Variable metric methods of minimisation

By J. D. Pearson*

Two basic approaches to the generation of conjugate directions are considered for the problem of unconstrained minimisation of quadratic functions. The first approach results in a projected gradient algorithm which gives '*n* step' convergence for a quadratic. The second approach is based on the generalised solution of a set of underdetermined linear equations, various forms of which generate various new algorithms also giving *n* step convergence. One of them is the Fletcher and Powell modification of Davidon's method.

Results of an extensive numerical comparison of these methods with the Newton–Raphson method, the Fletcher–Reeves method, and the Fletcher–Powell–Davidon method are included, the test functions being non-quadratic.

(Received October 1968)

1. Introduction

The problem considered here is to find the minimising vector x^* for a quadratic function $f(x)$

$$f(x) = \tfrac{1}{2}x'Ax + b'x + c \qquad (1)$$

where A is a $n \times n$ positive definite matrix, b is an arbitrary n-vector, and c is a scalar.

Clearly the minimum is given by the unique solution to the necessary condition $Ax^* + b = 0$ and as such this problem is trivial. In general, however, A may not be known explicitly but values of the gradient at x_i, $g_i = Ax_i + b$ may be given. Or alternatively, (1) may represent a quadratic approximation to a general function arising in some unconstrained minimisation technique, in which case A and b depend on x_i. It is no longer obvious that successively solving $A(x_i) \cdot x^* + b(x_i) = 0$ will be efficient.

It is generally argued that any method which can minimise a quadratic function efficiently will probably be good on more general functions since it can do well in a quadratic vicinity of the minimum. While this is not necessarily true it provides a motivation, and accordingly we are interested in methods which use only first order gradient information, assume that the function is at least locally quadratic and attempt to estimate A^{-1}, the inverse hessian of $f(x)$.

The methods considered here, called variously 'variable metric' [1], 'quasi-Newton' [2], [3] or 'conjugate gradient' methods [4], consist of choosing an $n \times n$ matrix H_i (which approximates A^{-1}) and forming a direction $d_i = H_i'g_i$ using g_i, the gradient at x_i. A new point is defined by $x_{i+1} = x_i + \alpha_i d_i$, where α_i is calculated to minimise $f(x_{i+1})$. The matrix H_i is then updated in some way. If $H_i = I_n$ this is the method of steepest descent. If $H_i = A^{-1}$ this is the generalised Newton–Raphson method.

Since, as is reviewed in §2, a quadratic can be minimised in n steps if $d_0, d_1, \ldots, d_{n-1}$ are conjugate directions, this paper studies a class of H_i matrices that will generate conjugate directions. In §3, H_i is chosen as a projection matrix and, in §4, H_i is chosen as a solution to an equation $H_iY_i = S_i$. The Davidon algorithm [5] is shown to be a member of this latter class. In §5 recursions are developed for the determinant of the metric. We then consider the use of prior information on A in starting the algorithms. A numerical comparison of several new algorithms with the Davidon and the Fletcher–Reeves algorithm is given in §6.

* *Research Analysis Corporation, McLean, Virginia*

Notation

At iteration i the following column vectors, etc., occur:

- x_i is the current solution.
- g_i is the gradient of $f(x)$ at x_i.
- H_i is the current direction matrix or metric.
- d_i is the search direction from x_i.
- $s_i = x_{i+1} - x_i = \alpha_i d_i$ is the step in x_i.
- $y_i = g_{i+1} - g_i = As_i$ is the step in g_i.
- α_i is the step length, a negative scalar.
- g_i' denotes g_i transpose, a row vector.
- $S_i = [s_0, s_1, \ldots, s_{i-1}]$ denotes a matrix with columns $s_0, \ldots, s_{i-1}$ and also without ambiguity $[s_0, s_1, \ldots, s_{i-1}]$ denotes the subspace spanned by the vectors $s_0, s_1 \ldots, s_{i-1}$.
- $Y_i = [y_0, y_1, \ldots, y_{i-1}]$ denotes an $n \times i$ matrix with columns y_j.
- I_n is the $n \times n$ identity matrix.

2. Methods using conjugate directions

A set of n independent and non-zero directions, $d_0, d_1, \ldots, d_{n-1}$ are said to be conjugate if, given a symmetric positive definite matrix A, they satisfy [6]

$$d_i'Ad_j = 0 \quad 0 \leqslant i \neq j \leqslant n \qquad (2)$$

and, of course $d_i'Ad_i > 0$.

If, in accordance with the notation, the directions d_i are the actual search directions, then at iteration $j+1$, $x_{j+1} = x_0 + \sum_{i=0}^{j} \alpha_i d_i$ so that the value of the quadratic function takes the form

$$\begin{aligned} f(x_n) &= \tfrac{1}{2}x_n'Ax_n + x_n'b + c \\ &= \tfrac{1}{2}x_0'Ax_0 + b'x_0 + c \\ &\quad + \sum_{i=0}^{n-1} [\tfrac{1}{2}\alpha_i^2 d_i'Ad_i + \alpha_i d_i'(b + Ax_0)]. \end{aligned}$$

Because α_i is calculated as the minimum value of $f(x_{i+1}) = f(x_i + \alpha_i d_i)$, at x_{i+1} the gradient g_{i+1} must satisfy $d_i'g_{i+1} = 0$. However, $g_{i+1} = Ax_0 + b + \sum_{j=0}^{i} \alpha_j Ad_j$ from which we find

$$\alpha_i = -d_i'(b + Ax_0)/d_i'Ad_i$$

On the other hand the expression above for $f(x_n)$ assumes its minimum value when each α_i is a solution of $\partial f(x_n)/\partial\alpha_i = 0$, i.e.,

$$\alpha_i = -d_i'(b + Ax_0)/d_i'Ad_i$$

as before.

Thus the process of making n one-dimensional minimisations down the directions d_i is equivalent to minimising $f(x_n)$ as a function of the α_i values, and no more than n searches are necessary.

