *semi-definite point*. A sufficient condition for x^* to solve (1.1) is that both $g^* = 0$ and G^* is positive definite. These conditions play a fundamental part in the development of methods for unconstrained optimization. The proof of these results and more details pertaining to the subject can be found for example in Fletcher [21].

Methods for unconstrained optimization differ according to how much information the user feels able to provide. The most desirable situation from the point of view of providing useful information is that the user provides subroutines from which $f(x)$, $g(x)$ and $G(x)$ can be evaluated for any x (a *second derivative method*). However programming the Hessian can be quite demanding for the user and often the lesser requirement of providing subroutines for $f(x)$ and $g(x)$ is made (a *gradient method*). Most of this paper relates to gradient methods. Even less demanding is that the user only provides a subroutine for $f(x)$ (a *no-derivative method*). Such methods have proved considerably less effective and robust than gradient methods and I shall not discuss them here. Probably the best advice is to use high precision arithmetic and estimate first derivatives by finite differences, in conjunction with a gradient method. Recent methods for optimization may also ask for information about the structure of $f(x)$ in some way, and this can lead to significant improvements in efficiency. Also recent developments in automatic differentiation (see Dixon's paper at this conference) make second derivative methods much more attractive.

Methods for unconstrained optimization are generally iterative methods in which the user typically provides an initial estimate $x^{(1)}$ of x^*, and possibly some additional informa-

tion. A sequence of iterates $\{x^{(k)}\}$ is then generated according to some algorithm. Usually the algorithm is such that the sequence of function values $\{f^{(k)}\}$ is monotonically decreasing ($f^{(k)}$ denotes $f(x^{(k)})$ etc.). A practical algorithm must be well-defined and converge from any $x^{(1)}$, which is the issue of *global convergence*. This issue is usually addressed by aiming to show theoretically that accumulation points of the sequence $\{x^{(k)}\}$ satisfy the necessary conditions for a local solution, and in particular that they are stationary points. Another important theoretical issue is that of *local convergence* and involves a study of the rate at which the sequence converges. In practice the sequence is terminated after a finite number of iterations, if for example the gradient vector is sufficiently small, or negligible progress is being made. In addition to theoretical studies it is important that methods are shown to work well in practice. To this end, libraries of suitable *test problems* (e.g. Moré, Garbow and Hillstrom [43]) have been compiled, which can be used for algorithm development.

The usual practice is to design a method on the basis of some *model* of the objective function. Most fruitful has been the use of a *quadratic model* in which $f(x)$ is approximated on the kth iteration by a Taylor series approximation about $x^{(k)}$, that is

$$f(x^{(k)} + \delta) \approx q^{(k)}(\delta) = f^{(k)} + \delta^T g^{(k)} + \tfrac{1}{2}\delta^T B^{(k)} \delta \tag{1.2}$$

where δ represents a displacement to $x^{(k)}$, and $B^{(k)}$ is either the Hessian $G^{(k)}$ or some approximation to it. A fundamental algorithm known as *Newton's method* calculates the displacement $\delta^{(k)}$ on iteration k as the minimizer of $q^{(k)}(\delta)$. Newton's method is a second derivative method so that $B^{(k)} = G^{(k)}$ is used. A unique minimizer of $q^{(k)}(\delta)$ exists if and only if $G^{(k)}$ is positive definite and Newton's method is only well-defined in this case. Then $\delta^{(k)}$ is obtained by finding the stationary point of $q^{(k)}(\delta)$, which requires the solution of the linear system

$$G^{(k)}\delta^{(k)} = -g^{(k)}. \tag{1.3}$$

This is most readily done by finding the LL^T (Choleski) or LDL^T factors of $G^{(k)}$, which also enables the positive definite requirement to be checked. The next iterate is then defined by

$$x^{(k+1)} = x^{(k)} + \delta^{(k)}. \tag{1.4}$$

Usually $G^{(k)}$ is positive definite when $x^{(k)}$ is in some neighbourhood of x^*. It can then be proved that the sequence $\{x^{(k)}\}$ converges, and the order of convergence is second order. These local convergence properties represent the ideal local behaviour which other algorithms aim to emulate as far as possible. In fact *superlinear convergence* of any algorithm is obtained if and only if the step $\delta^{(k)}$ is asymptotically equal to the step given by solving (1.3). This fundamental result due to Dennis and Moré [17] emphasizes the importance of the Newton step for local convergence.

When $x^{(k)}$ is remote from x^* then Newton's method may not converge, and may not be defined (when $G^{(k)}$ is not positive definite). Globally convergent methods are often designed on the basis of some *prototype algorithm* which specifies an overall approach within which some variation might be possible. The two main protoypes are *line search methods* and *trust region methods*. In a line search method the kth iteration involves the following steps

- determine a direction of search $s^{(k)}$
- find $\alpha^{(k)}$ to minimize $f(x^{(k)} + \alpha s^{(k)})$ with respect to α
- set $x^{(k+1)} = x^{(k)} + \alpha^{(k)} s^{(k)}$.

Different methods correspond to different ways of choosing $s^{(k)}$. This choice is of significant importance in obtaining an efficient method and we return to it below. It is necessary (when $g^{(k)} \neq 0$) that $s^{(k)}$ is a *descent direction* so that a reduction in $f(x)$ can be guaranteed for sufficiently small $\alpha > 0$. This is equivalent to the condition

$$s^{(k)^T} g^{(k)} < 0, \tag{1.5}$$

showing that the slope of $f(x)$ along the line $x = x^{(k)} + \alpha s^{(k)}$ is negative. Finding $\alpha^{(k)}$ is the *line search subproblem* which is carried out by repeatedly sampling $f(x)$ and $g(x)$ (if available) at different points along the line $x = x^{(k)} + \alpha s^{(k)}$. In the form stated above, this step is idealized (an *exact line search*) insofar as the exact minimizer cannot usually be calculated in a finite number of arithmetic operations. It is also not efficient to find $\alpha^{(k)}$ to high accuracy when $x^{(k)}$ is remote from x^*. In practice an *inexact line search* is used which enables a significant reduction in $f(x)$ to be made at the cost of only a few evaluations of $f(x)$ and $g(x)$. Algorithms for the line search subproblem are quite sophisticated and are described in more detail for example in [21].

An ideal choice for $s^{(k)}$ when $G^{(k)}$ is positive definite is the Newton direction obtained by solving (1.3). Likewise when $B^{(k)}$ approximates $G^{(k)}$ then $s^{(k)}$ can be obtained by solving

$$B^{(k)} s^{(k)} = -g^{(k)} \tag{1.6}$$

by analogy with (1.3). Such methods are referred to as *Newton-like methods*. It is important that $B^{(k)}$ is positive definite which is a sufficient condition for the descent property (1.5) (because $s^{(k)^T} g^{(k)} = -s^{(k)^T} B^{(k)} s^{(k)} < 0$). Otherwise it is possible that $s^{(k)^T} g^{(k)} = 0$ when $g^{(k)} \neq 0$, and the method fails to make further progress. A simple choice is $B^{(k)} = I$ (the unit matrix) giving $s^{(k)} = -g^{(k)}$. This is known as the *steepest descent method* on account of the steepest slope property of the gradient vector. Despite this apparently attractive property, the worst case behaviour of the method is to exhibit very slow linear convergence and it is this that is usually observed in practice. When $G^{(k)}$ is available but is indefinite, a good strategy is to choose $B^{(k)} = G^{(k)} + D^{(k)}$ where $D^{(k)}$ is a sufficiently large diagonal matrix. A popular method is that of Schnabel and Eskow [63], and the method of (1.10) below can also be interpreted in this way. In the case of a gradient method, an attractive way of specifying $B^{(k)}$ is that used in the class of *quasi-Newton methods*. Another type of line search method that uses quadratic information in a different way is the *conjugate gradient method*. Both these developments are described in detail below.

A different prototype is the *trust region method*, which may be regarded as addressing the difficulties posed by the lack of a minimizer for the quadratic model (1.2) when $B^{(k)}$ is indefinite. On iteration k a *trust region* of radius $\rho^{(k)}$ about $x^{(k)}$ is defined by the set

$$\Omega^{(k)} = \{x : \|x - x^{(k)}\| \leq \rho^{(k)}\}. \tag{1.7}$$

and $x^{(k+1)}$ is constrained to lie within this set. Thus $x^{(k+1)} = x^{(k)} + \delta^{(k)}$ is determined by choosing $\delta^{(k)}$ as the solution of the *trust region subproblem*

$$\text{minimize} \quad q^{(k)}(\delta) \qquad \text{subject to} \quad \|\delta\| \leq \rho^{(k)}, \tag{1.8}$$

and it is unnecessary to require that $B^{(k)}$ is positive definite. The aim is to choose the radius so that $q^{(k)}(\delta)$ agrees with $f(x^{(k)}+\delta)$ to a certain extent for $x^{(k)}+\delta$ in $\Omega^{(k)}$, hence the term trust region. The algorithm adjusts $\rho^{(k)}$ adaptively to obtain a certain level of agreement between the actual and predicted reduction in $f(x)$ that is obtained in the algorithm, whilst keeping $\rho^{(k)}$ as large as possible. If the Newton-like step $\delta^{(k)}$ defined by solving (1.3) or (1.6) is feasible in (1.8) then it is taken by the trust region algorithm. Otherwise, and when the 2-norm is used in (1.7), then (1.8) reduces to the solution of the nonlinear equation (in λ)

$$\|\delta(\lambda)\|_2^2 = \rho^{(k)2} \tag{1.9}$$

in which $\delta(\lambda)$ is defined by solving the linear system

$$(B^{(k)} + \lambda I)\delta = -g^{(k)}. \tag{1.10}$$

In general (1.9) cannot be solved in a finite number of operations, but an acceptable approximation can usually be obtained in two or three inner iterations, each of which requires the solution of (1.10) (see Moré [42]). If the ∞-norm is used in (1.7) then (1.8) becomes a quadratic programming problem with simple bounds. Although this can be solved finitely, it is usually more expensive in practice, and if $B^{(k)}$ is not positive definite then the existence of local but not global solutions to (1.10) may cause potential difficulty.

An attractive feature of trust region methods is the strong global convergence properties that can be proved. It is only necessary to assume that $\|B^{(k)}\|$ is $O(k)$, in order to show that the sequence $\{x^{(k)}\}$ has an accumulation point which is stationary, and this result allows $B^{(k)}$ to be indefinite. If $B^{(k)} = G^{(k)}$ then it can also be proved that the accumulation point is semi-definite. For a line search descent method the global convergence results are less strong and some aspects of these are now described. First of all it is necessary to ensure that a sufficient decrease in f is made in the line search. A possible strategy is to accept a step $\alpha^{(k)}$ if it satisfies the *Wolfe conditions*

$$f(x^{(k)} + \alpha^{(k)}s^{(k)}) \leq f^{(k)} + \rho\alpha^{(k)}{g^{(k)}}^T s^{(k)} \tag{1.11}$$

$$-g(x^{(k)} + \alpha^{(k)}s^{(k)})^T s^{(k)} \leq -\sigma {g^{(k)}}^T s^{(k)} \tag{1.12}$$

where $0 \leq \rho < \sigma \leq 1$ are preset parameters. A modified form (the *Strong Wolfe conditions*) is obtained by taking the modulus of the left hand side of (1.12). For some algorithms this is preferable as it allows an exact line search to be approached by taking σ arbitrarily close to zero. It is possible to develop line search algorithms which find such an $\alpha^{(k)}$ in a finite number of operations (see e.g. [21]).

