Conjugate-Gradient Methods for Large-Scale Minimization in Meteorology

I. M. NAVON

Supercomputer Computations Research Institute, Florida State University, Tallahassee, FL 32306

DAVID M. LEGLER

Mesoscale Air–Sea Interaction Group, Department of Meteorology, Florida State University, Tallahassee, FL 32306

(Manuscript received 5 May 1986, in final form 31 October 1986)

ABSTRACT

During the last few years new meteorological variational analysis methods have evolved, requiring large-scale minimization of a nonlinear objective function described in terms of discrete variables. The conjugate-gradient method was found to represent a good compromise in convergence rates and computer memory requirements between simpler and more complex methods of nonlinear optimization. In this study different available conjugate-gradient algorithms are presented with the aim of assessing their use in large-scale typical minimization problems in meteorology. Computational efficiency and accuracy are our principal criteria.

Four different conjugate-gradient methods, representative of up-to-date available scientific software, were compared by applying them to two different meteorological problems of interest using criteria of computational economy and accuracy. Conclusions are presented as to the adequacy of the different conjugate-gradient algorithms for large-scale minimization problems in different meteorological applications.

1. Introduction

The last few years saw the emergence of several new variational analysis methods in meteorology necessitating large-scale nonlinear minimization. Such were the methods of augmented-Lagrangian (Navon, 1982; Le-Dimet, 1982; Navon and de Villiers, 1983; Navon, 1983; Le-Dimet and Segot, 1985; Navon, 1985; Navon and de Villiers, 1986; Navon, 1986; Le-Dimet and Talagrand, 1986). Hoffman (1982, 1984) proposed a variational analysis method for SeaSat-A satellite scatterometer winds. In his work a functional of 10^4 variables had to be minimized using a conjugate-gradient method. Finally, a new method—the adjoint model technique—based on optimal-control theory, pioneered by Marchuk (1974) and described by Kontarev (1980), has recently been used by Talagrand (1985), Courtier (1985), Le-Dimet and Talagrand (1986), Derber (1985), and Lewis and Derber (1985) for four-dimensional data assimilation. Again, in this method large-scale minimization of a discrete functional with respect to one of the analysis states is required. Derber (1985) used a conjugate-gradient method when implementing the adjoint model technique. The conjugate-gradient method has also been used in a variational analysis by Testud and Chong (1983).

Different nonlinear minimization algorithms are available (see, for instance, Gill et al., 1981). On one hand, we have simple methods such as the steepest descent method, which converges more slowly than the conjugate-gradient algorithms, while on the other hand we have Newton and quasi-Newton methods, which have quadratic rates of convergence and superlinear rates of convergence, respectively, but require storage of Hessian matrices of second derivatives of size $(N \times N)$.

Conjugate-gradient algorithms require storage of only a few vectors of length N, where N is the number of variables of the discretized meteorological model. For large meteorological models N easily exceeds 10^4, and in such cases the conjugate-gradient method with its better rate of convergence is almost the only choice open.

The conjugate-gradient algorithm is an iterative method for unconstrained minimization that produces a better approximation to the minimum of a general unconstrained nonlinear function of N variables, x_1, $x_2, \ldots, x_N$ with each iteration.

Within a given iteration an estimate is made of the best way to change each component of the vector $\mathbf{x}$, so as to produce the maximum reduction of the function, by finding the gradient of the function with respect to the variables and combining this gradient with information from the previous iterations to produce a search direction. The search direction is an estimate of the relative change in each component of the vector $\mathbf{x}$ to produce the maximum reduction in the function F. To find the magnitude of the changes along the search direction, an optimal step size must be estimated. The new vector after an iteration of the conjugate gradient $\mathbf{x}_{k+1}$ is given by the previous vector $\mathbf{x}_k$ plus an optimal step size times the search direction.

The conjugate-gradient method has been applied in variational analysis to minimize an objective function, measuring the lack of fit to data and constraints with the aim to remove the ambiguity of the SeaSat-A satellite scatterometer (SASS) winds by Hoffman (1982, 1984). It was chosen because it is the fastest minimization method that can be implemented when the number of variables, *n*, is large and because it requires only between six and eight *n*-vectors of storage.

Based on a suggestion by Le Dimet and Talagrand (1986), Derber (1985) used a method based on optimal control theory to variationally perform four-dimensional data assimilation by adjusting only the initial fields. The variational adjustment technique uses the conjugate-gradient algorithm to iteratively find a solution to a forecast model that best fits the data. The best fit is defined in terms of a functional that is a weighted sum of squared differences between the model solution and prespecified analyses. Each iteration of the minimization algorithm (here, the conjugate-gradient) requires the calculation of the gradient of the functional with respect to the initial conditions.

