JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 23, No. 3, NOVEMBER 1977

A Conjugate Direction Algorithm Without Line Searches[1,2]

L. NAZARETH[3]

Communicated by H. Y. Huang

Abstract. We develop an algorithm which generates conjugate search directions and maintains finite termination, when applied to quadratic functions, without requiring that line searches be exact. The technique requires $O(n)$ storage, where n is the dimension of the problem. Results when the algorithm is applied to a number of standard test problems are included.

Key Words. Unconstrained minimization, conjugate-gradient methods, conjugate directions without line searches, projection methods, high-dimensional function minimization.

1. Introduction

When selecting an algorithm for minimizing an unconstrained function on a computer, a factor that often plays a decisive role is the demand that the algorithm places on computer resources. Conjugate-gradient algorithms, first developed by Hestenes and Stiefel (Ref. 1), and then applied to the minimization of functions by Fletcher and Reeves (Ref. 2), have proven to be popular, since they are economical in their usage of computer storage and in their operation count, and since their convergence rate has been found to be satisfactory.

Conjugate-gradient methods minimize a function $F(x)$ of n variables, starting from initial point x_1, by searching in sequence along directions $d_1, d_2, \ldots, d_j, \ldots$ and obtaining successive approximations

[1] Work was performed under the auspices of the US Energy Research and Development Administration.

[2] The author would like especially to thank Marie-Anne Neimat, who put much effort into the programming of the algorithm and the generation of test results.

[3] Assistant Computer Scientist, Applied Mathematics Division, Argonne National Laboratory, Argonne, Illinois.

$x_1, x_2, \ldots, x_j, \ldots$ to the minimum. They are described by the relations

$$\begin{aligned} d_1 &= -g_1 \\ d_j &= -g_j + \beta_j d_{j-1}, \\ x_{j+1} &= x_j + \lambda_j d_j, \qquad \lambda_j \neq 0, \end{aligned} \tag{1}$$

where g_j is the gradient of $F(x)$ at x_j, β_j is a scalar most commonly given by

$$\beta_j = g_j^T g_j / g_{j-1}^T g_{j-1}, \tag{2}$$

and $\lambda_j \neq 0$ is a scalar chosen so that x_{j+1} minimizes $F(x)$ along d_j, i.e.,

$$d_j^T \nabla F(x_{j+1}) = 0. \tag{3}$$

Among the properties of such methods when applied to a quadratic function

$$\psi(x) = a + b^T x + \tfrac{1}{2} x^T A x,$$

where A is an $n \times n$ positive-definite, symmetric matrix are the following:

(a) Generation of conjugate directions, i.e., directions satisfying the relation

$$d_i^T A d_j = 0, \qquad \forall i \neq j.$$

(b) Orthogonality of gradients at different iterates.

(c) Finite termination, in at most n steps.

However, in order for these properties to hold, it is necessary that relation (3) be satisfied exactly at each step, i.e., that λ_j be chosen so that the line search is exact.

Many variants of the basic method have been proposed. Fletcher (Ref. 3) suggests

$$\beta_j = y_{j-1}^T g_j / y_{j-1}^T d_{j-1}, \tag{4}$$

where

$$y_{j-1} \equiv g_j - g_{j-1}. \tag{5}$$

The suggestion has the merit that

$$d_j^T y_{j-1} = 0,$$

even for nonquadratic functions and inexact line searches. This relation implies that d_j and d_{j-1} are conjugate for quadratic functions.

The Polyak–Polak–Ribiere suggestion (Ref. 4) is

$$\beta_j = y_{j-1}^T g_j / g_{j-1}^T g_{j-1}. \tag{6}$$

It follows from properties (a) and (b) above that, for quadratic functions and exact line searches, the alternative choices for β_j are equivalent; but, for

nonquadratic functions or inexact line searches, each choice leads to a distinct algorithm.

In this paper, a method is proposed that resembles the conjugate-gradient method, but has the property that it maintains mutual conjugacy of all search directions over a quadratic, even when relation (4) is not satisfied, i.e., even when the line searches are not exact.