To prove global convergence for a line search algorithm also requires an assumption that the angle $\theta^{(k)}$ between the vectors $s^{(k)}$ and $-g^{(k)}$ does not approach $\frac{\pi}{2}$ too rapidly. It follows from (1.11), (1.12) and Lipchitz continuity of $g(x)$ that

$$\sum_{k=1}^{\infty} \cos^2\theta^{(k)} \|g^{(k)}\|^2 < \infty. \tag{1.13}$$

If the sequence $\{s^{(k)}\}$ is such that there is an $\varepsilon > 0$ for which

$$\cos\theta^{(k)} \geq \varepsilon \tag{1.14}$$

then it readily follows from (1.13) that $g^{(k)} \to 0$. A sufficient condition for a Newton-like method to satisfy (1.14) is that the condition number $\|B^{(k)}\|\|B^{(k)^{-1}}\| \leq \varepsilon^{-1}$ is uniformly bounded. Unfortunately for most practical algorithms (1.14) cannot be asserted. However if the lesser aim of proving $g^{(k)} \to 0$ on a subsequence is considered, then it is only necessary to show that

$$\sum_{k=1}^{\infty} \cos^2 \theta^{(k)} = \infty. \tag{1.15}$$

It has been possible to show that some practical algorithms satisfy (1.15) but there are many other algorithms for which the question is open.

Line search algorithms are less appropriate when second derivatives are available in that for example they allow convergence to a saddle point which is not a semi-definite point. More significantly, they do not adequately use the knowledge that $G^{(k)}$ may have negative eigenvalues. Trust region algorithms provide one solution to this problem, but another advance has been the development of trajectory search algorithms which use directions of negative curvature. The idea is to determine a *descent pair* of directions $(s^{(k)}, d^{(k)})$ on each iteration, where loosely speaking, $s^{(k)}$ represents a descent direction calculated on the basis of positive curvature information in $G^{(k)}$, and $d^{(k)}$ is a *negative curvature descent direction* with ${d^{(k)}}^T G^{(k)} d^{(k)} < 0$. If $G^{(k)}$ is positive semi-definite then $d^{(k)} = 0$. This descent pair is used in a search along the trajectory

$$x = x^{(k)} + \alpha d^{(k)} + \alpha^2 s^{(k)}. \tag{1.16}$$

These ideas are formalised by Moré and Sorensen [44] who show that if the sequences $\{s^{(k)}\}$ and $\{d^{(k)}\}$ are bounded and satisfy

$${g^{(k)}}^T s^{(k)} \to 0 \quad \Rightarrow \quad g^{(k)} \to 0 \quad \text{and} \quad s^{(k)} \to 0 \tag{1.17}$$

and

$${d^{(k)}}^T G^{(k)} d^{(k)} \to 0 \quad \Rightarrow \quad \min\{\lambda_{\min}(G^{(k)}), 0\} \to 0 \quad \text{and} \quad d^{(k)} \to 0 \tag{1.18}$$

then accumulation points of the sequence $\{x^{(k)}\}$ are both stationary and semi-definite. Moré and Sorensen give analogues of the Wolfe conditions for finding an acceptable step $\alpha^{(k)}$ in the trajectory search. Within the scope of (1.17) and (1.18) there are many ways of defining the descent pair $(s^{(k)}, d^{(k)})$. Moré and Sorensen define a method based on the Bunch-Parlett factors of $G^{(k)}$. A recent paper of Forsgren, Gill and Murray [27] gives a method based on partial Choleski factors of $G^{(k)}$, which reduces to the regular Choleski factors when $G^{(k)}$ is positive definite.

The rest of this section is relevant to gradient methods, and examines different ways in which quadratic information can be estimated when the Hessian $G^{(k)}$, and hence the quadratic model (1.2), is not directly available. One idea has been to derive methods with the property of *quadratic termination*, in which the method is able to locate the minimizer of a quadratic function in a known finite number of iterations, yet which can be applied iteratively to minimize non-quadratic functions. A particular way of obtaining

quadratic termination is to invoke the concept of the *conjugacy* of a set of non-zero vectors $s^{(k)}$, $k = 1, 2, ..., n$ relative to a given positive definite matrix G. This is the property that

$$s^{(i)T} G s^{(j)} = 0 \qquad \forall\, i \neq j. \tag{1.19}$$

A *conjugate direction method* is a line search method which generates such directions when applied to a quadratic function with Hessian G. It is readily shown in this case that a conjugate direction method terminates in at most n iterations if exact line searches are made (e.g. [21]). (Subsequent references in this paper to termination properties for quadratic functions will assume that the Hessian is positive definite.) Of particular importance are *conjugate gradient methods* in which directions are generated by

$$\begin{aligned} s^{(1)} &= -g^{(1)} \\ s^{(k)} &= -g^{(k)} + \beta^{(k)} s^{(k-1)} \qquad \forall\, k > 1. \end{aligned} \tag{1.20}$$

The merit of this approach lies in the fact that the method can be implemented with only a few vectors of storage. Various choices of the scalar $\beta^{(k)}$ exist which give different performance on non-quadratic functions, yet are equivalent for quadratic functions. Such methods, and developments thereof, are considered in some detail in Section 3.

Another general approach is to determine a matrix $B^{(k)}$ which approximates to $G^{(k)}$. This approximation can then be used in various ways, e.g. by using (1.2) in a trust region algorithm or (1.6) in a line search algorithm. Mostly we shall concentrate on the latter case. One possibility is to estimate $B^{(k)}$ by finite differences using repeated evaluation of the gradient vector. This is disadvantageous both on the grounds of cost and the possible lack of symmetry and positive definiteness in $B^{(k)}$. These disadvantages are removed in the very important class of *quasi-Newton methods*. In these methods, $B^{(k)}$ is initially set to an arbitrary symmetric positive definite matrix (often $B^{(1)} = I$), and $B^{(k)}$ is *updated* after each iteration to obtain a new and hopefully better approximation $B^{(k+1)}$. The information available to do this is the pair of difference vectors

$$\delta^{(k)} = x^{(k+1)} - x^{(k)} \tag{1.21}$$

and

$$\gamma^{(k)} = g^{(k+1)} - g^{(k)}. \tag{1.22}$$

These vectors are related by a Taylor series expansion

$$\gamma^{(k)} = \bar{G}^{(k)} \delta^{(k)} \tag{1.23}$$

where

$$\bar{G}^{(k)} = \int_0^1 G(x^{(k)} + \theta \delta^{(k)})\, d\theta \tag{1.24}$$

is the average Hessian matrix along the step. By analogy with (1.23), $B^{(k+1)}$ is chosen to satisfy

$$\gamma^{(k)} = B^{(k+1)} \delta^{(k)} \tag{1.25}$$

known as the *secant equation* or *quasi-Newton condition*. Within this framework, many possible *updating formulae* are possible, and it is important to make a choice which is both readily computed and effective in practice. Over the last 20 years or so the *BFGS updating formula* has generally been regarded as an effective choice. We study this formula and the claims of alternative formulae in some detail in Section 2.

A special case of unconstrained optimization of considerable importance occurs when the objective function is a sum of m squared terms

$$f(x) = \sum_{i=1}^{m} [r_i(x)]^2 . \tag{1.26}$$

This is the *nonlinear least squares problem* as it can be viewed as an attempt to solve the system of m equations in n unknowns

$$r_i(x) = 0 \qquad\qquad i = 1, 2, ..., m. \tag{1.27}$$

When $m > n$ the system is *over-determined* and an exact solution cannot usually be found, but an approximate solution obtained by solving (1.26) is often of interest.

The problem can be solved by any available gradient or second derivative method, and derivatives of $f(x)$ are given by

$$g(x) = 2Ar \tag{1.28}$$

and

$$G(x) = 2AA^T + 2\sum_{i=1}^{m} r_i \nabla^2 r_i, \tag{1.29}$$

where r is the column vector with elements $r_i(x)$, and $A = A(x)$ is the *Jacobian matrix* whose columns are the vectors $\nabla r_i(x)$ for $i = 1, 2, ..., m$. Using (1.29) directly in Newton's method is somewhat inconvenient because the Hessian matrices $\nabla^2 r_i$ for all the functions r_i must be evaluated. However it is often the case (on account of (1.27)) that the functions $r_i(x)$ are close to zero at the solution. This suggests that a good approximation to $G(x)$ might be obtained by neglecting the final term in (1.29) to give

$$B^{(k)} = 2A^{(k)} {A^{(k)}}^T, \tag{1.30}$$

where $A^{(k)}$ denotes $A(x^{(k)})$. Use of (1.30) in a line search method is referred to as the *Gauss-Newton method*. The important feature is that using only the information (r and A) required to determine the gradient vector, it is often possible to obtain a good approximation to the Hessian matrix. Conversely a quasi-Newton method might take up to n iterations to build up a good approximation. Also (1.30) usually gives a positive definite matrix (unless $A^{(k)}$ is rank deficient).

Although the Gauss-Newton method often works well, there are some cases in which it works badly (e.g. problems with large residuals or in which $A^{(k)}$ becomes rank deficient). Various suggestions have been made for methods which improve the approximation given

by (1.30), and some of these are reviewed in [21]. My current preference is for a *hybrid method* in which $B^{(k)}$ is either the Gauss-Newton matrix (1.30), or the result of applying the BFGS updating formula to $B^{(k-1)}$, depending on the outcome of a certain test.

There are many other ideas relating to unconstrained optimization that are interesting and might occur in future methods. These include developments of simplex methods (Torczon [74], Wood [75]), non-monotone line searches (Grippo, Lampariello and Lucidi [35]), and steepest descent steps without line search (Barzilai and Borwein [3]) to name but a few.

2. Quasi-Newton Methods

In this section we examine quasi-Newton methods as described above, and look in particular at the many possible updating formulae that have been suggested. Mostly the context is that of a line search method, but the ideas are also relevant to use in trust region methods. The current preference for the BFGS formula is critically debated.

The quasi-Newton updating formula prescribes $B^{(k+1)}$ by making a change to $B^{(k)}$ so that the secant equation (1.25) is satisfied. The aim is to provide an update which is readily computed, and effective in practice. It is often convenient that the update should preserve positive definiteness in $B^{(k)}$. A particularly simple update is obtained if the correction is taken to be a symmetric rank-one matrix. The secant equation then determines the update uniquely as

$$B^{(k+1)} = B^{(k)} + \frac{(\gamma^{(k)} - B^{(k)}\delta^{(k)})(\gamma^{(k)} - B^{(k)}\delta^{(k)})^T}{(\gamma^{(k)} - B^{(k)}\delta^{(k)})^T\delta^{(k)}}, \tag{2.1}$$

and this is known as the *Symmetric Rank 1 (SR1)* formula. This update is suggested independently by various authors around 1967-8 (see [21] for references). The good and bad features of the update are discussed below, but it can be noted from (1.6) that the vector $\gamma^{(k)} - B^{(k)}\delta^{(k)}$ is a linear combination of $g^{(k)}$ and $g^{(k+1)}$. It turns out that the update preserves invariance under an affine transformation of the variables x, if the correction to $B^{(k)}$ is composed out of gradient vectors. The significance of this fact is discussed in more detail in [21], Chapter 3.3, but it is an indication that the resulting method is not unduly affected by, for example, changes in the relative scaling of the variables.