The augmented-Lagrangian method is an algorithm in which an original constrained minimization problem is transformed into a sequence of unconstrained minimization problems. Le Dimet and Segot (1985) applied the method for variational analysis of the wind field and geopotential at 500 mb using as a constraint the steady-state shallow-water equations. The minimization of the augmented-Lagrangian functional associated with the variational analysis was carried out using a conjugate-gradient method.

Navon and de Villiers (1983) used an augmented-Lagrangian algorithm to maintain integral constraints in the time integration of a meteorological numerical model. Again, the minimization of the augmented-Lagrangian functional was carried out using a conjugate-gradient method.

We see that a deeper understanding of conjugate-gradient methods is a necessary prerequisite for researchers interested in the application of variational analysis methods in meteorology.

In the present paper we aim to bring to the attention of the meteorological community recent experience with different conjugate-gradient methods for large-scale minimization, applied to meteorological problems of interest, as well as to introduce first-time users to the basics of the conjugate-gradient methods.

The plan of the paper is the following. In section 2, an introduction to the basics of conjugate-gradient methods is presented. In section 3, different conjugate-gradient methods are detailed, representing, as a function of time and availability of corresponding software, increasing sophistication in the methods. In section 4, four different conjugate-gradient methods are compared, applied to two different meteorological problems. One of these problems concerns enforcing of conservation of integral invariants via the augmented-Lagrangian method, while the other arises from a problem in objective analysis of wind-stress data [Legler et al., 1987 (in preparation)]. The comparison emphasizes accuracy and computational efficiency as a function of the number of function and gradient evaluations, storage and actual CPU used. In section 5, new developments and alternative methods for large-scale minimization are briefly discussed. Conclusions are presented in section 6.

2. Conjugate-gradient methods

The conjugate-gradient method, or more generally, the conjugate-direction method, was initiated by Hestenes and Stieffel (1952). For a full account of the historical development of the method see Hestenes (1980). To introduce the topic here, the concept of *conjugacy* will be defined first and a few useful theorems stated without proof. The interested reader will be directed to adequate literature (Fletcher, 1980; Walsh, 1975; Himmelblau, 1972).

a. Conjugacy

Consider a quadratic function given by

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x} + \mathbf{b}^T\mathbf{x} + c \tag{1}$$

where $\mathbf{G}$ is a positive definite symmetric matrix, $\mathbf{b}$ a vector, and c a scalar. Then the directions represented by the two vectors $\mathbf{u} \neq 0$ and $\mathbf{v} \neq 0$ are conjugate with respect to $\mathbf{G}$ if

$$\mathbf{u}^T\mathbf{G}\mathbf{v} = 0. \tag{2}$$

Geometrically (for simplicity in two-dimensions), the level curves

$$F(\mathbf{x}) = \gamma \tag{3}$$

for different values of γ are concentric ellipses. The concept of conjugacy has its origin in the theory of poles and polars of an ellipse (see Hestenes, 1980). Some useful theorems without proof will be formulated here on the properties of conjugate directions.

Theorem 1. *If the vectors* $\mathbf{d}_j$ *are mutually conjugate (i.e.,* $\mathbf{d}_i{}^T\mathbf{G}\mathbf{d}_j = 0$ *for* $i \neq j$*, for all i and j), then they are linearly independent.*

It follows that there exists at least one set of n independent vectors mutually conjugate with respect to the matrix $\mathbf{G}$; the set of eigenvectors of $\mathbf{G}$ forms such a set. Let us now look at minimizing $F(\mathbf{x})$ subject to $\mathbf{x} \in \mathbb{R}^n$.

Theorem 2. *Let* $\mathbf{x}_k$ *and* $\mathbf{x}_{k+1}$ *be consecutive current points in a minimization of* $F(\mathbf{x})$*. If (i)* $\mathbf{x}_k$ *minimizes* $F(\mathbf{x})$ *in direction* $\mathbf{d}_l$*, (ii)* $\mathbf{x}_{k+1}$ *minimizes* $F(\mathbf{x})$ *in the direction* $\mathbf{d}_m$*, (iii)* $\mathbf{d}_l$ *and* $\mathbf{d}_m$ *are conjugate-directions, then* $\mathbf{x}_{k+1}$ *also minimizes* $F(\mathbf{x})$ *in the direction* $\mathbf{d}_l$*.*

If we denote by $\mathbf{g}_k$:

$$\mathbf{g}_k = \nabla F(\mathbf{x}_k) \tag{4}$$

the gradient of $F(\mathbf{x}_k)$, then condition (i) and condition

(iii) imply $g^T_{l+1}\mathbf{d}_i = 0$ for each $i = 0, \cdots, l$, (for proof see appendix E) and

$$\mathbf{d}_l{}^T\mathbf{G}\mathbf{d}_m = 0. \quad (5)$$

For the quadratic function F we have

$$\mathbf{g}_{k+1} - \mathbf{g}_k = \mathbf{G}(\mathbf{x}_{k+1} - \mathbf{x}_k) \quad (6)$$

and from (ii)