The choice of conjugate directions results in conjugate steps. For what follows it is convenient to argue in terms of the steps

$$s_i = x_{i+1} - x_i$$
$$y_i = g_{i+1} - g_i.$$

Since $g = Ax + b$, we note that

$$y_i = (Ax_{i+1} + b) - (Ax_i + b) = As_i.$$

We will define $n \times i$ matrices Y_1 and S_1 as follows:

$$Y_i = [y_0, y_1, \ldots, y_{i-1}]$$
$$S_i = [s_0, s_1, \ldots, s_{i-1}].$$

Thus conjugacy of the steps $s_i = \alpha_i d_i$ implies

$$\left.\begin{aligned} s_i'As_j &= 0 \; i \neq j \\ Y_i's_j &= 0 \; i \leqslant j \\ S_i'y_j &= 0 \; i \leqslant j. \end{aligned}\right\} \qquad (3)$$

A third orthogonality condition will be useful.

Lemma

Let $f(x)$ be a strictly convex differentiable function of x which has an unconstrained minimum. A necessary and sufficient condition that $x_i = x_0 + \sum_{j=0}^{i-1} \alpha_j d_j$ is the unique minimum of $f(x)$ on an i-dimensional hyperplane through x_0 and spanned by $[d_0, d_1, \ldots d_{i-1}]$ is that $d_i'g_i = 0$ for $j = 0, 1, 2, \ldots, i-1$.

Proof: If x_i is a minimum with respect to each α_j then it is necessary that $\partial f(x_i)/\partial\alpha_j = g_i'd_j = 0$ for $j = 0, 1, 2, \ldots, i-1$. On the other hand suppose $d_j'g_i = 0$ for $j = 0, 1, 2, \ldots i-1$ at x_i, but for some $\hat{x}_i = x_i + \sum_{j=0}^{i-1} \hat{\alpha}_j d_j \neq x_i$ we have $f(\hat{x}_i) \leqslant f(x_i)$. Then using the strict convexity of $f(x)$,

$$f(\hat{x}_i) > f(x_i) + g_i'(\hat{x}_i - x_i) = f(x_i).$$

This is a contradiction unless $x_i = \hat{x}_i$.

If $\alpha_0, \alpha_1, \ldots, \alpha_{i-1}$ are non-zero then the condition above is equivalent to $S_i'g_i = 0$, the form in which it will be used. Consequently any method which successively goes from $S_i'g_i = 0$ to $S_{i+1}'g_{i+1} = 0$ with independent directions will minimise a quadratic in n steps.

3. A projected gradient algorithm

A set of steps s_i and y_i, $i = 0, 1, \ldots, j-1$ are conjugate if $Y_i's_i = 0$, $i = 1, 2, \ldots, j-1$. Successively conjugate directions can be generated by incorporating y_{j-1} into Y_{j-1} to form Y_j, and choosing s_j so that $Y_j's_j = 0$. It is natural to use a search direction d_j which is the gradient g_j projected orthogonal to $[y_0, y_1, \ldots, y_{j-1}]$ and the following recursion achieves this.

Let R be a positive definite symmetric matrix, and let

$$\left.\begin{aligned} i = 0 \quad & H_0 = R, \\ 1 < i < n \quad & H_i = R - RY_i(Y_i'RY_i)^{-1}Y_i'R. \end{aligned}\right\} \qquad (4)$$

At stage i form the gradient g_i, define the search direction $d_i = H_ig_i$ and choose α_i to minimise $f(x_i + \alpha_i d_i)$. Evaluate $x_{i+1} = x_i + \alpha_i d_i$ and g_{i+1}, form s_i and y_i, then repeat for $i+1$. The following theorem shows that the algorithm is successful; a similar result is proved by Goldfarb [8].

Theorem 1

If R and A are positive definite symmetric matrices, the projected gradient algorithm minimises a quadratic form with hessian A in n or fewer steps.

Proof: For $i = 0$, $d_0 = Rg_0$ and the minimisation yields $g_1'd_0 = 0$, i.e., $\alpha_0 = -d_0'g_0/d_0'Ad_0$. Thus if the initial gradient g_0 is not zero then α_0, s_0 and y_0 are non-zero and $s_0'g_1 = \alpha_0 d_0'g_1 = 0$.

Now proceed by induction and assume S_i, Y_i have rank i and $S_i'g_i = 0$. If $g_i \neq 0$ then a cycle of the algorithm gives S_{i+1}, Y_{i+1} of rank $i+1$ and $S_{i+1}'g_{i+1} = 0$. To establish this, note that the new direction $d_i = H_ig_i$. Suppose d_i is identically zero, then since H_i is R times a projection matrix, g_i must have the form Y_iw for some i-vector w. However $S_i'g_i = S_i'Y_iw = S_i'AS_iw = 0$ implies $w = 0$. Thus $d_i \neq 0$ if $g_i \neq 0$. The minimisation yields $g_{i+1}'d_i = 0$, i.e., $\alpha_i = -g_i'H_ig_i/d_i'Ad_i$. Now

$$\begin{aligned} g_i'H_ig_i &= g_i'[R - RY_i(Y_i'RY_i)^{-1}Y_i'R]g_i \\ &= g_i'[R - RY_i(Y_i'RY_i)^{-1}Y_i'R]R^{-1}[R - RY_i \\ &\qquad (Y_i'RY_i)^{-1}Y_i'R]g_i \\ &= d_i'R^{-1}d_i > 0. \end{aligned}$$

Thus if $d_i \neq 0$ then $\alpha_i \neq 0$ and as a result y_i and s_i are both non-zero. The projection property of H_i ensures that $Y_i's_i = 0 = S_i'y_i = S_i'(g_{i+1} - g_i)$. But $S_i'g_i = 0$ by hypothesis and $s_i'g_{i+1}$ as a result of the minimisation. These are equivalent to $S_{i+1}'g_{i+1} = 0$. It is possible that the new Y_{i+1} and S_{i+1} have rank i. However, since $s_iAs_i > 0$ and $Y_i's_i = 0$ implies $s_j'As_i = 0$, $j < i$, then s_i cannot be in the subspace $[s_0, s_1, \ldots, s_{i-1}]$. Thus S_{i+1} and Y_{i+1} must have rank $i+1$.