This consideration suggests that we might try an update in which the correction is a symmetric rank 2 matrix composed from $g^{(k+1)}$ and $g^{(k)}$, or equivalently (using (1.6)) from $\gamma^{(k)}$ and $B^{(k)}\delta^{(k)}$. This gives an update with three free parameters, and imposing the secant equation (1.25) fixes two of these. Thus a one-parameter family (the *Broyden family*) (Broyden [5]) is obtained

$$B^{(k+1)} = B^{(k)} - \frac{B^{(k)}\delta^{(k)}{\delta^{(k)}}^T B^{(k)}}{{\delta^{(k)}}^T B^{(k)}\delta^{(k)}} + \frac{\gamma^{(k)}{\gamma^{(k)}}^T}{{\delta^{(k)}}^T\gamma^{(k)}} + \theta^{(k)} w^{(k)} {w^{(k)}}^T \tag{2.2}$$

where $\theta^{(k)}$ is the free parameter and

$$w^{(k)} = ({\delta^{(k)}}^T B^{(k)}\delta^{(k)})^{\frac{1}{2}} \left(\frac{\gamma^{(k)}}{{\delta^{(k)}}^T\gamma^{(k)}} - \frac{B^{(k)}\delta^{(k)}}{{\delta^{(k)}}^T B^{(k)}\delta^{(k)}} \right). \tag{2.3}$$

The *DFP formula* given by $\theta^{(k)} = 1$ (Davidon [14], Fletcher and Powell [25]) was the first such formula to be discovered, and showed the effectiveness of this approach. However the *BFGS formula*, (Broyden [5], Fletcher [20], Goldfarb [30], Shanno [64]), corresponding to $\theta^{(k)} = 0$ in the Broyden family, is now the usual formula of choice and we look at the reasons for this below. An alternative way of presenting the family is to define $H = B^{-1}$ throughout (i.e. $H^{(k)} = {B^{(k)}}^{-1}$ etc.) in which case (2.2) becomes

$$H^{(k+1)} = H^{(k)} - \frac{H^{(k)}\gamma^{(k)}{\gamma^{(k)}}^T H^{(k)}}{{\gamma^{(k)}}^T H^{(k)}\gamma^{(k)}} + \frac{\delta^{(k)}{\delta^{(k)}}^T}{{\delta^{(k)}}^T\gamma^{(k)}} + \phi^{(k)} v^{(k)} {v^{(k)}}^T \tag{2.4}$$

where $\phi^{(k)}$ is now the free parameter and

$$v^{(k)} = ({\gamma^{(k)}}^T H^{(k)}\gamma^{(k)})^{\frac{1}{2}} \left(\frac{\delta^{(k)}}{{\delta^{(k)}}^T\gamma^{(k)}} - \frac{H^{(k)}\gamma^{(k)}}{{\gamma^{(k)}}^T H^{(k)}\gamma^{(k)}} \right). \tag{2.5}$$

It is indicated in [21], p.76 that $\theta^{(k)}$ and $\phi^{(k)}$ are related by

$$\theta^{(k)} = (\phi^{(k)} - 1)/(\phi^{(k)} - 1 - \phi^{(k)}\mu^{(k)}) \tag{2.6}$$

where $\mu^{(k)} = {\gamma^{(k)}}^T H^{(k)}\gamma^{(k)}\delta^{(k)T} B^{(k)}\delta^{(k)}/({\delta^{(k)}}^T\gamma^{(k)})^2$ ($\mu^{(k)} \geq 1$ by Cauchy's inequality). Thus in (2.5) $\phi^{(k)} = 1$ corresponds to the BFGS formula and $\phi^{(k)} = 0$ to the DFP formula. Note that the DFP update for H is obtained from the BFGS update for B by interchanging $H \leftrightarrow B$ and $\delta \leftrightarrow \gamma$. Formulae related in this way are said to be *dual*. The correction in (2.5) reduces to being rank 1 when $\phi^{(k)} = {\delta^{(k)}}^T\gamma^{(k)}/({\delta^{(k)}}^T\gamma^{(k)} - {\gamma^{(k)}}^T H^{(k)}\gamma^{(k)})$ giving the SR1 formula, which is seen to be self-dual from (2.6). It can often be useful to write the BFGS update of $H^{(k)}$ ($\phi^{(k)} = 1$ in (2.4)) in the form

$$H^{(k+1)} = \left(I - \frac{\delta^{(k)}{\gamma^{(k)}}^T}{{\delta^{(k)}}^T\gamma^{(k)}} \right) H^{(k)} \left(I - \frac{\gamma^{(k)}{\delta^{(k)}}^T}{{\delta^{(k)}}^T\gamma^{(k)}} \right) + \frac{\delta^{(k)}{\delta^{(k)}}^T}{{\delta^{(k)}}^T\gamma^{(k)}} \tag{2.7}$$

and the DFP update of $B^{(k)}$ is the dual of this.

The property of a formula that it maintains positive definite matrices is important, and a necessary condition from (1.25) is that ${\delta^{(k)}}^T\gamma^{(k)} > 0$. For a line search method, this inequality follows from (1.12). There is a critical value $\bar{\theta}^{(k)} = 1/(1 - \mu^{(k)}) < 0$ such that $B^{(k+1}$ is singular, and $B^{(k)}$ is positive definite implies $B^{(k+1)}$ is positive definite if $\theta^{(k)} > \bar{\theta}^{(k)}$. Thus both the BFGS and DFP formulae retain positive definite matrices. The same is true for any formulae in the so-called *convex class* $\theta^{(k)} \in [0, 1]$. However the SR1 is never in the convex class and may give rise to an indefinite matrix. It is also possible for SR1 that $B^{(k)}$ might be singular ($\theta^{(k)} = 1/(1 - \mu^{(k)})$) or infinite ($\phi^{(k)} = 1/(1 - \mu^{(k)})$). These aspects of the SR1 formula have deterred many users, but there is currently some resurgence of interest. There are other self-dual formulae, some of which are in the convex class (e.g. $\phi^{(k)} = {\delta^{(k)}}^T\gamma^{(k)}/({\delta^{(k)}}^T\gamma^{(k)} + {\gamma^{(k)}}^T H^{(k)}\gamma^{(k)})$ and $\phi^{(k)} = 1/(1+\mu^{(k)\frac{1}{2}})$, the latter following by setting $\theta^{(k)} = \phi^{(k)}$ in (2.6)). Although self-duality seems a desirable symmetry

property for positive definite updates, such formulae have attracted little interest and little is known about their properties.

We now turn to the properties of quasi-Newton methods that use updates from the Broyden family. It is assumed that the updates do not break down due to $B^{(k)}$ or $H^{(k)}$ becoming singular. If exact line searches are carried out then the sequence of iterates $\{x^{(k)}\}$ is independent of the choice of $\phi^{(k)}$. If $f(x)$ is quadratic, then the method terminates at the solution in at most n iterations, and $B^{(n)} = G$. Also the search directions are conjugate, and are equivalent to directions generated by the conjugate gradient method if $B^{(1)} = I$. A remarkable result due to Dixon [19] shows that non-dependence on $\phi^{(k)}$ (assuming an exact line search) carries over to non-quadratic functions. Thus Powell's result [55] that DFP converges globally if the line search is exact and $f(x)$ is strictly convex, implies that the same holds for any update from the Broyden family. Practical comparisons on test problems (e.g. [21]) support this theory, showing that if the parameter σ in the strong Wolfe conditions is small ($\sigma \leq 0.1$ say) then there is little to choose between the various update formulae.

It is usually the case however that quasi-Newton methods require fewer function and gradient evaluations overall, when a very weak line search criterion (say $\sigma = 0.9$) is used. In this case the DFP method often performs quite badly, which is not true of the BFGS method. Much effort has been expended in trying to explain this, and in promoting the claims of other updating formulae. Local convergence theory (see for example Dennis and Schnabel [18]) does not hint at any difference between DFP and BFGS. If $x^{(1)}$ is sufficiently close to a local minimizer x^*, and $B^{(1)}$ is sufficiently close to G^*, then the iterates of both the BFGS and DFP methods with steps $\alpha^{(k)} = 1$ converge to x^* superlinearly. In fact it has become clear that the inefficiency of the DFP method is caused by the inability to reduce large eigenvalues in $B^{(k)}$. The main contribution here is due to Powell [59] and is described in the next paragraph. However a similar conclusion is also drawn from global convergence results. Powell [56] proves global convergence of the BFGS method when $f(x)$ is convex and the Wolfe conditions are satisfied in the line search. Byrd, Nocedal and Yuan [9] extend this result for all $\theta^{(k)} \in [0, 1)$ but *not* for $\theta^{(k)} = 1$ (the DFP method). The method of proof in the latter case fails because a term which reduces large eigenvalues of $B^{(k)}$ is no longer present. Convergence of the DFP method under these conditions is an open question of some interest *. Another open question which has defied researchers for many years is the global convergence of the BFGS method for *non-convex* functions. Nonetheless in most implementations of a line search method, the BFGS formula has usually been the update formula of choice, due to its good theoretical properties in the convex case, and its good performance on test problems and in practice. Later in this section we discuss whether this preference is likely to continue in the future, and what possibilities there are for alternative formulae.

Powell [59] isolates a critical difference between the DFP and BFGS methods by considering the worst case behaviour on quadratic functions. Because of invariance under linear transformations, he assumes without loss of generality that $G = I$. He also assumes that steps $\alpha^{(k)} = 1$ are taken, and that the parameter σ in the Wolfe conditions is such

* Some discussion at the conference suggested that Sargent has proved this using techniques developed by Griewank and Toint, but there does not seem to be a paper available.

that the step is accepted. Because $G = I$ it follows that $\gamma^{(k)} = \delta^{(k)}$ and (1.25) then shows that $B^{(k+1)}$ always has a unit eigenvalue. It is known (Fletcher [20]) that under these circumstances the eigenvalues of $B^{(k)}$ converge monotonically towards 1 for any formula in the convex class. Powell considers the simple case that $n = 2$ and assumes that $B^{(k)}$ has eigenvalues 1 and $\lambda^{(k)}$ for all k. He then analyses the convergence of the DFP and BFGS methods to see how fast $\lambda^{(k)}$ approaches 1. The initial approximation $x^{(1)}$ is chosen so that the worst case behaviour is attained for the formula being studied. Ultimately both formulae give rise to superlinear convergence, but there are significant differences in the number of iterations required to reach a neighbourhood in which superlinear convergence becomes apparent. When $\lambda^{(1)}$ is much smaller than 1 then both methods are very efficient and only a few iterations are required. Conversely when $\lambda^{(1)}$ is much larger than 1 then the DFP method may become very inefficient, and in the worst case Powell shows that

$$\lambda^{(k+1)} \approx \lambda^{(k)} - 1 \tag{2.8}$$

when $\lambda^{(k)}$ is large. Thus for example if $\lambda^{(1)} = 1000$ then about 1000 iterations are required to solve the problem. The behaviour of the resulting method is that a sequence of very short steps are taken and the iterates $x^{(k)}$ approach the solution in a very slowly converging spiral. Similar behaviour has been observed for the DFP method in non-quadratic problems and it is likely that a similar mechanism is at work. In contrast the BFGS method does not exhibit this behaviour and for the $n = 2$ quadratic it requires only about $2.39 \log_{10} \lambda^{(1)}$ iterations in the worst case.

It is nonetheless the case that the BFGS method is better at correcting small eigenvalues than large eigenvalues, which holds out the interesting possibility of using negative values of $\theta^{(k)}$ in a controlled way. Of course it would be necessary to have $\theta^{(k)} > \bar{\theta}^{(k)}$ to ensure that positive definite matrices are retained. Zhang and Tewarson [76] perform numerical tests with fixed values of $\theta^{(k)} < 0$ and report a moderate but consistent improvement over the BFGS method. Experience of Luksan [41] supports this observation, and I also obtained encouraging results with a limited test using the formula

$$\theta^{(k)} = \max(-1, \bar{\theta}^{(k)}/2). \tag{2.9}$$

Zhang and Tewarson are also able to modify Powell's global convergence theorem [56] to allow values of $\theta^{(k)}$ in the range $(1-\nu)\bar{\theta}^{(k)} \leq \theta^{(k)} \leq 0$ where $\nu \in (0,1)$ is a fixed constant. Byrd, Liu and Nocedal [7] investigate the case $\theta^{(k)} < 0$ in some detail. They observe that if the iterates converge q-superlinearly and $\{H^{(k)}\}$ is bounded, then $\bar{\theta}^{(k)} \to -\infty$. This appears to give plenty of scope for choosing negative values of $\theta^{(k)}$. However they also show that a necessary condition for superlinear convergence is $\theta^{(k)}/\bar{\theta}^{(k)} \to 1$. These observations would admit a constant value $\theta^{(k)} = C$, but Byrd et al. also show that superlinear convergence may not occur if $C < -1$ (hence the choice in (2.9) above). Byrd and Nocedal do not suggest any obvious choice of the parameter $\theta^{(k)} < 0$, but give some limited numerical tests with one particular choice, compared against the BFGS update. A noticeable improvement over BFGS is obtained when the initial eigenvalues are large, with no overall loss of performance for small eigenvalues. Unfortunately the value of $\theta^{(k)}$ chosen requires the Hessian matrix $G^{(k)}$ to be available, so is not a practical option

for a gradient method, but it does give a further indication that negative values of $\theta^{(k)}$ are effective. Davidon's choice [15] is also tested (Davidon chooses $\theta^{(k)}$ to minimize the condition number of $H^{(k)}B^{(k+1)}$, which allows the SR1 formula to be selected in some cases where it maintains positive definiteness) but this does not show much improvement on BFGS.