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{d}_k \quad (7)$$

where α_k is determined by the line minimization

$$F(\mathbf{x}_k + \alpha_k\mathbf{d}_k) = \min_\alpha F(\mathbf{x}_k + \alpha\mathbf{d}_k). \quad (8)$$

Theorem 3. *Let* $\mathbf{d}_i$, $i = 1, \cdots, m$ $(m \leqslant n)$ *be mutually conjugate directions. Then the global minimum of* $F(\mathbf{x})$ *can be found from an arbitrary starting point* $(\mathbf{x}^0)$ *by a finite descent computation in which each of the* $\mathbf{d}_i$ *is used as a descent direction only once.*

An algorithm that uses mutually conjugate directions of search possesses the property of quadratic termination.

b. *Construction of a set of mutually conjugate directions*

Given a set of linearly independent vectors $\mathbf{v}_0, \ldots, \mathbf{v}_{n-1}$, one can construct a set of mutually $\mathbf{G}$-conjugate directions $\mathbf{d}_0, \ldots, \mathbf{d}_{n-1}$ by the following procedure. Set

$$\mathbf{d}_0 = \mathbf{v}_0 \quad (9)$$

and then for $i = 1, \cdots, n-1$ successively define

$$\mathbf{d}_i = \mathbf{v}_i + \sum_{j=0}^{i-1} a_{ij}\mathbf{d}_j \quad (10)$$

where a_{ij} are coefficients chosen so that $\mathbf{d}_i$ is $\mathbf{G}$-conjugate to the previous directions $\mathbf{d}_{i-1}, \mathbf{d}_{i-2}, \ldots, \mathbf{d}_0$. This is possible if, for $l = 0, \cdots, i-1$,

$$\mathbf{d}_i{}^T\mathbf{G}\mathbf{d}_l = \mathbf{v}_i{}^T\mathbf{G}\mathbf{d}_l + \sum_{j=0}^{i-1} a_{ij}\mathbf{d}_j{}^T\mathbf{G}\mathbf{d}_l = 0. \quad (11)$$

If previous coefficients a_{ij} were chosen so that at $\mathbf{d}_0, \ldots, \mathbf{d}_{i-1}$ are $\mathbf{G}$-conjugate, then we have

$$\mathbf{d}_j{}^T\mathbf{G}\mathbf{d}_l = 0 \quad \text{if} \quad j \neq l \quad (12)$$

and from (11) we get

$$a_{ij} = -\frac{\mathbf{v}_i{}^T\mathbf{G}\mathbf{d}_j}{\mathbf{d}_j{}^T\mathbf{G}\mathbf{d}_j} \quad \text{for all} \quad i = 1, \cdots, n-1$$
$$j = 0, \cdots, i-1. \quad (13)$$

Thus the set of directions $\mathbf{d}_0, \ldots, \mathbf{d}_{n-1}$ defined by (9)–(13) is $\mathbf{G}$-conjugate and the subspaces spanned by $\mathbf{d}_0, \ldots, \mathbf{d}_i$ and by $\mathbf{v}_0, \ldots, \mathbf{v}_i$ are the same.

c. *The conjugate-gradient method*

If in the constructive procedure described above we take

$$\mathbf{v}_0 = -\mathbf{g}_0 \quad (14)$$

(i.e., the initial step is in the direction of the steepest descent) and the rest of the vectors are $\mathbf{v}_1 = -\mathbf{g}_1, \cdots, \mathbf{v}_{n-1} = -\mathbf{g}_{n-1}$, we then find

$$\mathbf{x}_1 = \mathbf{x}_0 + \alpha_0\mathbf{d}_0 \quad (15)$$

by a line search. [Here α_0 is the value of α that minimizes $F(\mathbf{x}_0 + \alpha\mathbf{d}_0)$.] The second direction is then d_1, found by using (10), (11) and (9) with $\mathbf{v}_0 = -\mathbf{g}_0$ and $\mathbf{v}_1 = -\mathbf{g}_1$. This gives

$$\mathbf{d}_1 = -\mathbf{g}_1 + \frac{g_1{}^T\mathbf{G}\mathbf{d}_0}{\mathbf{d}_0{}^T\mathbf{G}\mathbf{d}_0}\mathbf{d}_0 \quad (16)$$

and using equation (6)

$$\mathbf{g}_1 - \mathbf{g}_0 = \mathbf{G}(\mathbf{x}_1 - \mathbf{x}_0) = \alpha_0\mathbf{G}\mathbf{d}_0. \quad (17)$$