The induction thus proceeds until either $i < n$, $g_i = 0$ and the minimum of the quadratic function is found, or $S_n'g_n = 0$ at stage n. However, since S_n has rank n, $g_n = 0$. This completes the proof.

Clearly the directions generated are conjugate with respect to A, since $Y_i's_i = 0$ by design for $i = 1, 2, \ldots, n$.

The choice of H_0 is open and although the most obvious choice is $H_0 = I_n$, H_0 can be used to take advantage of a partial inverse of A. See § 5.

Equations (4) and (5) have a well known equivalent form derivable directly from Appendix A, or by induction.

Algorithm 1

$$\left.\begin{aligned} 0 \leqslant i < n \quad & H_{i+1} = H_i - H_iy_iy_i'H_i/y_i'H_iy_i \\ & H_0 = R. \end{aligned}\right\} \qquad (5)$$

4. A class of variable metric algorithms

Practically speaking it is advantageous to use a method in which the metric H_n tends to the inverse hessian A^{-1}. If this is the case, it seems likely that on a general function, if the method reaches a quadratic neighbourhood of the minimum it would do better than any pure conjugate gradient scheme, because the former can do in one step what the latter must do in n steps.

One way of estimating A^{-1} is to use the following idea. For a quadratic function we must have $A^{-1}Y_i=S_i$. Suppose H_i is a matrix satisfying $H_iY_i = S_i$ and we define the search direction as $d_i = H_i'g_i$. Then $Y_i'd_i = Y_i'H_i'g_i = S_i'g_i$. Thus if $S_i'g_i = 0$, $s_i = \alpha_i d_i$, then $Y_i's_i = 0$ and the new step will be conjugate to the previous ones when the function is a quadratic. In addition, at stage n, $H_nY_n = S_n$ and under appropriate conditions $H_n = A^{-1}$ with $d_n = A^{-1}g_n$ the desired Newton step.

A class of general solutions to $H_iY_i = S_i$ can be defined by taking

$$H_i = S_iY_i^* + R(I_n - Y_iY_i^{**}) \tag{6}$$

where Y_i^* and Y_i^{**} have the form $(Y_i'MY_i)^{-1}Y_i'M$ for symmetric positive definite matrices M. (Y_i^* and Y_i^{**} have the property that $x^* = Y_i^*b$ minimises $(Y_ix - b)'M(Y_ix - b)$. See [9])

This general idea will now be formalised into a class of four algorithms.

Let R and M be positive definite symmetric matrices and define the general algorithm as follows:

$$\begin{aligned} &\text{for } i = 0 && H_0 = R \\ &\text{for } 1 \leqslant i \leqslant n && H_i = S_iY_i^* + R(I_n - Y_iY_i^{**}) \end{aligned} \tag{7}$$

where

$$Y_i^*,\ Y_i^{**} \text{ have the form } (Y_i'MY_i)^{-1}Y_i'M \tag{8}$$

with $M = A^{-1}$ or R independently for each term, giving four choices.

Starting with the value $i = 0$, we form the gradient g_i and the search direction $H_i'g_i$. If $g_i = 0$, the algorithm terminates, x_i being the required minimum. Otherwise α_i is calculated to minimise $f(x_i + \alpha_1H_i'g_i)$, and we set $x_{i+1} = x_i + \alpha_iH_i'g_i$ and i is increased by one for the next iteration.

Theorem 2

If R, M and A are positive definite symmetric matrices, M being equal to A^{-1} or R, and if the algorithm is applied to a quadratic function with hessian A, then it terminates after at most n iterations. If n iterations are required, then $H_n = A^{-1}$.

The proof follows the proof of Theorem 1. In particular the argument for $i = 0$, which begins the inductive proof, is identical.

To treat the values of $i \geqslant 1$, we assume as in Theorem 1, that we have $g_i \neq 0$, that Y_i and S_i have rank i, and $S_i'g_i = 0$. Now we shall show that the next iteration causes these results to hold for $i + 1$, except that we may have $g_{i+1} = 0$, in which case the required minimum is found.

The search direction,

$$\begin{aligned} H_i'g_i = Y_i^{*\prime}S_i'g_i + (I_n - Y_i^{**\prime}Y_i')Rg_i \\ + [I_n - MY_i(Y_i'MY_i)^{-1}Y_i']Rg_i \end{aligned}$$

is zero if and only if $M^{-1}Rg_i \in [y_0, y_1, \ldots, y_{i-1}]$. But for $M = R$ or $M = A^{-1}$ it can be shown that $M^{-1}Rg_i \in [y_0, y_1, \ldots, y_{i-1}]$ and $S_i'g_i = 0$ implies $g_i \equiv 0$. Thus if $g_i \neq 0$ then $H_i'g_i \neq 0$.

Computing α_i at stage i, $\alpha_i = -g_i'H_i'g_i/g_i'H_iAH_i'g_i$, and is non-zero if and only if $g_i'H_i'g_i \neq 0$. However, if $M = R$, $g_i'H_i'g_i = g_i'H_iR^{-1}H_i'g_i > 0$, and if $M = A^{-1}$ then $g_i'H_i'g_i = g_i'Rg_i > 0$. Thus if $g_i \neq 0$ then $\alpha_i \neq 0$.

By construction $Y_i'(x_{i+1} - x_i) \equiv Y_i's_i = 0$. Using the argument of Theorem 1 we must also have $S_{i+1}'g_{i+1} = 0$. Furthermore, the reasoning of Theorem 1 shows that if $g_i \neq 0$, then Y_{i+1} and S_{i+1} must have rank $i + 1$.

This completes the inductive cycle. Either at stage $i < n$, $g_i = 0$ and the algorithm terminates with $x_i = x^*$, or it continues to stage $i + 1$. However, at stage n, S_n and Y_n will have rank n with $S_n'g_n = 0$. Consequently $g_n = 0$ and then $H_nY_n = S_n$ has a unique solution $H_n = A^{-1}$.

As before, since by construction $Y_i's_i = 0$ for all i, the search directions for a quadratic turn out to be conjugate. This is not the case if a restart procedure is used, as will be described in §5.