The recurrence relations for developing successive search directions are a particular expression of the three-term recurrence for a self-adjoint operator in an inner product space and are given by

$$d_1 = -g_1,$$
$$d_j = -y_j + (y_{j-1}^T y_j / y_{j-1}^T d_{j-1}) d_{j-1} + (y_j^T y_j / y_j^T d_j) d_j, \tag{7}$$

where

$$d_0 \equiv 0.$$

This recurrence relation is derived in Section 2. Next, we prove two simple lemmas which indicate why it may be worthwhile to maintain conjugacy of the search directions; these lemmas form the basis of a technique for retaining quadratic termination without requiring the storage of n vectors. The algorithm and some details of implementation are outlined in the next section; finally, Section 4 sets out the results when the method is applied to several standard test problems.

2. Basic Relation

In deriving the basic relation, three conditions are imposed.

(a) For $j = 1, 2, \ldots, n-1$, we require that each computed search direction, typically d_{j+1}, lie in the subspace spanned by the gradients, say $\{g_1, g_2, \ldots, g_{j+1}\}$, at all iterates, say $\{x_1, x_2, \ldots, x_{j+1}\}$, which have been generated so far.

(b) We require that the search directions be conjugate, i.e.,

$$d_i^T A d_j = 0, \qquad \forall i \neq j.$$

(c) We assume that we are working with a quadratic function

$$\psi(x) = a + b^T x + \tfrac{1}{2} x^T A x,$$

where A is a positive-definite symmetric matrix.

Defining

$$y_j = g_{j+1} - g_j,$$

we see that condition (a) above implies that

$$d_{j+1} = -y_j + \sum_{i=1}^{j} r_{ij} d_i, \qquad r_{ij} \in \mathbb{R}. \tag{8}$$

From conjugacy, we have

$$y_k^T d_{j+1} = 0, \qquad 1 \le k \le j.$$

Thus, from (8),

$$r_{kj} = y_k^T y_j / y_k^T d_k, \qquad 1 \le k \le j. \tag{9}$$

Also, since y_k is in the space spanned by $d_1, \ldots, d_{k+1}$, we have

$$y_k^T y_j = 0, \qquad 1 \le k \le j-2,$$

again using the conjugacy of search directions. Thus,

$$d_{j+1} = -y_j + (y_{j-1}^T y_j / y_{j-1}^T d_{j-1}) d_{j-1} + (y_j^T y_j / y_j^T d_j) d_j. \tag{10}$$

Analogously to the conjugate-gradient method, a number of alternative expansions for the relation (10) may be derived and are equivalent for quadratic functions, e.g.,

$$d_{j+1} = -y_j + (y_{j-1}^T y_j / y_{j-1}^T d_{j-1}) d_{j-1} + (1 + y_j^T g_{j+1} / y_j^T d_j) d_j. \tag{11}$$

In (10) and (11), we use the convention

$$d_0 \equiv 0, \qquad d_1 = -g_1, \qquad \text{and } 1 \le j \le n-1.$$

If some g_{j+1} becomes linearly dependent upon $g_1, \ldots, g_j$, with $g_1, \ldots, g_j$ being linearly independent, it is not difficult to show that the minimum point $x_{\min}$ of a quadratic must lie in the affine space V defined by x_1 and the j mutually conjugate directions $d_1, \ldots, d_j$, i.e.,

$$V = \left\{ z : z = x_1 + \sum_{k=1}^{j} t_k d_k \right\}, \qquad t_k \in \mathbb{R}. \tag{12}$$

However, we do not incorporate this feature directly into our algorithm, since, *for arbitrary functions*, the recognition of linear dependence of gradients would require $O(n^2)$ storage.

With the conjugate-gradient algorithm, we observe that, when line searches are not exact, the directions generated for a quadratic are no longer conjugate. However, directions generated by (10) or (11) remain conjugate even with inexact line searches, i.e., when the relation

$$d_j^T \nabla \psi(x_{j+1}) = 0$$

is not satisfied. We utilize this property and the results of the next two lemmas to develop a technique for retaining finite quadratic termination without having to store n search directions.