We can write (16) as

$$\mathbf{d}_1 = -\mathbf{g}_1 + \frac{\mathbf{g}_1{}^T(\mathbf{g}_1 - \mathbf{g}_0)}{\mathbf{d}_0{}^T(\mathbf{g}_1 - \mathbf{g}_0)}\mathbf{d}_0. \quad (18)$$

Repeating the procedure with $\mathbf{v}_0 = -\mathbf{g}_0$, $\mathbf{v}_1 = -\mathbf{g}_1$, and $\mathbf{v}_k = -\mathbf{g}_k$, we obtain at step $(k+1)$:

$$\mathbf{d}_k = -\mathbf{g}_k + \sum_{j=0}^{k-1} \frac{\mathbf{g}_k{}^T\mathbf{G}\mathbf{d}_j}{\mathbf{d}_j{}^T\mathbf{G}\mathbf{d}_j}\mathbf{d}_j \quad (19)$$

or

$$\mathbf{d}_k = -\mathbf{g}_k + \sum_{j=0}^{k-1} \frac{\mathbf{g}_k{}^T(\mathbf{g}_{j+1} - \mathbf{g}_j)}{\mathbf{d}_j{}^T(\mathbf{g}_{j+1} - \mathbf{g}_j)}\mathbf{d}_j. \quad (20)$$

Using the fact that the subspace spanned by $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1}$ is also the subspace spanned by $\mathbf{d}_0, \ldots, \mathbf{d}_{k-1}$ and the fact that

$$\mathbf{g}_k{}^T\mathbf{d}_j = 0, \quad j = 0, \cdots, k-1 \quad (21)$$

we obtain

$$\mathbf{g}_k{}^T\mathbf{g}_j = 0, \quad j = 0, \cdots, k-1 \quad (22)$$

so that (20) reduces to the much simpler formula

$$\mathbf{d}_k = -\mathbf{g}_k + \beta_k\mathbf{d}_{k-1} \quad (23)$$

with

$$\beta_k = \frac{\mathbf{g}_k{}^T(\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{d}^T_{k-1}(\mathbf{g}_k - \mathbf{g}_{k-1})}. \quad (24)$$

Using the equalities

$$\mathbf{g}_k{}^T\mathbf{g}_j = \mathbf{g}_k{}^T\mathbf{d}_j = 0 \quad j = 0, \cdots, k-1 \quad (25)$$

$$\mathbf{d}_{k-1} = -\mathbf{g}_{k-1} + \beta_{k-1}\mathbf{d}_{k-2} \quad (26)$$

the coefficient β_k can be written as

$$\beta_k = \frac{\mathbf{g}_k{}^T(\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{g}^T_{k-1}\mathbf{g}_{k-1}} = \frac{\mathbf{g}_k{}^T\mathbf{g}_k}{\mathbf{g}^T_{k-1}\mathbf{g}_{k-1}}. \quad (27)$$

The important conclusion is that, *in order to generate the direction* $\boldsymbol{d}_k$ *in the minimization process,* we need only know current and next previous gradients $\mathbf{g}_k$ and $\mathbf{g}_{k-1}$ and the previous direction $\mathbf{d}_{k-1}$ (i.e., storage of three vectors). Another observation is that for the conjugate-gradient method described we use only *vector* formulae. This issue is of vital significance when n is large and problems of computer storage preclude other methods. This method, applied to a positive definite quadratic function of n variables, will find the minimum in, at most, n iterations. (For proof see appendix D.)

3. Different conjugate-gradient methods

a. The Fletcher–Reeves algorithm (1964)

Fletcher and Reeves (1964), based on previous work of Hestenes and Steiffel (1952), proposed a conjugate-gradient method applied to general nonlinear functions. When used on nonquadratic functions the method loses the property of finding the minimum in, at most, n iterations and becomes an iterative method.

Fletcher and Reeves (1964) suggest that the direction of search should revert periodically (after a cycle of n linear searches) to the direction of steepest descent, all previous information on directions being discarded.

For the line searches the cubic interpolation method of Davidon (1959) was used. (See also appendix A.) Fletcher and Reeves (1964) initially used a rather stringent convergence criterion of stopping the minimization process only when a complete cycle of (n + 1) iterations, starting with a steepest descent search, produces no further reduction in the value of the objective function.

Their general algorithm is for the following problem:

$$\min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x})$$

$$\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n)$$

$$\mathbf{x}_0 \quad \text{arbitrary} \tag{28}$$

and proceeds in the following steps:

1) Compute

$$\mathbf{g}_0 = \mathbf{g}(\mathbf{x}_0) = \nabla F(\mathbf{x}_0). \tag{29}$$

Set

$$\mathbf{d}_0 = -\mathbf{g}_0 \quad \text{(steepest descent)}. \tag{30}$$

2) For k = 1, 2, $\cdots$, n + 1, set

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k \tag{31}$$

[position of minimum of $F(\mathbf{x})$ on the line through $\mathbf{x}_k$ in the direction $\mathbf{d}_k$], where α_k is obtained by a line search

$$F(\mathbf{x}_k + \alpha_k \mathbf{d}_k) = \min_{\alpha} F(\mathbf{x}_k + \alpha \mathbf{d}_k) \tag{32}$$

using Davidon's (1959) cubic interpolation method for linear search of the stepsize.