Next the implementation of the BFGS method itself is considered. In early implementations $H^{(k)}$ rather than $B^{(k)}$ is stored, so that (1.6) can be written in the form

$$s^{(k)} = -H^{(k)}g^{(k)} \tag{2.10}$$

and solved in $n^2 + O(n)$ flops. Updating $H^{(k)}$ as in (2.7) takes a further $2n^2 + O(n)$ flops. Later Gill and Murray (e.g. [29]) pioneered the idea of updating Choleski factors

$$B^{(k)} = L^{(k)}{L^{(k)}}^T \tag{2.11}$$

($L^{(k)}$ is lower triangular), which also allows (1.6) to be solved in $n^2 + O(n)$ flops. Although no practical advantage has been reported, even on ill-conditoned problems (Grandinetti [32]), (2.11) does provide an assurance that $B^{(k)}$ is positive definite in the presence of round-off errors. For the BFGS formula ($\theta^{(k)} = 0$ in (2.2)) the update of $L^{(k)}$ can be computed in an attractive way. The BFGS formula can be written as

$$B^{(k+1)} = L^{(k)}\left(I - \frac{{L^{(k)}}^T\delta^{(k)}\delta^{(k)T}L^{(k)}}{\delta^{(k)T}L^{(k)}{L^{(k)}}^T\delta^{(k)}}\right){L^{(k)}}^T + \frac{\gamma^{(k)}\gamma^{(k)T}}{\delta^{(k)T}\gamma^{(k)}}. \tag{2.12}$$

Note that the vector ${L^{(k)}}^T\delta^{(k)} = -\alpha^{(k)}{L^{(k)}}^{-1}g^{(k)}$ is a by-product of the computation of $s^{(k)}$, since we assume that $s^{(k)}$ is computed using (1.6) and (2.11). To update $L^{(k)}$, an orthogonal matrix P is determined such that

$$P^T{L^{(k)}}^T\delta^{(k)} = \pm e_1\|{L^{(k)}}^T\delta^{(k)}\| \tag{2.13}$$

(e_1 denotes the first coordinate vector). P^T is calculated by applying a backward sequence of plane rotation operations to $\delta^{(k)T}L^{(k)}$ in positions $(n, n-1), (n-1, n-2), \ldots, (2, 1)$ in turn. This gives

$$B^{(k+1)} = L^{(k)}P(I - e_1e_1^T)P^T{L^{(k)}}^T + \frac{\gamma^{(k)}\gamma^{(k)T}}{\delta^{(k)T}\gamma^{(k)}}. \tag{2.14}$$

Calculating $L^{(k)}P$ by applying the same plane rotations to $L^{(k)}$ gives rise to a lower Hessenberg matrix. Then the $I - e_1e_1^T$ operation deletes column 1 of $L^{(k)}P$ which can be overwritten by the vector $\gamma^{(k)}/\sqrt{\delta^{(k)T}\gamma^{(k)}}$. Finally the resulting lower Hessenberg matrix can be restored to lower triangular form by another forward sequence of plane rotations, giving $L^{(k+1)}$. The cost of updating $L^{(k)}$ is $2n^2 + O(n)$ rots (a *rot* is defined as the computation involved in $z = c * x + s * y$ for scalar x and y). If necessary the rotations can be implemented using square-root free Givens rotations, so that the cost of updating

$L^{(k)}$ reduces to close on $2n^2$ flops, giving a total of about $3n^2$ flops per iteration, which is the same as for the update of $H^{(k)}$. In fact this figure is not quite optimal and Goldfarb [31] shows how to update $L^{(k)}$ so that the total cost is about $2.5n^2$ flops.

A plausible idea to improve quasi-Newton methods is to attempt to correct $B^{(k)}$ by scaling, rather than through the use of low rank corrections. This for example might alleviate difficulties caused by large eigenvalues in $B^{(k)}$. An idea which combines both scaling and low rank corrections (Oren and Luenberger [51]) is simply to replace $B^{(k)}$ by the scaled matrix $\tau^{(k)}B^{(k)}$ in the update formula (2.2). It can be shown that quadratic termination and conjugate gradient properties with exact line searches remain valid. Various choices for $\tau^{(k)}$ have been proposed (see also Oren and Spedicato [52], Oren [50]), but the use of

$$\tau^{(k)} = \frac{\gamma^{(k)^T}\delta^{(k)}}{\delta^{(k)^T}B^{(k)}\delta^{(k)}} = \frac{\delta^{(k)^T}\bar{G}^{(k)}\delta^{(k)}}{\delta^{(k)^T}B^{(k)}\delta^{(k)}} \tag{2.15}$$

is attractive in that the curvature of $\tau^{(k)}B^{(k)}$ along $\delta^{(k)}$ is equated to that of the average Hessian $\bar{G}^{(k)}$ in (1.24). Unfortunately practical experience (e.g. Shanno and Phua [65]) has often been disappointing. One explanation (Nocedal and Yuan [48]) is that the unit step $\alpha^{(k)} = 1$ no longer ensures that superlinear convergence takes place. Another pointer in the quadratic function/exact line search case is that the conditions (1.25) from previous iterations

$$B^{(k+1)}\delta^{(j)} = \gamma^{(j)} \qquad\qquad j = 1, 2, ..., k-1 \tag{2.16}$$

are not preserved by the scaled updates (in contrast to say BFGS). Consequently the property $B^{(n+1)} = G$ no longer holds, which suggests that the method may be less effective at approximating the Hessian for a non-quadratic problem. Thus scaling is commonly used only to scale $B^{(1)}$ after the initial line search, which can be quite effective. However there is still much interest in scaling, see for example Luksan [41].

The above discussion suggests that it would be better if scaling could be done whilst preserving (2.16), as well as (1.25). An implementation of BFGS which allows this is given by Powell [60] who works with the factors

$$H^{(k)} = Z^{(k)}Z^{(k)^T} \tag{2.17}$$

where $Z^{(k)}$ is nonsingular. The idea is readily explained as an instance of (2.12) to (2.14), but without the final sequence of forward plane rotations that keeps $L^{(k+1)}$ lower triangular. Thus $L^{(k)}$ is in general not triangular and we denote

$$Z^{(k)} = L^{(k)^{-T}}. \tag{2.18}$$

The same rotations which define the matrix P are calculated, again using the fact that $L^{(k)^T}\delta^{(k)}$ is proportional to $Z^{(k)^T}g^{(k)}$ which is available from the computation of $s^{(k)}$. In the update of $L^{(k)}$ it is possible to write

$$L^{(k+1)} = \left[\begin{array}{c|c} \dfrac{\gamma^{(k)}}{\sqrt{\delta^{(k)^T}\gamma^{(k)}}} & L^{(k)}PJ \end{array}\right] \tag{2.19}$$

where J is the matrix obtained by deleting column 1 of I, so that $L^{(k)}PJ$ expresses the operation of deleting column 1 of $L^{(k)}P$. It is readily verified that the inverse transpose of (2.19) is given by

$$Z^{(k+1)} = \left[\frac{\delta^{(k)}}{\sqrt{\delta^{(k)\,T}\gamma^{(k)}}} \;\middle|\; \left(I - \frac{\delta^{(k)}\gamma^{(k)\,T}}{\delta^{(k)\,T}\gamma^{(k)}} \right) Z^{(k)}PJ \right] . \tag{2.20}$$

Thus the corresponding update of $Z^{(k)}$ is obtained by applying the same rotations to $Z^{(k)}$, deleting column 1, multiplying by $I - \delta^{(k)}\gamma^{(k)\,T}/\delta^{(k)\,T}\gamma^{(k)}$ (a rank one change), and finally introducing $\delta^{(k)}/\sqrt{\delta^{(k)\,T}\gamma^{(k)}}$ as a new first column.

It is readily observed from (2.20) that at the start of iteration k, the contents of columns $k, k+1, ..., n$ of $Z^{(k)}$ are an orthogonal linear combination of the columns of $Z^{(1)}$, which is arbitrary, whilst 'new' information has only filtered into columns 1 to $k-1$ of $Z^{(k)}$. In fact it can be shown in the quadratic function/exact line search case that columns 1 to $k-1$ of $Z^{(k)}$ are the vectors $\delta^{(j)}/\sqrt{\delta^{(j)T}\gamma^{(j)}}$ for $j = k-1, ..., 2, 1$ (note the reverse order). This interpretation makes it clear that previous secant conditions (2.16) are preserved if scaling is applied only to columns $k, k+1, ..., n$ of $Z^{(k)}$ before applying the update (2.20). Siegel [67] describes a strategy of this type and reports results on a single test problem with increasingly ill-conditioned choices of $Z^{(1)}$, which show an improved performance over BFGS. Lalee and Nocedal [38] propose an alternative strategy, and give conditions on the scaling parameters for the algorithm to exhibit global and superlinear convergence. Such column scaling strategies seem to provide an appropriate way of implementing scaling in BFGS, albeit at the cost of doubling the storage requirement of the algorithm ($Z^{(k)}$ is square whilst $L^{(k)}$ in (2.11) is triangular). The operation count is also higher if the algorithm is implemented as indicated by (2.20). However the operation count can be improved using the idea of Siegel [65,67]. Essentially the idea is to use rotations so that $g^{(k+1)T}Z^{(k)}P$ is zero except in columns 1 and 2.

Finally in this section we examine the claims of other formulae. Recent interest has been shown in the SR1 formula (2.1) although researchers are ambivalent about its merits. It has already been observed that the update is not in the convex class, does not preserve positive definite matrices, and may break down altogether. On the other hand it has some redeeming features. It shares the invariance properties of the Broyden class, but does not need the condition $\delta^{(k)\,T}\gamma^{(k)} > 0$ to hold. This makes it suitable for situations such as trust region algorithms, partially separable optimization (see Section 4), and nonlinear programming where this condition may not be readily obtained. It also has a unique quadratic termination property, not dependent on exact line searches. If the difference vectors $\delta^{(k)}$ $k = 1, 2, ..., n$ are linearly independent then $B^{(n+1)} = G$ is obtained and the method terminates on the next Newton step. Conn, Gould and Toint [10] report good practical experience with a trust region algorithm. They also give a theorem [11] which shows that $B^{(k)} \rightarrow G^*$ if the denominator in (2.1) is not too close to zero, and the difference vectors are uniformly linearly independent. However Khalfan, Byrd and Schnabel [37] question whether the latter assumption is realistic, and provide alternative local convergence results.

Another formula that can be used when ${\delta^{(k)}}^T\gamma^{(k)} \leq 0$ is the PSB formula (see (4.2) below), the motivation for which is discussed in more detail in Section 4. Like the SR1 formula, positive definite matrices are not preserved by the PSB formula. Also a termination result does not hold in the quadratic case, although it can be shown that the error in the Hessian approximation is monotonically decreasing. Moreover, and in contrast to SR1, the update does not fail in any way. Unfortunately practical experience has been disappointing and this is usually ascribed to a lack of invariance to linear transformations such as holds for the Broyden family. There is no doubt that researchers would welcome the development of a formula which works as well as BFGS when $\bar{G}^{(k)}$ in (1.24) is positive definite, but allows indefinite matrices to arise when $\bar{G}^{(k)}$ is indefinite, and is numerically stable like the PSB update.