Particular algorithms can be obtained by alternate choices of M in Y_i^* and Y_i^{**}. Recursions for H_{i+1} in terms of H_i, y_i and s_i can then be found by applying the expansion formula (17) of Appendix A.

Let $M = A^{-1}$ in Y_i^* and Y_i^{**} defined by (8).

$$\begin{aligned} H_i &= S_i(S'Y_i)^{-1}S_i' + R[I_n - Y_i(S_i'Y_i)^{-1}S_i'] \\ &= (S_i - RY_i)(S_i'Y_i)^{-1}S_i' + R. \end{aligned} \tag{9}$$

This has the form of (15) in Appendix A, where $A = (S_i - RY_i)$, $B = S_i$, $C = Y_i$, $D = S_i$ and $E = R$. A recursion formula is obtained by noting that at stage $i + 1$, $S_{i+1} - RY_{i+1} = [S_i - RY_i,\ s_i - Ry_i] = [A, a]$, where $a = (s_i - Ry_i)$. Similarly in the notation of Appendix A, we find that $b = s_i$, $c = y_i$ and $d = s_i$. So that H_{i+1} can be written in the form of (16) which reduces to

$$\begin{aligned} H_{i+1} = H_i + [(s_i - Ry_i) \\ - (S_i - RY_i)(S_i'Y_i)^{-1}S_i'y_i][s_i - S_i(Y_i'S_i)^{-1}Y_i's_i]'/\Delta \end{aligned}$$

where $\Delta = s_i'[I_n - Y_i(Y_i'S_i)^{-1}S_i']y_i$. However, on a quadratic function the algorithm guarantees that $Y_i's_i = 0$. Using this and substituting for H_i reduces the general expression above to a simple recursion.

Algorithm 2

$$\begin{aligned} H_{i+1} &= H_i + (s_i - H_iy_i)(s_i)'/s_i'y_i \\ H_0 &= R. \end{aligned} \tag{10}$$

G. P. McCormick first derived this formula using a rank 1 perturbation argument similar to Broyden [2]. Note that H_i will generally be unsymmetric.

Now make an alternative choice of $M = R$ in the formula for Y_i^* and Y_i^{**}.

$$\begin{aligned} H_i &= S_i(Y_i'RY_i)^{-1}Y_i'R + R[I_n - Y_i(Y_i'RY_i)^{-1}Y_i'R] \\ &= (S_i - RY_i)(Y_i'RY_i)^{-1}Y_i'R + R. \end{aligned} \tag{11}$$

Proceeding exactly as before Appendix A can be used to give the expression for H_{i+1} in terms of H_i. Using the additional fact that $Y_i's_i = 0$ when the function is quadratic yields:

Algorithm 3

$$H_{i+1} = H_i + (s_i - H_i y_i)(H_i y_i)'/y_i' H_i y_i$$
$$H_0 = R. \qquad (12)$$

Again H_i is unsymmetric and $H_i' y_i \neq H_i y_i$. Now consider the cases where $Y_i^* \neq Y_i^{**}$.

$$H_i = S_i(S_i' Y_i)^{-1} S_i' + R[I_n - Y_i(Y_i' R Y_i)^{-1} Y_i' R]. \qquad (13)$$

This is obtained by substituting $M = A^{-1}$ in Y_i^* and $M = R$ in Y_i^{**}. Expanding each term independently using (17) recursively shows that (13) is equivalent to

$$H_{i+1} = H_i + [s_i - S_i(S_i' Y_i)^{-1} S_i' y_i] \cdot [s_i - S_i(Y_i' S_i)^{-1} Y_i' s_i]'/\Delta_1 - [Ry_i - RY_i(Y_i' R Y_i)^{-1} Y_i' R y_i][Ry_i - RY_i (Y_i' R Y_i)^{-1} Y_i' R y_i]/\Delta_2$$

where $\Delta_1 = s_i'[I_n - Y_i(S_i' Y_i)^{-1} S_i']y_i$
$\Delta_2 = y_i'[I_n - RY_i(Y_i' R Y_i)^{-1} Y_i']Ry_i.$

However, when the function to be minimised is a quadratic form the algorithm forces $Y_i' s_i = S_i' y_i = 0$. In addition using the definition of H_i in (13)

$$H_i y_i = S_i(S_i' Y_i)^{-1} S_i' y_i + R[I_n - Y_i(Y_i' R Y_i)^{-1} Y_i' R] y_i$$
$$= [I_n - RY_i(Y_i' R Y_i)^{-1} Y_i'] R y_i.$$

Substituting these results lead to:

Algorithm 4

$$H_{i+1} = H_i + s_i s_i'/s_i' y_i - (H_i y_i)(H_i y_i)'/y_i' H_i y_i \qquad (14)$$
$$H_0 = R.$$

This is recognisable as Fletcher and Powell's modification of the Davidon algorithm. Here $H_i = H_i'$.

Finally, a fifth algorithm can be derived by substituting $M = R$ in the expression for Y_i^* and $M = A^{-1}$ in Y_i^{**}

$$H_i = S_i(Y_i' R Y_i)^{-1} Y_i' R + R[I_n - Y_i(S_i' Y_i)^{-1} S_i'].$$

However, the expansion of H_{i+1} does not seem to be representable in terms of H_i, y_i and s_i.

5. Further properties of variable metric algorithms

In this section we give recursions for the determinant of H_i and indicate how to make use of partial information on the hessian A.

It is of interest occasionally to keep track of the determinant of H_i. During calculations involving penalty functions $|H_i|$ often approaches zero and this indicates difficulties due to ill-conditioning of the associated hessian A.