3) Compute

$$\mathbf{g}_{k+1} = \nabla F(\mathbf{x}_{k+1}) \tag{33}$$

4) Generate a new search direction $\mathbf{d}_{k+1}$ by

$$\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k \tag{34}$$

where

$$\beta_k = \frac{\mathbf{g}^{\mathrm{T}}_{k+1}\mathbf{g}_{k+1}}{\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_k} = \frac{\mathbf{g}^2_{k+1}}{\mathbf{g}_k^{\,2}}. \tag{35}$$

5) A convergence criterion for stopping the iterations is tested ($\|\mathbf{g}^k| \leqslant \epsilon$), and, if satisfied, the iterations must stop.

6) Otherwise, continue for n iterations and then reset the direction to $\mathbf{d}^{(n+1)} = -\mathbf{g}^{(n+1)}$ in the steepest descent direction and start another cycle of n iterations.

b. A nonlinear example

Minimize

$$F(x,y) = x - \ln x + y^2 \tag{36}$$

or using a different notation find

$$\min F(x_1, x_2) = x_1 - \ln x_1 + x_2{}^2. \tag{37}$$

The exact solution is

$$\mathbf{x}^* = (1,0) \tag{38}$$

i.e.,

$$x_1 \equiv x = 1, \quad x_2 \equiv y = 0. \tag{39}$$

Case a: A nearby starting point

$$\mathbf{x}^0 = (1+\epsilon, \epsilon) \tag{40}$$

$$\mathbf{d}_0 = -\mathbf{g}_0 = -\nabla F(\mathbf{x}^0)$$

$$= -\left(\frac{\epsilon}{1+\epsilon}, 2\epsilon\right)^{\mathrm{T}} \approx (-\epsilon, -2\epsilon)^{\mathrm{T}} + \mathrm{O}(\epsilon^2). \tag{41}$$

A step-size scalar α, which minimizes $F(\mathbf{x}^0 + \alpha\mathbf{d}_0)$, is approximated by

$$\alpha_0 = \frac{\nabla F_0{}^{\mathrm{T}}\mathbf{d}_0}{\mathbf{d}_0{}^{\mathrm{T}}\nabla^2 F_0 \mathbf{d}_0} \tag{42}$$

where we use the notation $\mathbf{G} = \left(\dfrac{\partial^2 F}{\partial x_i \partial x_j}\right) \equiv \nabla^2 F$ for the Hessian matrix of F (see also appendix D), resulting in

$$\alpha_0 = \frac{5}{9} + \mathrm{O}(\epsilon). \tag{43}$$

Then

$$\mathbf{x}^1 = \mathbf{x}^0 + \alpha_0 \mathbf{d}_0 = \left(1 + \epsilon - \epsilon\cdot\frac{5}{9}, \epsilon - 2\epsilon\cdot\frac{5}{9}\right)$$

$$= \left(1 + \frac{4}{9}\epsilon, -\frac{1}{9}\epsilon\right) + \mathrm{O}(\epsilon^2). \tag{44}$$

Then the new conjugate-gradient direction is

$$\mathbf{d}_1 = -\mathbf{g}(\mathbf{x}^1) + \beta_0 \mathbf{d}_0 \tag{45}$$

$$\beta_0 = \frac{\mathbf{g}(\mathbf{x}^1)^2}{\mathbf{g}(\mathbf{x}^0)^2} \tag{46}$$

i.e.,

$$\mathbf{d}_1 = \left(-\frac{40}{81}\epsilon, \frac{10}{81}\epsilon\right)^{\mathrm{T}} + \mathrm{O}(\epsilon^2). \tag{47}$$

We can then calculate

$$\mathbf{x}_2 = \mathbf{x}_1 + \alpha_1 \mathbf{d}_1 \tag{48}$$

and α_1 is obtained from

$$\min_{\alpha}(\mathbf{x}_1+\alpha\mathbf{d}_1) \tag{49}$$

resulting in

$$\alpha_1=\frac{9}{10}+\mathrm{O}(\epsilon).$$

Then

$$\mathbf{x}_2=\mathbf{x}_1+\alpha_1\mathbf{d}_1=\left[\frac{4}{9}\epsilon+1+\frac{9}{10}\left(-\frac{40}{81}\epsilon\right),-\frac{1}{9}\epsilon+\frac{9}{10}\left(\frac{10}{81}\epsilon\right)\right]$$