3. Conjugate Gradients and Extensions

The class of conjugate gradient methods (1.20) is important in that it may provide the only practicable approach for unconstrained optimization problems in which n is very large and only a few vectors can be stored. In this section different versions of the method are described, along with recent developments in convergence theory. Various ideas for extending the method are described when rather more storage is available, but not enough to implement a quasi-Newton method. Another important application of conjugate gradients is in approximately solving the Newton equations ((1.3) or (1.8)) in a line search or trust region method, and this is also considered. A summary of current recommendations is given.

The conjugate gradient (CG) method is a line search method in which directions are given by

$$\begin{aligned} s^{(1)} &= -g^{(1)} \\ s^{(k)} &= -g^{(k)} + \beta^{(k)} s^{(k-1)} \qquad \forall\, k > 1. \end{aligned} \tag{3.1}$$

This formula derives from the CG method for solving a linear system $Ax = b$ in which A is positive definite (Hestenes and Stiefel [36]). This is equivalent to minimizing the quadratic function $\frac{1}{2}x^TAx + b^Tx + c$. A method for non-quadratic optimization is then obtained by using the same parameter choice for $\beta^{(k)}$, and choosing $\alpha^{(k)}$ by a line search. This development gives rise to the *Fletcher-Reeves method* [26] in which $\beta^{(k)}$ is defined by

$$\beta^{(k)}_{FR} = \frac{{g^{(k)}}^T g^{(k)}}{g^{(k-1)T} g^{(k-1)}}. \tag{3.2}$$

The correspondence with linear conjugate gradients ensures that the method has the property of quadratic termination if exact line searches are carried out. In this case it can also be proved (whilst $g^{(k)} \neq 0$) that

$$\begin{aligned} &\{g^{(k)}\} \text{ is an orthogonal set} \\ &\{s^{(k)}\} \text{ is a } G\text{−conjugate set.} \end{aligned} \tag{3.3}$$

The use of CG methods for minimizing non-quadratic functions gives rise to a number of issues which are not easily resolved. One is the type of line search to use, and how

accurate it should be. Experimental studies indicate that a fairly accurate line search works best (say $\sigma = 0.1$ in the the strong Wolfe conditions). Another question is whether there may be an advantage in periodically resetting $s^{(k)}$ to the steepest descent direction. The termination result perhaps suggests that this should be done every n iterations. However the methods are unlikely to be used for small n since the BFGS method is usually much more efficient and robust. For large n, resetting every n iterations is largely irrelevant since that number of iterations is unlikely to be exceeded.

The most interesting question is that of which of many alternative formulae for $\beta^{(k)}$ to use. It follows directly from (3.3) that, for quadratic functions, (3.2) is equivalent to

$$\beta_{PR}^{(k)} = \frac{{g^{(k)}}^T(g^{(k)} - g^{(k-1)})}{g^{(k-1)T}g^{(k-1)}}, \tag{3.4}$$

due to Polak and Ribière [53]. Various other equivalent formulae can also be generated from the properties of quadratic functions (see [21]). We only mention one of these

$$\beta^{(k)} = \frac{{g^{(k)}}^T(g^{(k)} - g^{(k-1)})}{s^{(k-1)T}\gamma^{(k-1)}}, \tag{3.5}$$

corresponding to a search direction (on iteration $k+1$)

$$s^{(k+1)} = -\left(I - \frac{\delta^{(k)}{\gamma^{(k)}}^T}{{\delta^{(k)}}^T\gamma^{(k)}}\right) g^{(k+1)}. \tag{3.6}$$

We draw some parallels using this formula later in the section.

In practice the Fletcher-Reeves method can be implemented with three n-vectors of storage, whilst the Polak-Ribière method requires four. However experimental studies favour the Polak-Ribière method. We now look at what is known theoretically about these two methods. An important result due to Al-Baali [1] is that if the Fletcher-Reeves method is implemented in conjunction with the strong Wolfe conditions, with $\sigma < \frac{1}{2}$, then the search directions satisfy a sufficient slope property of the form

$$-{g^{(k)}}^T s^{(k)} \geq c\|g^{(k)}\|^2 \tag{3.7}$$

where $c > 0$ is a constant independent of k. This result is established by an elegant inductive argument. Al-Baali uses (3.7) to deduce (1.15) and hence proves global convergence of the Fletcher-Reeves algorithm. This result can be extended to any method for which

$$|\beta^{(k)}| \leq \beta_{FR}^{(k)} \tag{3.8}$$

(Gilbert and Nocedal [28]) and this result is tight in a certain sense.

Practical experience with the Fletcher-Reeves method shows a propensity to take short steps remote from the solution, leading to slow convergence and even stalling (despite the global convergence proof). Powell [57] offers a possible explanation for this, and suggests

why the Polak-Ribière method may not suffer from this disadvantage. In this case, if $g^{(k)} \approx g^{(k-1)}$, then $\beta_{PR}^{(k)} \approx 0$ and the Polak-Ribière method tends to reset $s^{(k)} \approx -g^{(k)}$ to the steepest descent direction. Powell also shows that the Fletcher-Reeves direction can be much worse than the steepest descent direction, thus indicating a preference for the Polak-Ribière method. This argument would lead us to expect a global convergence result for the Polak-Ribière method. Powell also shows that this can be proved if the assumption is made that the steps $\|d^{(k)}\|$ converge to zero. However this assumption cannot be removed (when an exact line search is used) as Powell [58] shows in a later paper. He constructs a remarkable counter-example with $n = 3$ in which the sequence $\{x^{(k)}\}$ is bounded and has six accumulation points, none of which is a stationary point.

As yet however I do not think we have a really convincing explanation of why the Polak-Ribière method is more successful than Fletcher-Reeves. Since we know that the steepest descent method is often very inefficient, it is not clear that the ability to reset $s^{(k)} = -g^{(k)}$ will necessarily result in an effective method. My feeling is that the Polak-Ribière formula succeeds because it is closely related to (3.5), and (3.5) is a formula which forces conjugacy to the previous direction for non-quadratic functions (in the sense that ${s^{(k)}}^T \bar{G}^{(k)} s^{(k+1)} = {\gamma^{(k)}}^T s^{(k+1)}/\alpha^{(k)} = 0$ from (3.6)). This local conjugacy property seems to be an important factor that has largely been ignored in discussing the effectiveness of CG methods.

The counter-example of Powell [58] requires that $\beta^{(k)} < 0$ occurs infinitely, leading Powell to suggest a modification

$$\beta^{(k)} = \max(\beta_{PR}^{(k)}, 0) \tag{3.9}$$

of the Polak-Ribière formula. Gilbert and Nocedal [28] show that this leads to a globally convergent method if (a) strong Wolfe conditions are used, and (b) the strong descent condition (3.7) holds. Gilbert and Nocedal show that, because $\beta^{(k)} \geq 0$, it is always possible to design a line search to satisfy both (a) and (b).

A feature of a number of papers has been to restart the CG sequence when a significant deviation from the quadratic model is detected. Powell [57] uses the test

$$|{g^{(k)}}^T g^{(k-1)}| \geq 0.2\|g^{(k)}\|^2 \tag{3.10}$$

as an indication to restart. He also uses the three term recurrence of Beale [4] in order to avoid restarting with a steepest descent search. Shanno and Phua [65] and Buckley and LeNir [6] also suggest algorithms based on the use of restarts. Algorithms such as these have proved more effective than the simple Polak-Ribière method, albeit at the cost of requiring more vectors of storage. However there are other ways in which extra storage can usefully be used, which have given rise to even more effective algorithms. There is also a worry with restart algorithms that restarts may be triggered too often, therefore degrading the overall efficiency of the method.

We now consider how fast is the local convergence of CG methods. For exact line searches Crowder and Wolfe [13] show that the order of convergence is linear and cannot be Q-superlinear. This is not encouraging, and there are other results that also do not give

cause for hope. Yet the Polak-Ribière method can be spectacularly successful for certain large problems. The following argument in the quadratic case provides some explanation. It is possible to deduce from (3.1) that

$$g^{(k+1)} = P_k(G)g^{(1)} \tag{3.11}$$

where P_k is a polynomial of degree k with $P_k(0) = 1$. If G has eigenvalues λ_j and orthonormal eigenvectors ξ_j, then $g^{(1)}$ can be expanded as

$$g^{(1)} = \sum_{j=1}^{n} \rho_j \xi_j \tag{3.12}$$

where $\rho_j = \xi_j^T g^{(1)}$. It follows from (3.11) that

$$g^{(k+1)} = \sum_{j=1}^{n} \rho_j P_k(\lambda_j)\xi_j. \tag{3.13}$$

A term in (3.13) can be small (i.e. close to zero) either if ρ_j is small or if $P_k(\lambda_j)$ is small. In regard to the latter case, if the number of distinct eigenvalues of G is k then it is a property of the polynomial P_k that $P_k(\lambda_j) = 0$ for $j = 1, 2, ..., n$. Thus $g^{(k+1)}$ can be close to zero if the problem has certain symmetries such that either many of the ρ_j are near-zero, or there are many multiple or near-multiple eigenvalues in G so that $P_k(\lambda_j) \approx 0$ for $k \ll n$. Under these circumstances the CG method will approximately solve the quadratic problem in at most $k \ll n$ iterations.

A related method to (3.1) is the *preconditioned conjugate gradient (PCG) method* defined by

$$\begin{aligned} s^{(1)} &= -Hg^{(1)} \\ s^{(k)} &= -Hg^{(k)} + \beta^{(k)} s^{(k-1)} \qquad \forall\, k > 1 \end{aligned} \tag{3.14}$$

where $\beta^{(k)}$ is given either by

$$\beta^{(k)} = \frac{{g^{(k)}}^T H g^{(k)}}{{g^{(k-1)}}^T H g^{(k-1)}} \tag{3.15}$$

(c.f. (3.2)) or by a similar formula derived from (3.4) for example. Here H is a *fixed* symmetric positive definite matrix, and the PCG method is equivalent (see [21]) to the CG method applied in a transformed coordinate system ($H = I$ gives the CG method). For quadratic functions and exact line searches the BFGS method with $H^{(1)} = H$ generates the same sequences $\{x^{(k)}\}$ and $\{s^{(k)}\}$ as the PCG method. If $H = G^{-1}$ then the PCG method terminates in one iteration. Thus the rationale of the PCG method is to select a readily available matrix H which is a good approximation to G^{-1} in the sense that $H^{\frac{1}{2}} G H^{\frac{1}{2}}$ has many near-multiple eigenvalues. Then by virtue of the argument of the previous paragraph applied to the transformed problem, there is the possibility of obtaining rapid convergence.

We are now in a position to consider the case in which substantially more storage is available than the $4n$ locations needed for the Polak-Ribière method, but less than the

$\frac{1}{2}n^2 + O(n)$ locations for BFGS. Methods applicable to this case are referred to as *limited memory methods*. Various such methods have been suggested but there are two early methods worthy of note. In the method of Nocedal [47], a parameter m ($m \ll n$) is selected depending on the storage available ($(2m+4)n + O(m)$ locations are required). The most recent m difference pairs $\delta^{(i)}$, $\gamma^{(i)}$, $i = k-m, ..., k-2, k-1$ are stored, and a matrix $H^{(k)}$ is defined implicitly by applying the BFGS formula to a given diagonal matrix D for each difference pair in turn. Nocedal gives a recurrence relation derived from (2.7) that enables $s^{(k)}$ in (2.10) to be computed in $4nm + O(n)$ flops without calculating $H^{(k)}$. Usually D is chosen as a multiple of the unit matrix, using for example (2.15). After each iteration (once $k > m$) the oldest difference pair is deleted from storage and the new difference pair is introduced. In practice a suitable choice for m has been found to be $m = 5$ and it is interesting and somewhat surprising that larger values of m do not seem to give significant increases in performance, even on very large problems. Another interesting observation is that in the case $m = 1$, if the line search is exact, then it follows from (2.7) that the direction $s^{(k+1)}$ for Nocedal's method is the same as that for the CG method (3.6), so that these methods are equivalent.