Lemma 2.1. Suppose that $d_1, \ldots, d_n$ are a given set of n linearly independent directions. Let $x_1, x_2, \ldots, x_{n+1}$ be the sequence of points generated by searching along each of these directions in turn over the quadratic $\psi(x)$, starting from a given point x_1, i.e.,

$$x_{i+1} = x_i + \lambda_i d_i.$$

Also, let $\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_{n+1}$ be the sequence of points generated by exact searches along $d_1, \ldots, d_n$, starting from x_1 with

$$\hat{x}_1 = x_1,$$

i.e.,

$$\hat{x}_{i+1} = \hat{x}_i + \hat{\lambda}_i d_i.$$

Finally, suppose that

$$\tilde{x}_{i+1} = x_i + \tilde{\lambda}_i d_i,$$

where $\tilde{\lambda}_i$ is chosen so that

$$\psi(x_i + \tilde{\lambda}_i d_i) \leq \psi(x_i + \mu d_i),$$

for all μ. Define e_i and ϵ_i by

$$e_i = \hat{x}_1 - x_i, \qquad \epsilon_i = \tilde{\lambda}_i - \lambda_i.$$

Then,

$$e_{i+1} = e_i - (d_i^T A e_i / d_i^T A d_i) d_i + \epsilon_i d_i. \tag{13}$$

Proof. Since the gradient at $\hat{x}_{i+1}$ is orthogonal to d_i, this implies that

$$\hat{\lambda}_i = -d_i^T (A\hat{x}_i + b) / d_i^T A d_i.$$

Similarly,

$$\tilde{\lambda}_i = -d_i^T (Ax_i + b) / d_i^T A d_i.$$

Thus,

$$\hat{x}_{i+1} - \tilde{x}_{i+1} = e_i - (d_i^T A e_i / d_i^T A d_i) d_i.$$

Since

$$\tilde{x}_{i+1} - x_{i+1} = \epsilon_i d_i,$$

therefore

$$e_{i+1} \equiv \hat{x}_{i+1} - x_{i+1} = e_i - (d_i^T A e_i / d_i^T A d_i) d_i + \epsilon_i d_i \tag{14}$$

and

$$e_1 = 0.$$

Note, finally, that e_i is in the space spanned by $d_1, d_2, \ldots, d_{i-1}$. □

Corollary 2.1. If the directions $d_1, \ldots, d_n$ are mutually conjugate, then

$$e_{i+1} = \sum_{j=1}^{i} \epsilon_i d_i. \tag{15}$$

Proof. This follows directly from (14) and mutual conjugacy. Observe also that $\|e_{n+1}\|_2$ is then also the distance of x_{n+1} from the minimum of $\psi(x)$. □

The following lemma is almost obvious.

Lemma 2.2. For a quadratic function $\psi(x)$, given any iterate x_i and a search direction d_i, let

$$x_{i+1} = x_i + \lambda_i d_i,$$

for some step $\lambda_i \neq 0$ and

$$\hat{x}_{i+1} = x_i + \hat{\lambda}_i d_i,$$

where $\hat{\lambda}_i$ is chosen so that

$$d_i^T \nabla \psi(\hat{x}_{i+1}) = 0.$$

Then,

$$\epsilon_i = \hat{\lambda}_i - \lambda_i = \lambda_i(-g_{i+1}^T d_i / y_i^T d_i). \tag{16}$$

Proof. For a quadratic,

$$y_i \equiv g_{i+1} - g_i = \lambda_i A d_i.$$

Since

$$d_i^T \nabla \psi(\hat{x}_{i+1}) = 0,$$

then

$$\hat{\lambda}_i = -g_i^T d_i / d_i^T A d_i = -\lambda_i g_i^T d_i / y_i^T d_i.$$

Thus,

$$\epsilon_i = \hat{\lambda}_i - \lambda_i = -\lambda_i(g_{i+1}^T d_i / y_i^T d_i). \qquad \square$$

When applied to $\psi(x)$, successive directions $d_1, \ldots, d_n$ generated by (7) are conjugate, i.e.,

$$D^T A D = \alpha \equiv \operatorname{diag}(\alpha_1, \ldots, \alpha_n),$$

where

$$D = (d_1, \ldots, d_n), \qquad \alpha_i = d_i^T A d_i = d_i^T y_i / \lambda_i.$$

In order to retain quadratic terminations, the straightforward approach would be to take an additional Newton step to the minimum $x_{\min}$ given by

$$x_{\min} = x_{n+1} - A^{-1} g_{n+1} = x_{n+1} - (D\alpha^{-1} D^T) g_{n+1}. \tag{17}$$

This indeed is the approach taken by the projection method of Hestenes (Ref. 5); see also Powell (Ref. 6); and it requires retention of n directions.