$$=(1,0)+\mathrm{O}(\epsilon^2). \tag{50}$$

Case b: Suppose we start further away, i.e., suppose

$$\mathbf{x}_0=(1.12,0.05) \tag{51}$$

$$\mathbf{d}_0=-\mathbf{g}_0=(-0.107,-0.10). \tag{52}$$

Then α is the point where $\partial F/\partial\alpha \approx 0$ and where $(d^2f/d\alpha^2)\,(\mathbf{x}^0+\alpha\mathbf{d}_0)>0$ and using (42) we get $\alpha=0.7202$:

$$\mathbf{x}_1=\mathbf{x}_0+\alpha_0\mathbf{d}_0=(1.043,-0.02202) \tag{53}$$

$$\mathbf{d}_1=-\mathbf{g}(\mathbf{x}_1)+\beta_0\mathbf{d}_0=-g(\mathbf{x}_1)+\frac{\mathbf{g}(\mathbf{x}_1)^2}{\mathbf{g}(\mathbf{x}^0)^2}\mathbf{d}_0$$

$$=(-0.0605,0.02596). \tag{54}$$

Following this line and calculating α_1, we finally find

$$\mathbf{x}_1+\alpha_1\mathbf{d}_1=\mathbf{x}_2=(0.9977,-0.00264). \tag{55}$$

c. *The conjugate-gradient method as a linear equations solver*

The conjugate-gradient method can also be used as a method of solving a set of positive definite symmetric linear equations. Hestenes and Steiffel (1952) used it first for this purpose.

Take the quadratic function

$$F(\mathbf{x})=\frac{1}{2}\mathbf{x}^{\mathrm{T}}A\mathbf{x}+\mathbf{b}^{\mathrm{T}}\mathbf{x}+c \tag{56}$$

where $\mathbf{A}(n\times n)$ is symmetric, positive definite matrix and $\mathbf{b}(n\times 1)$ a vector. Then the condition for a minimum is

$$\nabla F=\mathbf{g}=0 \tag{57}$$

or as

$$\mathbf{g}=\nabla_x F(\mathbf{x})=\mathbf{A}\mathbf{x}+\mathbf{b} \tag{58}$$

$$\mathbf{g}=0\Leftrightarrow\mathbf{A}\mathbf{x}=-\mathbf{b}. \tag{59}$$

When used to solve system (59) the conjugate-gradient method is known as the linear conjugate-gradient method.

The linear conjugate-gradient method only uses products of a matrix with a vector and does not require the elements of the matrix explicitly. One can also solve for a positive definite matrix $\mathbf{R}$ without requiring it to be symmetric by using

$$\mathbf{A}=\mathbf{R}^{\mathrm{T}}\mathbf{R}. \tag{60}$$

This property is very useful for solving systems of linear equations arising from the discretization of elliptic partial differential equations. For the linear conjugate-gradient method let us denote by $\mathbf{r}_j$ the residual

$$\mathbf{r}_j=\mathbf{A}\mathbf{x}_j+\mathbf{b} \tag{61}$$

and, if we adopt the convention,

$$\beta_{-1}=\mathbf{d}_{-1}=0.$$

The linear conjugate-gradient algorithm takes the form, for $k=0,1,\cdots$, given $\mathbf{x}_0$ and

$$\mathbf{r}_0=\mathbf{b}+\mathbf{A}\mathbf{x}_0, \tag{62}$$

then for $k=0,1,\cdots$,

$$\mathbf{d}_k=-\mathbf{r}_k+\beta_{k-1}\mathbf{d}_{k-1} \tag{63}$$

$$\alpha_k=\frac{\|\mathbf{r}_k\|_2^2}{\mathbf{d}_k^{\mathrm{T}}\mathbf{A}\mathbf{d}_k} \tag{64}$$

$$\mathbf{x}_{k+1}=\mathbf{x}_k+\alpha_k\mathbf{d}_k \tag{65}$$

$$\mathbf{r}_{k+1}=\mathbf{r}_k+\alpha_k\mathbf{A}\mathbf{d}_k \tag{66}$$

with

$$\beta_k=\frac{\|\mathbf{r}_{k+1}\|_2^2}{\|\mathbf{r}_k\|_2^2}. \tag{67}$$

It can be shown (Hestenes, 1980), that the linear conjugate-gradient algorithm will obtain the solution of

$$\mathbf{A}\mathbf{x}=\mathbf{b} \tag{68}$$

in m ($m\leqslant n$), where m is the number of distinct eigenvalues of $\mathbf{A}$.