Another limited memory method is that of Buckley and LeNir [6]. This is a two-phase method, starting with the *QN-phase* in which for $k = 1, 2, ..., m$, $H^{(k)}$ is built up as in Nocedal's method, and the available storage is filled. The *CG-phase* follows, in which for $k = m+1, m+2, ..., H^{(m)}$ is used as the fixed matrix H in the PCG method. The CG phase is terminated using the test (3.10). All the difference pairs except for $\delta^{(k-1)}$, $\gamma^{(k-1)}$ are then deleted and the QN-phase is re-entered. Both methods are effective, but Liu and Nocedal [40] show a preference for Nocedal's method, based on a wide selection of large dimension problems. This may be an indication that restarting the Buckley and LeNir method by deleting most of the current difference pairs is a source of inefficiency.

For this reason there is considerable interest in limited memory methods which specify $H^{(k)}$ on the basis of the most recent m difference pairs, which we may express as columns of the matrices

$$\Delta^{(k)} = \left[\delta^{(k-m)}, ..., \delta^{(k-2)}, \delta^{(k-1)}\right] \tag{3.16}$$

and

$$\Gamma^{(k)} = \left[\gamma^{(k-m)}, ..., \gamma^{(k-2)}, \gamma^{(k-1)}\right]. \tag{3.17}$$

Nocedal's recurrence may be regarded as providing a useful approximate solution to the problem of finding a symmetric positive definite matrix $H^{(k)}$ which satisfies the *multiple secant equation*

$$H^{(k)}\Gamma^{(k)} = \Delta^{(k)}. \tag{3.18}$$

This problem also arises when finding the compliance matrix of an elastic structure, given measurements of loading and displacement. The matrix $M^{(k)} = {\Gamma^{(k)}}^T\Delta^{(k)}$ plays an important role. First we consider the case that $M^{(k)}$ is symmetric and positive definite, which arises either if $m = 1$ or if the differences are sampled from a quadratic function and $\text{rank}(\Delta^{(k)}) = m$. In this case exact solutions to (3.18) exist, and we can write down analogues of all the common update formulae in an obvious way. For example the *Multiple*

BFGS formula (applied to a matrix D) can be defined by analogy with (2.7) as

$$H^{(k)} = \left(I - \Delta^{(k)} {M^{(k)}}^{-1} {\Gamma^{(k)}}^T\right) D \left(I - \Gamma^{(k)} {M^{(k)}}^{-1} {\Delta^{(k)}}^T\right) + \Delta^{(k)} {M^{(k)}}^{-1} {\Delta^{(k)}}^T. \quad (3.19)$$

In general this gives a different matrix $H^{(k)}$ to that computed by Nocedal's recurrence, although the amount of computation is the same. However it is interesting to observe that if the columns of $\Delta^{(k)}$ are G-conjugate, then the same matrix $H^{(k)}$ is obtained. Unfortunately, for nonquadratic functions with $m > 1$, $M^{(k)}$ is almost always nonsymmetric. Schnabel [62] considers some ways in which $\Delta^{(k)}$ and $\Gamma^{(k)}$ might be modified so that $M^{(k)}$ is symmetric and positive definite, but does not strongly support any particular scheme. Allwright [1] considers finding a symmetric positive semi-definite matrix $H^{(k)}$ that best solves (3.18). However this matrix might be singular so there is also doubt about its merit. Given that Nocedal's recursion implicitly defines a symmetric positive definite $H^{(k)}$ without any restriction other than ${\delta^{(k)}}^T \gamma^{(k)} > 0$, it is not likely to be improved upon significantly.

Some recent work has looked at the possibility of limited memory algorithms which store only one $n \times m$ matrix and not two as in (3.16) and (3.17). Fletcher [22] describes a method in which $B^{(k)}$ is represented as

$$B^{(k)} = \left[S \middle| S^-\right] \begin{bmatrix} S^T \\ S^{-T} \end{bmatrix} \quad (3.20)$$

(omitting superscript k), where $S \in \mathbb{R}^{n \times k}$ and columns of S^- form an orthonormal basis for the null space of S. The BFGS method (with $B^{(1)} = I$) can then be expressed as follows. Initially $S^{(1)} = g^{(1)}/\|g^{(1)}\|$ and iteration k takes the form

(i) calculate search direction $s := (SS^T)^+ g$
(ii) line search to give $x^{(k+1)}$ and $g^{(k+1)}$
(iii) calculate $g^- := S^- S^{-T} g^{(k+1)}$ (3.21)
(iv) if $g^- \neq 0$ extend $\left[S \middle| g^-/\|g^-\|\right] =: \hat{S}$ else $\hat{S} := S$
(v) update $\hat{S}$ to give $S^{(k+1)}$ as for the BFGS method.

The algorithm is implemented by storing QL factors of S so that s and g^- are readily computed (using $(SS^T)^+ = QL^{-T}L^{-1}Q^T$ and $S^-S^{-T} = I - QQ^T$). Siegel [68] follows a similar approach and points out that algorithm (3.21) can be interpreted as one which updates a reduced Hessian $\hat{B} = Q^T B^{(k)} Q$. An attractive way to implement this is to update Choleski factors $\hat{B} = LL^T$ of the reduced Hessian matrix using reduced difference vectors, in much the same way as for (2.14). This indicates how to update S in step (v) of (3.21). Siegel injects another good idea, which is that Q should not be stored directly: rather Δ (c.f. (3.16)) should be stored and Q determined implicitly from QR factors of Δ. This enables the dominant computational cost to be reduced to $2nk$ flops per iteration, where k refers to the number of columns in Δ. (Using k also to denote the iteration number assumes that the *else* option in step (iv) is not activated.)

Algorithm (3.21) is exactly equivalent to BFGS and can be continued until the available nm storage locations (storing either S or Q or Δ) are filled. At this stage it is necessary

to deviate from the BFGS algorithm. Fletcher proceeds in the same fashion as Buckley and LeNir, using the matrix $H = B^{(m)-1}$ as a fixed preconditioner in PCG. Once test (3.10) fails then all the stored information is discarded and the algorithm is restarted. Siegel proceeds differently when the storage is filled and throws away only the oldest column of Δ. However it can be shown that both these algorithms retain the quadratic termination property. I have a student following up these ideas (Lefebure [39]) and hope to report in more detail at a later date.

Finally mention should be made of the *Truncated Newton* or *Truncated conjugate gradient* method. This type of method has an outer iteration in which the Newton equation (1.3) is solved approximately by applying an inner linear conjugate gradient iteration (Toint [73], Steihaug [70]). This cuts down the computational effort for solving (1.3), whilst rapid convergence is still possible if the accuracy requirement is made more severe as the outer iteration count increases (Dembo, Eisenstat and Steihaug [16]). Both trust region and line search approaches are possible. To make the methods efficient, a PCG inner iteration should be used, e.g. Nash [45], although the 'best' preconditioner is likely to be problem dependent. For the method to be applicable as a gradient method, vectors of the form $G^{(k)}p$ required in the inner PCG iteration must be estimated, and this may be done using the finite difference quotient

$$G^{(k)}p \approx \frac{g(x^{(k)} + hp) - g^{(k)}}{h} \tag{3.22}$$

where h is a small stepsize (O'Leary [49]). This does require an additional gradient evaluation on every inner PCG iteration. As more inner iterations are usually required as the outer iteration counter k increases, so the cost of each outer iteration also increases. A comparison of Truncated Newton and Nocedal's limited memory method is given by Nash and Nocedal [46]. Both methods are effective and neither is the outright winner. The tentative conclusion is that Truncated Newton is better on nearly quadratic problems, whilst Nocedal's method is better on highly nonlinear problems.

4. Structured Large Scale Optimization

The methods described in the previous section are most suitable for *unstructured* large scale optimization. Frequently however a large scale optimization problem has some structure that can be exploited, such as might be reflected in the Hessian matrix being sparse. In this case, and when second derivatives are available, there is little doubt that Newton's method (suitably modified in a line search or trust region method as described in Section 1) is a very attractive method which takes advantage of structure. Providing the organizational problems can be handled, this must be the method of choice. The increasing availability of automatic differentiation packages reinforces this conclusion. The only negative factor remains the need to account for an indefinite Hessian. However as we shall see below, the same is true for most other alternatives.

The rest of this section concerns the case that a gradient method is required. First of all it should be realized that use of a finite difference approximation to the Hessian becomes much more efficient in the sparse case. It is possible by a careful choice of

directions $d^{(i)}$, $i = 1, 2, ..., p$ to use the vectors

$$y^{(i)} = (g(x^{(k)} + hd^{(i)}) - g^{(k)})/h \approx G^{(k)}d^{(i)} \tag{4.1}$$

to estimate the non-sparse elements of $G^{(k)}$. For example if $G^{(k)}$ is a band matrix with p bands on or below the diagonal ($p = 2$ for a tridiagonal matrix) then only the p directions

$$d^{(i)} = (e_i + e_{i+p} + e_{i+2p} + ...) \qquad\qquad i = 1, 2, ..., p \tag{4.2}$$

are required. (Here e_i denotes the unit vector that is column i of I.) Powell and Toint [61] give a algorithm that extends this idea to general sparsity patterns. Moreover advantage can be taken of known fixed but non-zero elements in $G^{(k)}$ to further reduce the amount of differencing required.

We now go on to consider the possibilities for quasi-Newton methods that respect sparsity. First we digress to point out that many updating formulae possess a *variational property* from which they can be derived. A significant result due to Goldfarb [30] (based on the original idea due to Greenstadt [33]) is that the correction $E = H^{(k+1)} - H^{(k)}$ in the BFGS formula satisfies a minimum property with respect to a weighted Frobenius norm of the form

$$\|E\|_W^2 = (\text{trace}(EWEW))^{1/2} \tag{4.3}$$

where $W > 0$ and $W\delta^{(k)} = \gamma^{(k)}$. This result can be interpreted as showing that $H^{(k)}$ is changed by the minimum amount (in the sense of (4.3)) required to satisfy (1.25) and symmetry. This ensures that previous information accumulated in $B^{(k)}$ is disturbed as little as possible. The well-known DFP formula can also be interpreted in a similar way. Another formula that satisfies a minimum correction property is the PSB formula (Powell [54])

$$B^{(k+1)} = B^{(k)} + \frac{\eta\delta^T + \delta\eta^T}{\delta^T\delta} - \frac{\eta^T\delta}{\delta^T\delta^2}\delta\delta^T \tag{4.4}$$

where $\eta = \gamma - B^{(k)}\delta$ and where δ and γ denote $\delta^{(k)}$ and $\gamma^{(k)}$ respectively. In this case it is the Frobenius norm ($W = I$ in (4.3)) of the correction to $B^{(k)}$ that is minimized (subject to (1.25) and symmetry). Unfortunately the PSB update does not in general preserve $B^{(k)} > 0$ and practical experience has been disappointing as discussed in Section 2.