Using the results of Lemmas 2.1 and 2.2, however, we see that a correction term

$$\sum_{i=1}^{j} \epsilon_i d_i$$

may be accumulated in a single n-vector, with ϵ_i defined by (16). Thus, a final correction term

$$\sum_{i=1}^{n} \epsilon_i d_i$$

may be applied to x_{n+1}. Hence, finite quadratic termination is retained, since $\hat{x}_{n+1}$ is the minimum point of $\psi(x)$, a consequence of the fact that it is obtained by exact line searches along the n conjugate directions d_i. This device is included in our algorithm and has proven satisfactory in practice when the algorithm is applied to more general functions. Note that there is no justification for applying a correction of the form (13) for the conjugate-gradient method when line searches are inexact, since the directions generated are not then conjugate and $\hat{x}_{n+1}$ obtained by exact line searches along nonconjugate directions is not the minimum point of $\psi(x)$. Of course, for a quadratic, the minimum point and the gradient at this point when searching along any direction d_i can be deduced directly from a knowledge of the gradients at any two distinct points along d_i, and this point may be taken as the new current iterate along d_i; but it is difficult to imagine an algorithm applicable to arbitrary functions which uses such an *estimated* gradient at an iterate to determine subsequent search directions, rather than a *computed* gradient and function value.

3. Implementation

The basic steps of our algorithm may be summarized as follows.

Step 1. The current iterate is taken to be the initial point supplied.

Step 2. Set iteration count J to 1. Initialize correction vector C to zero.

Step 3. Set the current search direction to the negative gradient at the current iterate.

Step 4. If the directional derivative at the current iterate is ≥ 0, go to Step 11.

Step 5. Determine the initial stepsize.

Step 6. Develop the next iterate along the current search direction d_j. The condition

$$y_j^T d_j > 0$$

must be satisfied, but the line search need not be exact.

Step 7. Test for convergence.

Step 8. Develop the correction term ϵ_j given by (16), and update the correction vector by

$$C \leftarrow C + \epsilon_j d_j.$$

Step 9. $J \leftarrow J+1$. If $J > n$, then go to Step 11.

Step 10. Update the search direction using relations (10) or (11). Go to Step 4.

Step 11. Develop the next iterate along search direction given by C, with the initial step along this direction being unity. If the convergence test fails, then go to Step 2.

Remark 3.1. The direction d_j developed when the algorithm is applied to a quadratic function is the projected gradient conjugate to $d_1, \ldots, d_{j-1}$; thus it is orthogonal to $y_1, \ldots, y_{j-1}$. If g_j is linearly dependent upon previous gradients (and, thus, upon $y_1, \ldots, y_{j-1}$), then d_j vanishes. This is detected at Step 4, in which case the correction accumulated so far is applied at Step 11 giving quadratic termination.

Our primary aim in the computational study was to investigate the characteristics of our method. Therefore, in carrying out a comparison with the conjugate-gradient method, identical techniques for initial step length selection, line search, and convergence criteria are employed. The implementations of the two algorithms differ essentially in the way search directions are developed and in the use of the correction vector. The conjugate-gradient method used was that of Fletcher (Ref. 3) in a somewhat reformulated version made available to the author by K. E. Hillstrom.

Thus, in Step 6, we have used the cubic interpolation and extrapolation technique of Fletcher–Reeves as employed in Fletcher (Ref. 3). The method

seeks a new iterate x_{j+1} along d_j such that

$$|g_{j+1}^T d_j| / |g_j^T d_j| \leq P < 1,$$

where P is some fixed parameter. For the conjugate-gradient method, the value

$$P = 0.1$$

has been found in practice to be best, whereas, for variable-metric methods using this search strategy, the common setting is

$$P = 0.9.$$

The tables in the next section show the results of running our algorithm and the Fletcher–Reeves conjugate-gradient algorithm for various settings of P.

In Step 5, again in our algorithm we have used Fletcher's choice of the initial steplength given by

$$\alpha = 2(F(x_j) - F(x_{j-1}))/g_j^T d_j.$$

Finally, the convergence criterion used by Fletcher accepts an iterate such that

$$\|x_{j+1} - x_j\| \leq \text{ACC},$$

where ACC is user supplied. We have found that, on rare occasions, this results in false convergence to a nonminimum point. Therefore, we have included in both the conjugate-gradient method and our method an additional check on the gradient norm; prior to exit, if this test fails, the algorithms are restarted.