Appendix B contains a brief derivation of the effect of rank 1 and rank 2 perturbations of the determinant of I_n. An alternative derivation of the rank 1 result is given by Bodewig [18].

$$|I_n + x.y'| = 1 + x'y$$
$$|I_n + x.y' + uv'| = (1 + x'y)(1 + u'v) - x'v.y'u.$$

Clearly for Algorithm 1, $|H_i| = 0$ unless $i = 0$. For Algorithm 2, (10), if $|H_i| \neq 0$,

$$H_{i+1} = H_i[I_n + (H_i^{-1} s_i - y_i)(s_i/s_i' y_i)']$$

whence

$$|H_{i+1}| = |H_i|(s_i' H_i^{-1} s_i)/(s' y_i).$$

Similarly for Algorithm 3, (12),

$$|H_{i+1}| = |H_i|(s_i' y_i)(y_i' H_i y_i).$$

For Algorithm 4, the Davidon recursion, (14),

$$|H_{i+1}| = |H_i|(s_i' y_i)/(y_i' H_i y_i).$$

Since $s_i' y_i = s_i'(g_{i+1} - g_i) = -s_i' g_i > 0$ (because we are minimising) and assuming $H_i > 0$, it can be seen that all three recurrences preserve the sign of the determinant of H_i, which is equal to the product of the eigenvalues of H_i.

In a region where A is ill-conditioned small increments in s_i give large increments in y_i, quantities such as $s_i' y_i$ are small while $y_i' H_i y_i$ can be large. On penalty functions $(s_i' y_i / y_i' H_i y_i)$ can be smaller than 10^{-6} and $|H_i|$ rapidly becomes almost singular for these problems.

Fletcher and Powell [5] have shown that the Davidon algorithm does preserve the definiteness of the H_i. These results show directly that definite matrices tend to become singular when the problem hessian is ill-conditioned. Murtagh and Sargent make use of similar recursions [17].

Now consider the use of partial information on A. Suppose the first m rows and columns of A are known. It is possible to solve part of the minimisation problem immediately. Let the gradient g_i and x_i be partitioned into m and $n - m$ components as follows:

$$g_{i1} = A_{11} x_{i1} + A_{12} x_{i2} + b_1$$
$$g_{i2} = A_{21} x_{i1} + A_{22} x_{i2} + b_2$$

where A_{11}, $A_{12} = A_{21}'$ are the known parts of A. Since A_{11} is invertible because A is definite, we can satisfy $g_{i1} = 0$ in one step by moving to any point $x_m = (x_{m1}', x_{m2}')'$ satisfying

$$g_{m1} = A_{11} x_{m1} + A_{12} x_{m2} + b_1 = 0.$$

If in addition b_1 is unknown, then start at any point x_{m-1} and find $x_m = x_{m-1} + \alpha_{m-1} d_{m-1}$ by minimising $f(x_m)$ where [3],

$$d_{m-1} = \begin{pmatrix} A_{11}^{-1} & 0 \\ 0 & 0 \end{pmatrix} g_{m-1}.$$

Direct calculation shows that $g_{m1} = 0$ at the minimum.

We now wish to complete the minimisation in no more than $n - m$ further steps. This can be done by either (i) computing an equivalent H_i which would result had m equivalent steps been taken to reach x_m, or (ii) solve the remaining problem in terms of the $n - m$ independent variables on the hyperplane defined by $g_{i1} = 0$, $i \geqslant m$.

One approach to the first method is to identify m equivalent steps. Let $s_0, s_1, \ldots, s_{m-1}$ be the first m columns of the $n \times n$ identity matrix. Let $y_i = As_i$ for $i \leqslant m - 1$, then each y_i is the ith column of A. The resulting matrices Y_m', S_m both have rank m and $S_m' g_m = 0$. Note that the s_i are not conjugate and $Y_i' s_i \neq 0$ for $i \leqslant m - 1$. However, Theorems 1 and 2 do not assume conjugacy. In fact Y_m and S_m both have the properties required at stage m of the inductive proofs. Thus we can continue from x_m with H_m defined by (5), (7), (8)

using any of the methods. For example, with $R = I_n$ the projected gradient algorithm continues with

$$H_m = I_n - Y_m(Y_m'Y_m)^{-1}Y_m'$$

and the Davidon algorithm continues with

$$H_m = \begin{pmatrix} A_{11}^{-1} & 0 \\ 0 & 0 \end{pmatrix} + I_n - Y_m(Y_m'Y_m)^{-1}Y_m'$$

where $$Y_m' = (A_{11}A_{12}).$$

Since we do not have conjugacy for $i < m$, Algorithms 2, 3, and 4 cannot be used to construct H_m using the equivalent steps. This is inconvenient although it takes the same amount of work to construct H_m directly as it does to do it by updating. [See [18], p. 219.]

All subsequent moves for $i \geqslant m$ lie on the hyperplane since by Theorems 1 and 2, $Y_i's_i = 0$, $i \geqslant m$ and the first m columns of Y_i are the first m rows of A. Thus all later steps satisfy $Y_i's_i = S_i'y_i = 0$ and the Algorithms 2, 3, and 4 can be used. (Algorithm 1 always applies since the steps do not need to be conjugate for the derivation of (5).)

The second approach uses a reduced gradient format [20], starting with any point x_m on the $n - m$ dimension hyperplane $g_{m1} = 0$. Let $x = (y', z')$ where $y = -A_{11}^{-1}(A_{12}z + b_1)$ is an m vector and z is an $n - m$ vector of independent variables. Then $\partial y/\partial z = -A_{11}^{-1}A_{12}$, $y \equiv x_{i1}$, $z \equiv x_{i2}$, $i \geqslant m$.

Let $\bar{H}$ be an arbitrary $(n - m) \times (n - m)$ dimensional metric and $\bar{g}$ be the gradient of $f(x)$ defined by

$$g = \frac{\partial f}{\partial y} \cdot \frac{\partial y}{\partial z} + \frac{\partial f}{\partial z} = \begin{bmatrix} \partial y/\partial z \\ I_{n-m} \end{bmatrix}' . g.$$

If dx is the incremental step in x induced by an increment $dz = \bar{H}'\bar{g} \,.\, d\alpha$ in the subspace of independent variables then,

$$dx = \begin{bmatrix} \partial y/\partial z \\ I_{n-m} \end{bmatrix} dz = \begin{bmatrix} \partial y/\partial z \\ I_{n-m} \end{bmatrix} . \bar{H}'\bar{g} \,.\, d\alpha$$
$$= \left\{ \begin{bmatrix} \partial y/\partial z \\ I_{n-m} \end{bmatrix} \bar{H} \begin{bmatrix} \partial y/\partial z \\ I_{n-m} \end{bmatrix}' \right\}' g \,.\, d\alpha.$$