Quasi-Newton methods become less attractive when n is very large because of the storage and computational requirements associated with large dense matrices. However if the Hessian is a sparse matrix it is attractive to look for updating formulae which preserve the same sparsity in $B^{(k)}$. Thus we express the sparsity conditions on B as

$$B_{ij} = 0 \quad \forall\,(i,j) \in \mathcal{S} \tag{4.5}$$

where $\mathcal{S}$ is a set of pairs of integers in the range $[1 : n]$. Because of symmetry it is assumed that $(i,j) \in \mathcal{S}$ if and only if $(j,i) \in \mathcal{S}$. It is also assumed that $G(x)$ satisfies (4.5) for all $x \in \mathbb{R}^n$. The complementary set of index pairs not in $\mathcal{S}$ is denoted by $\mathcal{S}^-$. Because we are concerned with positive definite matrices it is assumed that

$$(i,i) \in \mathcal{S}^- \quad i = 1, 2, ..., n. \tag{4.6}$$

For such problems it is fruitful to determine a minimum correction update formula which is constrained by (4.5) in addition to the quasi-Newton condition and symmetry. In a seminal paper Toint [71] shows that it is reasonably straightforward to compute a minimum correction to $B^{(k)}$ in the Frobenius norm subject to these conditions. To present Toint's update we define the projection operator $\mathcal{G}(M) : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ by

$$\mathcal{G}(M)_{ij} = \begin{cases} 0 & (i,j) \in \mathcal{S} \\ M_{ij} & (i,j) \in \mathcal{S}^- \end{cases} \tag{4.7}$$

This has been colourfully dubbed the *gangster operator* since it shoots holes in M according to the sparsity pattern defined by (4.5). It is convenient to introduce the notation

$$\delta_{[i]} = \mathcal{G}(\delta e_i^T)e_i \tag{4.8}$$

in which the sparsity pattern of column i of the Hessian is imposed on δ. Toint shows that the resulting minimum correction satisfies

$$B^{(k+1)} = B^{(k)} + \mathcal{G}(\delta\lambda^T + \lambda\delta^T) \tag{4.9}$$

where $\lambda \in \mathbb{R}^n$ is obtained by solving the linear system

$$Q\lambda = r \tag{4.10}$$

in which $r = \gamma - B^{(k)}\delta$. It follows from (4.9) and (1.25) that column i of the matrix Q is defined by

$$Qe_i = \delta_i\delta_{[i]} + \delta_{[i]}^T\delta_{[i]}e_i. \tag{4.11}$$

It is easily shown that Q is symmetric positive semi-definite and that $\mathcal{G}(Q) = Q$ (i.e. Q satisfies the sparsity conditions (4.5)). In addition Q is positive definite if

$$\delta_{[i]} \neq 0 \quad i = 1, 2, ..., n \tag{4.12}$$

in which case sparse LDL^T factors of Q are calculated and (4.10) can readily be solved to obtain λ. If Q is singular then $\delta_{[i]} = 0$ for some i. However it then follows from (1.23) that $\gamma_i = 0$ and hence $r_i = 0$. Thus (4.10) is consistent and can be solved by deleting row and column i from Q (ignoring the effects of round-off error).

As with the dense PSB update, the condition $B^{(k)} > 0$ is not preserved by this update, and likewise practical performance has not been outstanding. In view of this it is natural to enquire what happens when the sparsity conditions (4.5) are included in the minimum correction property that defines the BFGS or DFP formula. Unfortunately as Toint [71] points out the use of a weighted Frobenius norm leads to formulae that are intractable in both cases (see also [72] for more details). However Toint [72] proves that if

$$\delta_i \neq 0 \quad i = 1, 2, ..., n, \tag{4.13}$$

if G is irreducible, and if $\delta^T\gamma > 0$ then there does exist a symmetric update which preserves positive definiteness. The condition $\delta^T\gamma > 0$ is clearly required, else (2.3) would imply

$\delta^T B\delta \leq 0$, contradicting $B > 0$. The assumption of irreducibility (that is G cannot be reduced to block diagonal form by a symmetric permutation) is not a serious restriction because if G is reducible then (1.1) can be decomposed into two or more problems which can be solved separately. It is assumed throughout what follows that G is irreducible. On the other hand, (4.13) is critical and Sorensen [69] shows that a positive definite update may not exist if $\delta_i = 0$ for some i, and that serious growth in B can occur in a neighbourhood of this situation. We return to these points below.

More recently an alternative approach to an optimal BFGS-like formula that preserves sparsity and positive definiteness has been described by Fletcher [24]. This arises from an observation of Fletcher [23] that the BFGS and DFP formulae can be derived by a variational argument using the measure function

$$\psi(A) = \text{trace}(A) - \ln\det(A) \tag{4.14}$$

that is introduced by Byrd and Nocedal [8] in the convergence analysis of quasi-Newton methods. This function is strictly convex on the set of positive definite matrices and is minimized by $A = I$. The function becomes unbounded as A becomes singular or infinite and so acts as a barrier function that keeps A positive definite. A suitable variational property is to minimize $\psi(H^{(k)}B)$ since in the absence of any constraints the solution is just $B = H^{(k)-1}$. Introducing the constraints of symmetry and (1.25) leads to an update in which $H = B^{-1}$ stays close to $H^{(k)}$ in this sense. Fletcher shows that it is the BFGS formula that solves this variational problem.

A theorem extending this result to include the sparsity conditions (4.5) is set out in [24]. It is shown that the solution is characterized by the existence of $\lambda \in \mathbb{R}^n$ such that

$$\mathcal{G}(H) = \mathcal{G}(H^{(k)} + \lambda\delta^T + \delta\lambda^T). \tag{4.15}$$

In the dense case it is straightforward to find λ such that H satisfies $H\gamma = \delta$ (that is (1.25)). In the sparse case no direct solution for λ appears to be possible but Fletcher shows how an iterative solution can be determined. Given a vector λ, the sequence of operations

- Calculate $\mathcal{G}(H)$ from (4.15)
- Find $B > 0$ such that $\mathcal{G}(B^{-1}) = \mathcal{G}(H)$. (4.16)
- Calculate $r := B\delta - \gamma$.

defines r as a function of λ ($\mathbb{R}^n \to \mathbb{R}^n$) and the update is determined by finding λ such that

$$r(\lambda) = 0, \tag{4.17}$$

which is a nonlinear system of n equations in n variables. Moreover it can be shown that $r(\lambda)$ is the gradient of a concave programming problem derived by using the Wolfe dual and this enables the solution of $r(\lambda) = 0$ to be undertaken in a reliable way. The Jacobian $Q(\lambda)$ of this system is important and plays a similar role to the matrix Q that arises in the linear system (4.10) in Toint's sparse PSB update. If a certain assumption related to fill-in is valid, the matrix Q satisfies the same structural and definiteness conditions as for Toint's matrix. Thus the nonlinear system can be solved by a few iterations of analogous

complexity to (4.10). Fletcher also suggests alternative possibilities for computing the update via what might be thought of as primal methods. The dual update has been implemented for tridiagonal systems and some numerical experiments show a significant reduction in the number of quasi-Newton iterations, as compared with limited memory, conjugate gradient, and dense BFGS codes. However the amount of computation required to compute the update is significantly greater than for the dense case.

A question of some interest is whether the information present in $\mathcal{G}(H)$ does indeed determine $B > 0$ (as required by (4.16)), and whether the outcome is unique. It is hopeful that $\mathcal{G}(H)$ contains the same number of non-zero elements as B. If the elements of $\mathcal{S}^-$ are chosen to include elements that fill-in when the LDL^T factors of B are calculated, then it is also shown in [24] that B is well-determined by $\mathcal{G}(H)$ and an algorithm is given for computing the LDL^T factors of B. This algorithm is shown to be particularly efficient when the sparsity pattern of B is formed from dense overlapping blocks on the diagonal. One feature of these results is particularly surprising. If B is an irreducible sparse matrix then $H = B^{-1}$ is generally dense. It is therefore most unexpected to find that it is only the elements of $\mathcal{G}(H)$ in (4.15) that determine the sparsity pattern of B.

Assumption (4.13) is critical for the existence of a positive definite sparse update. This is made clear by Sorensen [69] who essentially cites the following example in which B is required to solve

$$\begin{bmatrix} a & b & \\ b & c & * \\ & * & * \end{bmatrix} \begin{pmatrix} -1 \\ \varepsilon \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 2 \end{pmatrix}. \tag{4.18}$$

When $\varepsilon \neq 0$ the first equation implies $b = (a+1)/\varepsilon$, and $a > 0$ is required for positive definiteness. Thus b grows without limit as ε goes to zero. Moreover the inequality $ac > b^2$ implies that ac increases like ε^{-2} so the rate of growth is quadratic. If $\varepsilon = 0$ then the only solution has $a = -1$ and a positive definite update does not exist. Yet $\delta^T\gamma = 1 > 0$.

It is important to realise that such examples can only occur when the average Hessian matrix $\bar{G}$ in (1.24) is indefinite. Even if $\bar{G}$ is indefinite then a satisfactory update may yet be obtained (this happens for example in the dense case). Thus it may be that these difficulties arise relatively infrequently and an ad-hoc solution may be adequate. However when difficulties do arise then their effects are severe due to the ε^{-2} growth. Generally speaking, we have a choice to relax either the positive definite condition or the secant condition if we wish to retain sparsity. In the context of Fletcher's sparse update, a simple solution is to terminate the dual iteration if some preset upper limit on the size of B and H is exceeded. In fact the high cost of Fletcher's update may make it necessary to take only a small number of dual iterations, so that the secant equation is never satisfied exactly. It is difficult to say whether this will adversely affect the convergence of the outer quasi-Newton iteration.

Other research into structured unconstrained optimization has avoided the requirement that $B^{(k)}$ should be positive definite. Most promising has been the approach of Griewank and Toint (e.g. [34]) based on the *partially separable* optimization problem

$$\text{minimize} \quad f(x) = \sum_{j=1}^{n_e} f_j(x_{[j]}) \tag{4.19}$$

in which there are n_e *nonlinear element functions* $f_j(x_{[j]})$, each of which depends on a usually small subset $x_{[j]}$ of the components of x. Then the Hessian of $f(x)$ can be decomposed into a sum of Hessians of the $f_j(x_{[j]})$, the nontrivial submatrices of which can be treated as dense matrices. Similar remarks apply for the gradient vector. Griewank and Toint suggest that these element Hessian submatrices are approximated by the use of dense updating techniques. There are however some difficulties that have to be overcome. The element Hessian submatrices may not be positive definite so it is not possible to rely on the analogue of the condition $\delta^T\gamma > 0$ holding for each submatrix. Thus the BFGS formula cannot be used, and Toint uses the symmetric rank 1 formula in the Harwell Subroutine Library code VE08. Consequently the overall Hessian approximation obtained by summing the submatrix approximations is also not in general positive definite. Also the possibility of zero in the denominator of the updates has to be allowed for.

A more flexible format has recently been used by Conn, Gould and Toint (e.g. [12]) in the LANCELOT code. Here the *group partially separable function*

$$f(x) = \sum_{j=1}^{n_g} g_i(\alpha_i(x)) \tag{4.20}$$

is defined in which there are n_g *group functions* $g_i \in \mathrm{C}^2 : \mathbb{R} \to \mathbb{R}$. The argument of each group function is a weighted sum

$$\alpha_i(x) = \sum_{j \in \mathcal{J}_i} w_{ij} f_j(x_{[j]}) + a_i^T x - b_i \tag{4.21}$$

over a subset $\mathcal{J}_i$ of the nonlinear element functions, together with a linear term. It is envisaged that the vectors a_i in the linear term should usually be sparse.

It is difficult to predict whether such a format will become generally accepted. On the one hand it is very flexible and most large scale problems can be effectively expressed in this way. Numerical experience with the LANCELOT code on unconstrained problems is very satisfactory and often approaches the performance of Newton's method. This suggests that the difficulties of working with indefinite Hessians and updating using the SR1 formula have largely been overcome. On the other hand the organizational problems are more severe than those required merely to define the sparsity conditions in (4.6), and this may particularly deter the non-specialist user. It may be that there is still some merit in developing an effective positive definite sparse update, should this be possible.

5. References

[1] Al-Baali M. (1985) Descent property and global convergence of the Fletcher-Reeves method with inexact line search, *J. Inst. Maths. Applns.*, **5**, 121-124.

[2] Allwright J.C. (1988) Positive semi-definite matrices: Characterization via conical hulls and least-squares solution of a matrix equation, *SIAM J. Control Optim.*, **26**, 537-556 (see also errata in *SIAM J. Control Optim.*, **28**, (1990), 250-251).

[3] Barzilai J. and Borwein J.M. (1988) Two-point step size gradient methods, *IMA J. Numer. Anal.*, **8**, 141-148.

[4] Beale E.M.L. (1972) A derivation of conjugate gradients, in *Numerical Methods for Nonlinear Optimization*, ed. F.A.Lootsma, Academic Press, London.

[5] Broyden C.G. (1967) Quasi-Newton methods and their application to function minimization, *Maths. of Computation*, **21**, 368-381.