4. Results

The results for a number of text functions are tabulated in Tables 1–8. Our algorithm is denoted by EQR (standing for explicit QR method) and the conjugate-gradient method is denoted CG. For each value of P (the line search accuracy parameter discussed in the previous section), we tabulate for both algorithms the final function values obtained (Function value), the total number of iterations (Iterations), and the total number of calls to the function subroutine (Function calls), each consisting of one function and one gradient evaluation. In addition, following a suggestion of Davidon (Ref. 7), for each P we broke down Iterations and Function calls into three subsets corresponding to the number of iterations and function calls needed to

Table 1. Quadratic function, $N = 9$.

	Function value		Iterations		Function cells	
P	EQR	CG	EQR	CG	EQR	CG
0.1	0.649D-29	0.924D-24	10	10	26	27
0.2	0.649D-29	0.778D-32	10	10	26	26
0.3	0.649D-29	0.778D-32	10	10	26	26
0.4	0.138D-27	0.752D-36	11	20	25	44
0.5	0.870D-25	0.394D-12	11	26	24	52
0.6	0.870D-25	0.293D-13	11	27	24	54
0.7	0.562D-20	0.155D-12	11	35	23	65
0.8	0.219D-19	0.731D-12	11	40	22	69
0.9	0.727D-19	0.200D-11	11	64	21	93
0.95	0.545D-17	0.121D-08	11	64	20	89

reduce the function value below 3.0, to reduce it from 3.0 to 0.05, and finally to reduce it from 0.05 to the final value. These are not given here, since they would expand the tables to an unacceptable length. Also, x_0 is the starting vector.

The functions tested were as follows.

4.1. Quadratic function:

$$F(x) = \sum_{k=1}^{9} kx_k^2,$$

$x_0 = (1, 2, 3, \ldots, 9)$.

Table 2. Powell's quartic function, $N = 4$.

	Function value		Iterations		Function calls	
P	EQR	CG	EQR	CG	EQR	CG
0.1	0.137D-06	0.999D-07	26	70	67	174
0.2	0.502D-07	0.922D-07	76	74	180	161
0.3	0.290D-07	0.382D-07	47	79	105	178
0.4	0.207D-06	0.163D-06	32	93	68	184
0.5	0.144D-06	0.131D-06	69	119	145	225
0.6	0.728D-07	0.107D-06	65	129	118	227
0.7	0.993D-08	0.633D-06	86	179	158	285
0.8	0.731D-07	0.108D-05	81	272	141	414
0.9	0.746D-07	0.122D-04	42	250	67	354
0.95	0.109D-07	0.461D-05	45	487	70	640

Table 3. Box's exponential function, $N = 3$.

	Function value		Iterations		Function calls	
P	EQR	CG	EQR	CG	EQR	CG
0.1	0.149D-13	0.720D-15	35	28	95	79
0.2	0.124D-19	0.981D-11	32	33	76	87
0.3	0.436D-15	0.164D-17	32	69	75	163
0.4	0.134D-20	0.112D-12	21	69	58	143
0.5	0.428D-08	0.173D-08	32	25	67	61
0.6	0.147D-10	0.586D-08	49	69	105	138
0.7	0.236D-17	0.163D-09	42	209	86	381
0.8	0.112D-14	0.201D-08	37	200	71	351
0.9	0.868D-12	0.509D-06	50	640	86	>1000
0.95	0.771D-15	0.239D-05	50	667	88	>1000

4.2. Powell's quartic function:

$$F(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4,$$

$$x_0 = (3, -1, 0, 1).$$

4.3. Box's exponential function:

$$F(x) = \sum_{j=1}^{10} \{\exp(-x_1 j/10) - \exp(-x_2 j/10) - x_3[\exp(-j/10) - \exp(-j)]\}^2,$$

$$x_0 = (0, 20, 1).$$

Table 4. Helical function, $N = 3$.

	Function value		Iterations		Function calls	
P	EQR	CG	EQR	CG	EQR	CG
0.1	0.455D-19	0.184D-12	30	36	85	83
0.2	0.487D-18	0.145D-15	31	36	87	84
0.3	0.989D-14	0.319D-13	29	40	67	83
0.4	0.185D-15	0.636D-16	35	44	80	94
0.5	0.495D-12	0.663D-09	35	48	81	92
0.6	0.159D-10	0.215D-12	32	53	67	97
0.7	0.114D-07	0.598D-09	33	126	64	204
0.8	0.119D-12	0.105D-07	43	126	82	205
0.9	0.738D-10	0.821D-07	28	311	58	432
0.95	0.798D-12	0.182D-07	32	377	54	509

Table 5. Wood's quartic function, $N = 4$.