This is the same as using an $n \times n$ metric H given by,

$$H = \begin{bmatrix} \partial y/\partial z \\ I_{n-m} \end{bmatrix} \bar{H} \begin{bmatrix} \partial y/\partial z \\ I_{n-m} \end{bmatrix}', \quad d = H' \,.\, g.$$

At stage $i \geqslant m$ any of the previous algorithms can be used in place of $\bar{H}$, starting with $\bar{H}_m = I_{n-m}$ and updated using

$$\bar{y}_i = \bar{g}_{i+1} - \bar{g}_i$$
$$\bar{s}_i = z_{i+1} - z_i$$

where $x_i = (x_{i1}', x_{i2}')' = (y_i', z_i')'$, with the step $s_i = \alpha_i d_i$ found in n-space exactly as before. Since $[\partial y/\partial z] = -A_{11}^{-1}A_{12}$ is a constant, all moves are on the $n - m$ dimensional subspace $g_{i1} = 0$ and consequently the algorithms converge to $g_{i2} = 0$ in no more than $n - m$ further steps, as desired.

The idea of using variable metric methods in conjunction with reduced gradient techniques [20], as opposed to projection methods [8, 15], is due largely to McCormick.

6. Numerical results

Results of testing these algorithms on nonquadratic functions will now be given. The numerical procedure for the seven schemes considered is as follows.

Given $f(x)$, $g(x)$, and possibly $A(x)$, the matrix of second partial derivatives of $f(x)$ evaluated at x, and starting at x_0 with $H_0 = R = I_n$, carry out the following operations for $i = 0, 1, \ldots$.

(*a*) Find the first local minimum of $f(x_i + \alpha_i d_i)$

$$f(x_{i+1}) = \min_{\alpha_i} f(x_i + \alpha_i d_i)$$

where $$d_i = H_i'g_i/g_i'H_iH_i'g_i.$$

(*b*) Update H_i according to the algorithm used.

Algorithm 1, *Projected Gradient Method* (*P–G*)

$$H_{i+1} = H_i - (H_iy_i)(H_iy_i)'/(y_i'H_iy_i)$$

when i is a multiple of n, reset $H_i = R$.

Algorithm 2

$$H_{i+1} = H_i + (s_i - H_iy_i)s_i'/y_i's_i$$

Algorithm 3

$$H_{i+1} = H_i + (s_i - H_iy_i)(H_i'y_i)'/(y_i'H_iy_i)$$

Algorithm 4, *Fletcher–Powell–Davidon* (*F–P–D*)

$$H_{i+1} = H_i - (H_iy_i)(H_iy_i)'/(y_i'H_iy_i) + s_is_i'/y_i's_i$$

Algorithm 5, *Newton–Raphson* (*N–R*)

$$H_i = [A(x_i)]^{-1}$$

The program uses a modified Newton–Raphson step when it appears that $A(x_i)$ has negative eigenvalues as identified during the process of solving $A(x_i)g_i = d_i$ using the Crout procedure. In this case the direction of move is roughly along an eigenvector corresponding to a negative eigenvalue. By this means a region is located where the function is convex [10].

Algorithm 6, *Fletcher–Reeves* (*F–R*) [19]

$$d_0 = -g_0$$
$$d_{i+1} = -g_{i+1} + d_i(g_{i+1}'g_{i+1}/g_i'g_i)$$

when i is a multiple of $n + 1$ reset $d_i = -g_i$.

Algorithm 7, *Projected Newton–Raphson* (*P–N–R*)

$$H_{i+1} = H_i - (H_iy_i)(H_iy_i)'/(y_i'H_iy_i)$$
$$R_{i+1} = R_i + (s_i - R_iy_i)(H_iy_i)'/(y_i'H_iy_i)$$

when i is a multiple of n reset $H_i = R_i$.

The last method investigates the effect of solving $R_iY_i = S_i$ exactly using the schemes of §4 in the absence of quadraticity. H_iy_i provides the projection of y_i orthogonal to $[y_0, y_1, \ldots, y_{i-1}]$. Every n steps R_i is an approximation to $A(x_i)^{-1}$ and a Newton–Raphson move is made.

The *reset* form of the algorithm is obtained by resetting H_{n+1} for Algorithms 2, 3, 4, and 7 to R and restarting. Algorithm 1 must be reset every n steps and, in Algorithm 6, d_{n+1} is reset to $-g_{n+1}$ always.

The linear minimisation is performed by a Fibonacci search. Cubic interpolation works well on low-order polynomial functions but does not prove as rapid for the logarithmic penalty functions used in the Sequential Unconstrained Minimisation Technique (SUMT) for which Algorithms 1 to 7, plus several others, make up an experimental XMOVE subroutine [11], [12].

Five problems were considered. The data for these, and other information can be found in [16]. The numerical results are of course strictly comparative for each problem. Computing was done on an IBM 7044 in single precision, 36-bit arithmetic.

Table 1 gives results for Rosenbrock's banana-shaped valley [13].

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \qquad (40)$$

the starting point being $(x_1, x_2) = (-1{\cdot}2, 1{\cdot}0)$; the numbers quoted are the number of iterations required to obtain $f(x^*) < 10^{-13}$.

Table 1

Numerical results of problem 1

ALGORITHM	MODE	
	NORMAL	RESET
1, P–G	—	42
2	18	31
3	21	37
4, F–P–D	19	35
5, N–R	12	—
6, F–R	—	16
7, P–N–R	36	21

Table 2

Numerical results of problem 2

ALGORITHM	NORMAL	RESET
1, P–G	—	65
2	36	47
3	46	47
4, F–P–D	40	49
5, N–R	23	—
6, F–R	—	30
7, P–N–R	58	55

Table 2 gives results for a test function credited to C. F. Wood of Westinghouse Research Laboratory:

$$\begin{aligned} f(x) = {} & 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \\ & + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 \\ & + 10{\cdot}1(x_2 - 1)^2 + (x_4 - 1)^2 \\ & + 19{\cdot}8(x_2 - 1)(x_4 - 1) \end{aligned}$$

This is designed to have a non-optimal stationary point that can cause premature convergence. The initial point is $(x_1, x_2, x_3, x_4) = (-3, -1, -3, -1)$, and the number of iterations is that required to obtain $f(x^*) < 10^{-13}$.