[6] Buckley A. and LeNir A. (1983) QN-like variable storage conjugate gradients, *Math. Programming*, **27**, 155-175.

[7] Byrd R.H., Liu D.C. and Nocedal J. (1990) On the behavior of Broyden's class of quasi-Newton methods, Report NAM-01, Northwestern Univ., Dept. EECS.

[8] Byrd R.H. and Nocedal J. (1989) A tool for the analysis of quasi-Newton methods with application to unconstrained minimization, *SIAM J. Numer. Anal.*, **26**, 727-739.

[9] Byrd R.H., Nocedal J. and Yuan Y. (1987) Global convergence of a class of quasi-Newton methods on convex problems, *SIAM J. Numer. Anal.*, **24**, 1171-1190.

[10] Conn A.R., Gould N.I.M. and Toint Ph.L. (1988) Testing a class of methods for solving minimization problems with simple bounds on the variables, *Maths. of Computation*, **50**, 399-430.

[11] Conn A.R., Gould N.I.M. and Toint Ph.L. (1991) Convergence of quasi-Newton matrices generated by the symmetric rank one update, *Math. Programming*, **50**, 177-195.

[12] Conn A.R., Gould N.I.M. and Toint Ph.L. (1992) *LANCELOT: a Fortran package for large-scale nonlinear optimization (Release A)*, Springer Series in Computational Mathematics 17, Springer-Verlag, Heidelberg.

[13] Crowder H.P. and Wolfe P. (1972) Linear convergence of the conjugate gradient method, *IBM J. Res. and Dev.*, **16**, 431-433.

[14] Davidon W.C. (1959) Variable metric methods for minimization, Argonne Nat. Lab. Report ANL-5990 (rev.), and in *SIAM J. Optimization*, **1**, (1991), 1-17.

[15] Davidon W.C. (1975) Optimally conditioned optimization algorithms without line searches, *Math. Programming*, **9**, 1-30.

[16] Dembo R.S., Eisenstat S.C. and Steihaug T. (1982) Inexact Newton methods, *SIAM J. Numer. Anal.*, **19**, 400-408.

[17] Dennis J.E. and Moré J.J. (1974) A characterization of superlinear convergence and its application to quasi-Newton methods, *Maths. of Computation*, **28**, 549-560.

[18] Dennis J.E. and Schnabel R.B. (1983) *Numerical Methods for Unconstrained Optimization ana Nonlinear Equations*, Prentice-Hall Inc., Englewood Cliffs, NJ.

[19] Dixon L.C.W. (1972) Quasi-Newton algorithms generate identical points, *Math. Programming*, **2**, 383-387.

[20] Fletcher R. (1970) A new approach for variable metric algorithms, *Computer J.*, **13**, 317-322.

[21] Fletcher R. (1987) *Practical Methods of Optimization, (2nd. Edition)*, John Wiley, Chichester.

[22] Fletcher R. (1990) Low storage methods for unconstrained optimization, in *Computational Solution of Nonlinear Systems of Equations*, eds. E.L.Allgower and K.Georg, Lectures in Applied Mathematics Vol. 26, AMS Publications, Providence, RI.

[23] Fletcher R. (1991) A new result for quasi-Newton formulae, *SIAM J. Optimization*, **1**, 18-21.

[24] Fletcher R. (1992) An optimal positive definite update for sparse Hessian matrices, Report NA/145, Univ. of Dundee, to appear in *SIAM J. Optimization*.

[25] Fletcher R. and Powell M.J.D. (1963) A rapidly convergent descent method for minimization, *Computer J.*, **3**, 163-168.

[26] Fletcher R. and Reeves C.M. (1964) Function minimization by conjugate gradients, *Computer J.*, **7**, 149-154.

[27] Forsgren A., Gill P.E. and Murray W. (1993) Computing modified Newton directions using a partial Cholesky factorization, Royal Inst. of Tech., Sweden, Dept. of Math. Report TRITA/MAT-93-09.

[28] Gilbert J.C. and Nocedal J. (1990) Global convergence properties of conjugate gradient methods for optimization, Report 1268, INRIA.

[29] Gill P.E. and Murray W. (1972) Quasi-Newton methods for unconstrained optimization, *J. Inst. Maths. Applns.*, **9**, 91-108.

[30] Goldfarb D. (1970) A family of variable metric methods derived by variational means, *Maths. of Computation*, **24**, 23-26.

[31] Goldfarb D. (1976) Factorized variable metric methods for unconstrained optimization, *Maths. of Computation*, **30**, 796-811.

[32] Grandinetti (1979) Factorization versus nonfactorization in quasi-Newtonian algorithms for differentiable optimization, in *Methods for Operations Research*, Hain Verlay.

[33] Greenstadt J.L. (1970) Variations of variable metric methods, *Maths. of Computation*, **24**, 1-22.

[34] Griewank A. and Toint Ph.L. (1982) Partitioned variable metric updates for large structured optimization problems, *Numerische Math.*, **39**, 429-448.

[35] Grippo L., Lampariello F. and Lucidi S. (1990) A quasi-discrete Newton algorithm with a nonmonotone stabilization technique, *J. Optim. Theo. Applns.*, **64**, 485-500.

[36] Hestenes M.R. and Stiefel E. (1952) Methods of conjugate gradients for solving linear systems, *J. Res. NBS*, **49**, 409-436.

[37] Khalfan H., Byrd R.H. and Schnabel R.B. (1990) A theoretical and experimental study of the symmetric rank one update, Report CU-Cs-489-90, Univ. of Colorado at Boulder, Dept. CS.

[38] Lalee M. and Nocedal J. (1991) Automatic scaling strategies for quasi-Newton updates, Report NAM-04, Northwestern Univ., Dept. EECS.

[39] Lefebure B. (1993) Investigating new methods for large scale unconstrained optimization. M.Sc. Thesis, Univ. of Dundee, Dept. of Maths. and CS.

[40] Liu D.C. and Nocedal J. (1989) On the limited memory BFGS method for large scale optimization, *Math. Programming*, **45**, 503-528.

[41] Luksan L. (1993) Computational experience with known variable metric updates, Working paper.

[42] Moré J.J. (1983) Recent developments in algorithms and software for trust region methods, in *Mathematical Programming, The State of the Art*, eds. A.Bachem, M Grotschel and G.Korte, Springer-Verlag, Berlin.

[43] Moré J.J., Garbow B.S. and Hillstrom K.E. (1981) Testing unconstrained optimization software, *ACM Trans. Math. Softw.*, **7**, 17-41.

[44] Moré J.J. and Sorensen D.C. (1979) On the use of directions of negative curvature in a modified Newton method, *Math. Programming*, **16**, 1-20.

[45] Nash S.C. (1985) Preconditioning of truncated-Newton methods, *SIAM J. S.S.C.*, **6**, 599-618.

[46] Nash S.C. and Nocedal J. (1989) A numerical study of the limited memory BFGS method and the truncated-Newton method for large scale optimization, Report NAM-02, Northwestern Univ., Dept. EECS.

[47] Nocedal J. (1980) Updating quasi-Newton matrices with limited storage, *Maths. of Computation*, **35**, 773-782.

[48] Nocedal J. and Yuan Y. (1991) Analysis of a self-scaling quasi-Newton method, Report NAM-02, Northwestern Univ., Dept. EECS.

[49] O'Leary D.P. (1982) A discrete Newton algorithm for minimizing a function of many variables, *Math. Programming*, **23**, 20-33.

[50] Oren S.S. (1974) On the selection of parameters in self-scaling variable metric algorithms, *Math. Programming*, **7**, 351-367.

[51] Oren S.S. and Luenberger D.G (1974) Self-scaling variable metric (SSVM) algorithms: I Criteria and sufficient conditions for scaling a class of algorithms, *Management Sci.*, **20**, 845-862.

[52] Oren S.S. and Spedicato E. (1976) Optimal conditioning of self-scaling variable metric algorithms, *Math. Programming*, **10**, 70-90.

[53] Polak E. and Ribière G. (1969) Note sur la convergence de methodes de directions conjugées, *Rev. Française Informat Recherche Operationnelle, 3e Année*, **16**, 35-43.

[54] Powell M.J.D. (1970) A new algorithm for unconstrained optimization, in *Nonlinear Programming*, eds. J.B.Rosen, O.L.Mangasarian and K.Ritter, Academic Press, New York.

[55] Powell M.J.D. (1971) On the convergence of the variable metric algorithm, *J. Inst. Maths. Applns.*, **7**, 21-36.

[56] Powell M.J.D. (1976) Some global convergence properties of a variable metric algorithm for minimization without exact line searches, in *Nonlinear Programming, SIAM-AMS Proceedings Vol. IX*, eds. R.W.Cottle and C.E.Lemke, SIAM Publications, Philadelphia.

[57] Powell M.J.D. (1977) Restart procedures for the conjugate gradient method, *Math. Programming*, **12**, 241-254.

[58] Powell M.J.D. (1984) Nonconvex minimization calculations and the conjugate gradient method, in *Numerical Analysis, Dundee 1983*, ed. D.F.Griffiths, Lecture Notes in Mathematics 1066, Springer Verlag, Berlin.

[59] Powell M.J.D. (1986) How bad are the BFGS and DFP methods when the objective function is quadraticΓ, *Math. Programming*, **34**, 34-47.

[60] Powell M.J.D. (1987) Updating conjugate directions by the BFGS formula, *Math. Programming*, **38**, 29-46.

[61] Powell M.J.D. and Toint Ph. L. (1979) On the estimation of sparse Hessian matrices, *SIAM J. Numer. Anal.*, **16**, 1060-1074.

[62] Schnabel R.B. (1983) Quasi-Newton methods using multiple secant equations, Report CU-CS-247-83, Univ. of Colorado at Boulder, Dept. CS.

[63] Schnabel R.B. and Eskow E. (1990) A new modified Cholesky factorization, *SIAM J. Sci. Stat. Comp.*, **11**, 1136-1158.

[64] Shanno D.F. (1970) Conditioning of quasi-Newton methods for function minimization, *Maths. of Computation*, **24**, 647-656.

[65] Shanno D.F. and Phua K.H. (1980) Remark on algorithm 500, minimization of unconstrained multivariate functions, *ACM Trans. Math. Softw.*, **6**, 618-622.

[66] Siegel D. (1991) Updating of conjugate direction matrices using members of Broyden's family, Report DAMTP 1991/NA4, Univ. of Cambridge, Dept. AMTP.

[67] Siegel D. (1991) Modifying the BFGS update by a new column scaling technique, Report DAMTP 1991/NA5, Univ. of Cambridge, Dept. AMTP.

[68] Siegel D. (1992) Implementing and modifying Broyden class updates for large scale optimization, Report DAMTP 1992/NA12, Univ. of Cambridge, Dept. AMTP.

[69] Sorensen D.C. (1982) Collinear scaling and sequential estimation in sparse optimization algorithms, *Math. Programming Study*, **18**, 135-159.

[70] Steihaug T. (1983) The conjugate gradient method and trust regions in large scale optimization, *SIAM J. Numer. Anal.*, **20**, 626-637.

[71] Toint Ph. L. (1977) On sparse and symmetric updating subject to a linear equation. *Maths. of Computation*, **31**, 954-961.

[72] Toint Ph. L. (1981) A note on sparsity exploiting quasi-Newton methods. *Math. Programming*, **21**, 172-181.

[73] Toint Ph. L. (1981) Towards an efficient sparsity exploiting Newton method for minimization, in *Sparse Matrices and Their Uses*, ed. I.S.Duff, Academic Press, London.

[74] Torczon V. (1991) On the convergence of the multidimensional search algorithm, *SIAM J. Optimization*, **1**, 123-145.

[75] Wood G.R. (1992) The bisection method in higher dimensions, *Math. Programming*, **55**, 319-338.

[76] Zhang Y. and Tewarson R.P. (1988) Quasi-Newton algorithms with updates from the pre-convex part of Broyden's family, *IMA J. Numer. Anal.*, **8**, 487-509.