	Function value		Iterations		Function calls	
P	EQR	CG	EQR	CG	EQR	CG
0.1	0.799D-10	0.510D-09	186	132	511	317
0.2	0.386D-12	0.119D-10	186	156	467	363
0.3	0.635D-09	0.279D-16	173	180	418	400
0.4	0.102D-10	0.173D-14	169	186	401	384
0.5	0.788D-09	0.126D-09	186	191	405	378
0.6	0.900D-10	0.346D-09	188	299	369	558
0.7	0.958D-13	0.396D-07	179	341	316	590
0.8	0.139D-13	0.288D-04	180	635	319	>1000
0.9	0.620D-08	0.771D-01	185	664	299	>1000
0.95	0.444D-17	0.707D-01	191	739	289	>1000

4.4. Fletcher's helical function:

$$F(x) = 100[(x_3 - 10\theta)^2 + (R(x_1, x_2) - 1)^2] + x_3^2,$$

$$\theta = \begin{cases} \tan^{-1}(x_2/x_1)/2\pi, & x_1 > 0, \\ 0.75, & x_1 = 0, \\ \tan^{-1}(x_2/x_1)/2\pi + 0.5, & x_1 < 0, \end{cases}$$

$$R(x_1, x_2) = (x_1^2 + x_2^2)^{1/2},$$

$$x_0 = (-1, 0, 0).$$

Table 6. Powell's 2nd function, $N = 3$.

	Function value		Iterations		Function calls	
P	EQR	CG	EQR	CG	EQR	CG
0.1	0.599D-10	0.258D-12	7	9	18	21
0.2	0.352D-11	0.101D-11	10	7	32	19
0.3	0.352D-11	0.106D-11	10	7	32	18
0.4	0.0	0.436D-11	17	15	62	29
0.5	0.0	0.139D-16	14	11	60	24
0.6	0.0	0.397D-11	14	15	60	29
0.7	0.404D-11	0.187D-13	14	11	40	21
0.8	0.365D-07	0.187D-13	11	11	19	21
0.9	0.365D-07	0.187D-13	11	11	19	21
0.95	0.365D-07	0.155D-12	11	27	19	42

Table 7. Beale's function, $N = 2$.

	Function value		Iterations		Function calls	
P	EQR	CG	EQR	CG	EQR	CG
0.1	0.179D-16	0.136D-15	50	24	185	82
0.2	0.933D-11	0.288D-16	55	36	200	102
0.3	0.412D-12	0.642D-10	47	31	154	90
0.4	0.586D-13	0.623D-11	45	33	142	94
0.5	0.415D-22	0.372D-20	48	42	142	105
0.6	0.127D-10	0.117D-09	48	34	125	83
0.7	0.137D-19	0.270D-09	26	58	56	123
0.8	0.841D-22	0.187D-20	29	54	65	102
0.9	0.670D-15	0.147D-09	28	125	60	205
0.95	0.111D-17	0.403D-09	61	193	125	285

4.5. Wood's quartic function:

$$F(x) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2 + 90(x_4 - x_3^2)^2 + (x_3 - 1)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2) + 19.8(x_2 - 1)(x_4 - 1),$$

$$x_0 = (-3, -1, -3, -1).$$

4.6. Powell's 2nd function:

$$F(x) = 3 - 1/[1 - (x_1 - x_2)^2] - \sin(\pi x_2 x_3/2) - \exp\{-[(x_1 + x_2)/x_2 - 2]^2\}.$$

$$x_0 = (0, 1, 2).$$

Table 8. Rosenbrock's function, $N = 2$.

	Function value		Iterations		Function calls	
P	EQR	CG	EQR	CG	EQR	CG
0.1	0.238D-16	0.270D-11	44	28	130	66
0.2	0.109D-12	0.384D-10	44	23	122	54
0.3	0.203D-15	0.118D-09	44	28	120	66
0.4	0.718D-10	0.119D-10	49	30	130	69
0.5	0.468D-18	0.112D-09	51	28	127	65
0.6	0.435D-08	0.112D-09	52	28	128	65
0.7	0.508D-17	0.216D-22	49	66	82	128
0.8	0.268D-11	0.455D-07	43	62	70	130
0.9	0.296D-11	0.427D-07	45	141	75	272
0.95	0.166D-19	0.256D-08	53	283	83	499