Table 3 shows the results for a test problem formulated by the Shell Development Company:

$$f(x) = \sum_{j=1}^{5} e_j x_j + \sum_{j=1}^{5} \sum_{i=1}^{5} c_{ij} x_i x_j + \sum_{j=1}^{5} d_j x_j^3$$

subject to

$$x_j \geqslant 0, j = 1, 2, \ldots, 5$$

$$\sum_{j=1}^{5} a_{ij} x_j \geqslant b_i, \; i = 1, 2, \ldots, 10.$$

This is a linearly constrained problem, which for particular choices of e_j, c_{ij}, d_j has a convex objective [12]. For this problem, SUMT replaces $f(x)$ by $f(x) - r \sum_{i=1}^{10} \log_n g_i(x)$ for a parameter $r > 0$, where $g_i(x) \geqslant 0$ represents the ith inequality constraint. If $x^*(r)$ is the solution of the modified problem, then $x^*(r) \to x^*$ as $r \to 0$ where x^* is the solution to the actual problem [3].

Table 3

Numerical results of problem 3

ALGORITHM	$r = 1{\cdot}0$		$r = 1{\cdot}56 \times 10^{-2}$		$r = 2{\cdot}44 \times 10^{-4}$	
	NORMAL	RESET	NORMAL	RESET	NORMAL	RESET
1, P–G	—	26	—	55	—	70
2	27	22	44	41	62	60
3	33	22	50	40	67	54
4, F–P–D	27	22	46	40	60	56
5, N–R	11	—	17	—	22	—
6, F–R	—	34	—	>165	—	Fail
7, P–N–R	31	20	50	37	67	54

Table 4 gives results for the dual to the previous problem. Here the dual problem has a cubic objective and quadratic constraints [12], [16].

Table 4

Numerical results of problem 4

ALGORITHM	$r = 1{\cdot}0$		$r = 0{\cdot}25$		$r = 0{\cdot}0625$	
	NORMAL	RESET	NORMAL	RESET	NORMAL	RESET
1, P–G	—	120	—	169	—	211
2	134	98	195	132	221	187
3	136	100	220	134	246	168
4, F–P–D	406	97	473	133	500	169
5, N–R	30	—	36	—	48	—
6, F–R	—	>489	—	Fail	—	Fail
7, P–N–R	166	113	198	150	230	186

Finally, **Table 5** shows the results for an intriguing problem of maximising the area of a hexagon subject to the constraint that its maximum diameter is 1. It is interesting to note that the solution is *not* a regular

Table 5

Numerical results of problem 5

ALGORITHM	$r = 1{\cdot}0$		$r = 10^{-2}$		$r = 10^{-4}$		$r = 10^{-6}$	
	NORMAL	RESET	NORMAL	RESET	NORMAL	RESET	NORMAL	RESET
1, P–G	—	13	—	55	—	194	—	278
2	17	20	34	42	308	79	326	97
3	11	11	31	32	73	56	96	70
4, F–P–D	47	18	66	40	206	64	215	80
5, N–R	18	—	30	—	54	—	65	—
6, F–R	—	13	—	55	—	194	—	278
7, P–N–R	19	23	51	58	92	91	120	101

hexagon [14]. The particular formulation used had 9 variables and 13 inequality constraints, although there is a certain amount of redundancy.

Problems of this last kind with various number of sides form an excellent set of standard tests.

Summary of results

The range of examples used here illustrates the marked difference between minimising the polynomial functions of problems 1 and 2 and minimising penalty functions. The principal difference is that the hessian of the penalty functions at points where one or more constraints are binding is excessively ill conditioned. Indeed the determinant of H_i for all methods rapidly becomes effectively zero.

Ranking the methods depends on the test problem used. For the more well behaved tests 1 and 2, the Fletcher–Reeves algorithm is simple and fast, while the variable metric algorithms follow somewhat behind.

For the penalty function methods, the variable metric algorithms are much better and operate more efficiently when reset. This is presumably because their estimated H_i matrices become singular in directions of descent. Problems 3, 4, and 5 indicate that the two new algorithms (2 and 3) appear to be competitive.

The generalised Newton–Raphson algorithm always required fewer iterations, and when it can be used it generally proves to be the quickest method.

The one surprising result was the performance of a pure projection method of Algorithm 1, since it contains no second derivative information, yet it is better than the Fletcher–Reeves algorithm. Algorithm 7, showing the effect of resetting a projection to an estimate of the inverse hessian, indicates that an improvement can be made this way although it does not appear to be worth the trouble.

Conclusions

This paper has unified a series of algorithms in a single framework. Basically, this is that variable metric schemes depend on the generalised solution to a set of linear equations, and their associated projection properties give rise to conjugate directions. A result of this general approach has been three new algorithms whose comparative numerical properties are promising. Extensions to this work will be found elsewhere in [7] which is concerned with the rate of convergence of a general conjugate direction method.

The results of this paper raise the possibility of performing an error analysis of the variable metric schemes considered as methods of solving linear equations. This should provide a means of selecting the best algorithm for an ill-conditioned quadratic problem, and shed some light on its use for penalty functions.

Acknowledgements

The author wishes to thank G. P. McCormick for his inspiration during many fruitful discussions. The original draft of this paper has been considerably improved due to the encouragement and criticism of M. J. D. Powell. The work reported was performed under U.S. Army Research Office Contract no. DA–44–188–ARO–1.