The method will converge even quicker if the eigenvalues of $\mathbf{A}$ are clustered into groups of approximately equal number (see, for instance, Gill et al., 1981; Bertsekas, 1982; Hestenes, 1980). For more details see Golub and Van Loan (1983) and Concus et al. (1976), Manteuffel (1979) and Neethling (1977).

d. *The Polak–Ribiere algorithm*

The Polak–Ribiere (1969) conjugate-gradient algorithm differs from the Fletcher–Reeves algorithm only in the formula for β_k proposing

$$\beta_{k_{\mathrm{P.R.}}}=\frac{\mathbf{g}_{k+1}^{\mathrm{T}}(\mathbf{g}_{k+1}-\mathbf{g}_k)}{\mathbf{g}_k{}^{\mathrm{T}}\mathbf{g}_k}. \tag{69}$$

The Polak–Ribiere algorithm coincides with the Fletcher–Reeves algorithm whenever it is applied to a quadratic function, but differs from it when applied to nonlinear functions. (In the quadratic case it is easily verified that $\mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{g}_k=0$.) Due to nonquadratic terms in the objective function and possible inaccurate line searches, conjugacy of the directions generated by the conjugate-gradient method will be progressively lost and a situation can occur where the conjugate-gradient method will "jam" (see also Powell, 1976). This means that a generated search direction $\mathbf{d}_k$ is almost orthogonal to the gradient $\nabla\mathbf{F}(\mathbf{x}_k)$. When this occurs,

$$\mathbf{g}(\mathbf{x}_{k+1}) = \nabla F(\mathbf{x}_{k+1}) \approx \nabla F(\mathbf{x}_k) = \mathbf{g}_k. \tag{70}$$

In this case β_{k+1} following Polak and Ribiere will be nearly zero, and the next direction, $\mathbf{d}_{k+1}$ generated by

$$\mathbf{d}_{k+1} = -\nabla F(\mathbf{x}_{k+1}) + \beta_{k+1}\mathbf{d}_k, \tag{71}$$

will be close to $-\nabla F(\mathbf{x}_{k+1})$ and, therefore, will break the jam. This is not true for the β_k proposed by Fletcher and Reeves. The Polak–Ribiere algorithm is given as (see Polak, 1971)

Step 0: Select $\mathbf{x}_0 \in \mathbb{R}^m$. If $\nabla F(\mathbf{x}_0) = 0$ Stop. Else go to Step 1.

Step 1: Set $i = 0$ and $\mathbf{d}_0 = -\nabla F(\mathbf{x}_0)$. (72)

Step 2: Compute $\alpha_i > 0$ such that

$$F(\mathbf{x}_i + \alpha_i\mathbf{d}_i) = \min_{\alpha \geqslant 0}\{F(\mathbf{x}_i + \alpha\mathbf{d}_i)\}. \tag{73}$$

Step 3: Set $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i\mathbf{d}_i$. (74)

Step 4: Compute $\nabla F(\mathbf{x}_{i+1})$. (75)

Step 5: If $\nabla F(\mathbf{x}_{i+1}) = 0$ Stop. Else set

$$\mathbf{g}_{i+1} = -\nabla F(\mathbf{x}_{i+1}) \tag{76}$$

$$\mathbf{d}_{i+1} = \mathbf{g}_{i+1} + \beta_i\mathbf{d}_i \tag{77}$$

$$\beta_i = \frac{\mathbf{g}_{i+1}^{\mathrm{T}}(\mathbf{g}_{i+1} - \mathbf{g}_i)}{\mathbf{g}_i^{\mathrm{T}}\mathbf{g}_i} \tag{78}$$

and return to Step 2.

e. Beale's restart method

As was remarked previously, regardless of the different ways of calculating the scalar β_k, when we are minimizing a nonquadratic function and when the number of variables n is large (as is the case in meteorological problems), the conjugate-gradient method can generate inefficient directions of search after few iterations. The first applications operated the method in cycles of conjugate directions steps, using as the first step in each cycle a steepest descent step. However, as was pointed out by Powell (1977), there is a disadvantage in using the steepest descent direction as a restarting direction in the conjugate-gradient method—namely, that the short-term effect of restarting is poorer than if no restarting took place.

A more satisfactory restart procedure was proposed by Beale (1972). He asked the following question: If $\mathbf{d}_t$ is an arbitrary downhill restarting direction, if $F(\mathbf{x})$ is a quadratic function, and if the search direction $\mathbf{d}_k$ ($k > t$) is to be a linear combination of $\mathbf{d}_t$ and the calculated gradients $\mathbf{g}_{t+1}, \mathbf{g}_{t+2}, \ldots, \mathbf{g}_k$, what is the linear combination that makes the sequence of search directions, $\mathbf{d}_t, \mathbf{d}_{t+1}, \mathbf{d}_{t+2}, \ldots$, mutually conjugate? Beale (1972) has shown it to be

$$\mathbf{d}_k = -\mathbf{g}_k + \beta_k\mathbf{d}_{k-1} + \gamma_k\mathbf{d}_t \tag{79}$$

which can be viewed as the classical conjugate-gradient with the addition of a multiple of the restarting direction $\mathbf{d}_t$.