4.7. Beale's function:

$$F(x)=\sum_{i=1}^{3}[c_i-x_i(1-x_2^i)]^2,$$

$$c_1=1.5, \qquad c_2=2.25, \qquad c_3=2.625,$$

$$x_0=(10, 10).$$

4.8. Rosenbrock's valley function:

$$F(x)=100(x_2-x_1^2)^2+(x_1-1)^2,$$

$$x_0=(-1.2, 1).$$

5. Conclusions

For more accurate line searches, the correction vector does very little; in fact, sometimes it has a detrimental effect. However, as the search becomes progressively less accurate, it plays an important role. We have considered here an implementation that employs the correction vector in one of many possible ways and experimentation will no doubt yield better implementations.

The projection method of Hestenes (Ref. 5) is an alternative expression of the three basic relations underlying an algorithm; if the initial step is along the gradient direction, it must develop identical directions to our method for a quadratic. Also for a quadratic, when line searches are exact, gradients at different iterates are orthogonal, and our method may be shown to reduce to the conjugate-gradient method. Since the relations (10) or (11) are independent of λ_i, it is clear that both methods generate identical search directions.

For quadratic functions, our method stands in relation to projection methods (Refs. 5, 8–9) in much the same way as the conjugate-gradient method stands in relation to variable-metric methods (Refs. 10–13). For a more detailed discussion of this, see Ref. 14.

The three conjugate-gradient methods discussed in the introduction, when applied to nonquadratic functions and reset every n iterations, can be shown to have n-step quadratic convergence. This result holds for inexact line searches, though the proof requires that the searches become progressively more accurate. It is a reasonable conjecture that our method also has n-step quadratic convergence; and, using the device of the correction vector, it may be possible to achieve this without progressively more exact searches. This and the numerical properties of our method are currently under study.

References

1. HESTENES, M. R., and STIEFEL, E., *Methods of Conjugate Gradients for Solving Linear Systems*, Research Journal of the National Bureau of Standards, Vol. 49, pp. 409–436, 1952.
2. FLETCHER, R., and REEVES, C. M., *Function Minimization by Conjugate Gradients*, Computer Journal, Vol. 7, pp. 149–154, 1964.
3. FLETCHER, R., *A FORTRAN Subroutine for Minimization by the Method of Conjugate Gradients*, Atomic Energy Research Establishment, Harwell, Oxfordshire, England, Report No. R-7073, 1972.
4. POLAK, E., *Computational Methods in Optimization*, Academic Press, New York, New York, 1971.
5. HESTENES, M. R., *Multiplier and Gradient Methods*, Computing Methods in Optimization Problems, Vol. 2, Edited by L. A. Zadeh, L. W. Neustadt, and A. V. Balakrishnan, Academic Press, New York, New York, 1969.
6. POWELL, M. J. D., *Recent Advances in Unconstrained Optimization*, Mathematical Programming, Vol. 1, pp. 26–57, 1971.
7. DAVIDON, W. C., *Optimally Conditioned Optimization Algorithms Without Linear Searches*, Mathematical Programming, Vol. 9, pp. 1–30, 1975.
8. POWELL, M. J. D., *An Iterative Method for Finding Stationary Values of a Function of Several Variables*, Computer Journal, Vol. 7, pp. 303–307, 1962.
9. ZOUTENDIJK, G., *Methods of Feasible Directions*, Elsevier Publishing Company, Amsterdam, Holland, 1960.
10. DAVIDON, W. C., *Variable Metric Methods for Minimization*, Atomic Energy Commission, Argonne National Laboratory, Argonne, Illinois, Research and Development Report No. ANL-5990, 1959.
11. FLETCHER, R., and POWELL, M. J. D., *A Rapidly Convergent Method for Minimization*, Computer Journal, Vol. 6, pp. 163–168, 1963.
12. BROYDEN, C. G., *The Convergence of a Class of Double Rank Minimization Algorithms*, Journal of the Institute of Mathematics and its Applications, Vol. 6, pp. 79–90, 1970.
13. HUANG, H. Y., *A Unified Approach to Quadratically Convergent Algorithms for Function Minimization*, Journal of Optimization Theory and Applications, Vol. 5, pp. 405–423, 1970.
14. NAZARETH, L., *Unified Approach to Unconstrained Minimization via Basic Matrix Factorizations*, Journal of Linear Algebra and its Applications, Vol. 17, pp. 197–232, 1977.