Appendix A

Bordered inverse lemma

As an application of the matrix inverse lemma consider

$$H_i = A(B'C)^{-1}D' + E \tag{15}$$

where A, B, C, D are all $n \times i$ matrices with $i < n$ such that they have rank i. H_i and E are $n \times n$ matrices. Let a, b, c, d be n-vectors such that $[A, a]$, etc., have rank $i + 1$ and consider

$$H_{i+1} = [A, a]([B, b]'[C, c])^{-1}[D, d]' + E \tag{16}$$

$$= [A, a]\begin{bmatrix} B'C & B'c \\ b'C & b'c \end{bmatrix}^{-1}[D, d]' + E.$$

Applying the bordered inverse lemma to the centre matrix [15],

$$H_{i+1} = [A, a]\left(\begin{bmatrix} (B'C)^{-1} & 0 \\ 0 & 0 \end{bmatrix} + [-c'BR', 1]'\,\Delta^{-1}[-b'CR, 1]\right)[D, d]' + E$$

where

$$R = (B'C)^{-1}$$

$$\Delta = b'(I_n - CRB')c.$$

Now multiplying through yields

$$H_{i+1} = A(B'C)^{-1}D' + (-ARB'c + a)\Delta^{-1}(-b'CRD' + d') + E$$
$$= \frac{H_i + (a - A(B'C)^{-1}B'c)(d - D(C'B)^{-1}C'b)'}{b'(I_n - C(B'C)^{-1}B')c} \quad (17)$$

giving the basic formula used throughout this work.

Appendix B

Formula for the perturbation of a determinant

Let T be a non-singular transformation; then since

$$|T^{-1}AT| = |T^{-1}|.|A|.|T| = |A|$$

for any independent vectors y, v, we have

$$|I_n + xy' + uv'| = |I_n + T^{-1}x.y'T + T^{-1}u.v'T|.$$

Given y and v we calculate T to satisfy the equations

$$y'T = e_1', \; v'T = e_2'$$

where e_i are the unit coordinate vectors. Let

$$T^{-1}x = a, \; T^{-1}u = b,$$

where a, b are n vectors, then the determinant becomes,

$$|e_1 + a, e_2 + b, e_3, e_4, \ldots, e_n| = [(1 + a_1)(1 + b_2) - a_2b_1]$$

where a_1 is the first element of a and

$$a_1 = e_1'a = e_1'T^{-1}x = y'x, \text{ etc.}$$

Thus

$$|I_n + xy' + uv'| = (1 + x'y)(1 + u'v) - \boldsymbol{v'}x.y'u.$$

In particular, the case $u = 0$ gives

$$|I_n + xy'| = (1 + x'y).$$

References

(1) Davidon, W. C. (1959). *Variable Metric Method for Minimization*, Research and Development Rept. ANL-5990 (rev), US Atomic Energy Commission, Argonne National Laboratories.

(2) Broyden, C. G. (1967). Quasi-Newton Methods and Their Application to Function Minimization, *Math. of Computation*, Vol. 21, pp. 368–81.

(3) Fiacco, A. V., and McCormick, G. P. (1968). *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley & Sons, Inc., New York.

(4) Antosiewicz, H. A., and Rheinboldt, W. C. (1962). Numerical Analysis and Functional Analysis, Ch. 14 in *Survey of Numerical Analysis*, editor J. Todd, McGraw-Hill Book Co., New York.

(5) Fletcher, R., and Powell, M. J. D. (1963). A Rapidly Convergent Descent Method for Minimization, *Computer Journal*, Vol. 6, pp. 163–68.

(6) Hestenes, M. R. (1956). The Conjugate Gradient Method for Solving Linear Systems, *Proceedings of the Symposium on Applied Mathematics*, Vol. VI, pp. 83–102, McGraw-Hill Book Co., New York.

(7) McCormick, G. P., and Pearson, J. D. (1968). Variable Metric Methods and Unconstrained Optimization, Conference on Optimization, Keele Hall, Staffordshire, England, March 1968.

(8) Goldfarb, D. (1966). Extension of Davidon's Variable Metric Method to Maximization Under Linear Inequality and Equality Constraints (presented at SIAM National Meeting, Iowa City, Iowa, May 1966).

(9) Ben-Israel, A., and Charnes, A. (1963). Contributions to the Theory of Generalized Inverses, *SIAM J. Appl. Math.*, Vol. 11, pp. 667–99.

(10) McCormick, G. P., and Zangwill, W. I. (1967). A Technique for Calculating Second-Order Optima, technical paper in preparation, Research Analysis Corporation, McLean, Virginia.

(11) Fiacco, A. V., and McCormick, G. P. (1964). Computational Algorithm for the Sequential Unconstrained Minimization Technique for Nonlinear Programming, *Mgt. Sci.*, Vol. 10, pp. 601–17.

(12) McCormick, G. P., Mylander, W. C., and Fiacco, A. V. (1965). Computer Program Implementing the Sequential Unconstrained Minimization Technique for Nonlinear Programming, RAC-TP-151, Research Analysis Corporation, April 1965.

(13) Rosenbrock, H. H. (1960). Automatic Method for Finding the Greatest or Least Values of a Function, *Computer Journal*, Vol. 3, pp. 175–184.

(14) Mylander, W. C. (1967). *A Geometric Problem Solved by Nonlinear Programming*, Internal Memorandum, Research Analysis Corporation, McLean, Virginia, August 1967.

(15) Rosen, J. B. (1960). The Gradient Projection Method for Nonlinear Programming—Part 1, Linear Constraints, *SIAM J. Appl. Math.*, Vol. 8, pp. 181–217.

(16) Pearson, J. D. (1968). *On Variable Metric Methods of Minimization*, technical paper, RAC-TP-302, Research Analysis Corporation, McLean, Virginia, Feb. 1968. Also available as AD 666700 from CFSTI, Springfield, Virginia, 22151, USA.

(17) Murtagh, B. A., and Sargent, R. W. H. (1968). A Constrained Minimization Method with Quadratic Convergence presented in discussion at Conference on Optimization, Keele Hall, Staffordshire, England, March 1968.

(18) Bodewig, E. (1959). *Matrix Calculus*, North-Holland and Interscience.

(19) Fletcher, R., and Reeves, C. M. (1964). Function Minimization by Conjugate Gradients, *Computer Journal*, Vol. 7, pp. 149–154.

(20) Wolfe, P. (1963). Methods of Nonlinear Programming, in *Recent Advances in Nonlinear Programming*, editors Graves and Wolfe, McGraw-Hill, 1963.