In Beale's (1972) method, β_k and γ_k take the values:

$$\beta_k = \frac{(\mathbf{g}_{k+1} - \mathbf{g}_k)^{\mathrm{T}}\mathbf{g}_{k+1}}{(\mathbf{g}_{k+1} - \mathbf{g}_k)^{\mathrm{T}}\mathbf{d}_k} \tag{80}$$

$$\gamma_k = \frac{(\mathbf{g}_{t+1} - \mathbf{g}_t)^{\mathrm{T}}\mathbf{g}_{k+1}}{(\mathbf{g}_{t+1} - \mathbf{g}_t)^{\mathrm{T}}\mathbf{d}_t}. \tag{81}$$

The direction $\mathbf{d}_t$ is known as the *restart direction* and is the last direction of the previous conjugate-gradient cycle along which a linear search was made. For a derivation of Beale's restart method, see appendix B.

f. Powell's (1977) restart method

Powell (1977) has suggested a condition for restarting in the conjugate-gradient algorithm based upon the property that the gradient vectors are mutually orthogonal in the minimization of a quadratic function F. He proposes that a restart shall take place if the inequality

$$|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k+1}| \geqslant 0.2\|\mathbf{g}_{k+1}\|^2 \tag{82}$$

holds or if there have been n linear searches in the given particular conjugate-gradient cycle. The search direction is defined by Beale's Eqs. (80) and (81), except that $\gamma_k = 0$ when $k = t + 1$. A restart will also take place if the search direction $\mathbf{d}_{k+1}$ is not sufficiently downhill, an adequate downhill direction being one that satisfies the two inequalities:

$$\mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{d}_{k+1} \leqslant -0.8\|\mathbf{g}_{k+1}\|^2 \tag{83}$$

$$\mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{d}_{k+1} \geqslant -1.2\|\mathbf{g}_{k+1}\|^2. \tag{84}$$

The storage required by Powell's (1977) restart algorithm exceeds that of the Polak–Ribiere (1969) and Fletcher–Reeves (1964) methods because after a restart we need also the vectors $\mathbf{d}_t$ and $(\mathbf{g}_{t+1} - \mathbf{g}_t)$. As such, at least six vectors of storage are required by Powell's method, as compared with only four such vectors for Polak and Ribiere's (1969) method and only three vectors in Fletcher and Reeves' (1964) method.

g. The Newton and quasi-Newton methods

A general function expanded about a local minimizer $\mathbf{x}^*$ is approximated well by a quadratic function. Thus, methods based on quadratic function models should have a rapid ultimate rate of convergence. Moreover, the quadratic model is simple and a Taylor series of $F(\mathbf{x})$ about an arbitrary point $\mathbf{x}_k$, taken to quadratic terms, will agree with $F(\mathbf{x})$ to a given accuracy over a much greater neighborhood of $\mathbf{x}_k$ than will the series taken to linear terms.

If first and second derivatives of $F(\mathbf{x})$ are available, a quadratic model of the objective function is obtained by taking the first three terms of a truncated Taylor series expansion about the current point $\mathbf{x}_k$, i.e.,

$$F(\mathbf{x}_k+\mathbf{p}) \approx F(\mathbf{x}_k)+\mathbf{g}_k{}^{\mathrm{T}}\mathbf{p}+\frac{1}{2}\mathbf{p}^{\mathrm{T}}\mathbf{G}(\mathbf{x}_k)\mathbf{p} \qquad (85)$$

where $\mathbf{p} = \mathbf{x} - \mathbf{x}_k$ is the step to the minimum.

The minimum of the right-hand side of Eq. (85) will be achieved if $\mathbf{p}_k$ is the minimum of the quadratic function

$$Q(\mathbf{p})=\mathbf{g}_k{}^{\mathrm{T}}\mathbf{p}+\frac{1}{2}\mathbf{p}^{\mathrm{T}}\mathbf{G}_k\mathbf{p}. \qquad (86)$$

A stationary point $\mathbf{p}_k$ minimizing Eq. (86) will satisfy the linear system

$$\mathbf{G}_k\mathbf{p}_k=-\mathbf{g}_k \qquad (87)$$

giving the Newton method. The solution of (87) is the *Newton direction.* Here $\mathbf{G}_k \equiv \mathbf{G}(\mathbf{x}_k)$, $\mathbf{g}_k \equiv \mathbf{g}(\mathbf{x}_k)$.

Application of the Newton method consists of the iteration

$$\mathbf{x}_{k+1}=\mathbf{x}_k-\alpha_k\mathbf{G}_k{}^{-1}\mathbf{g}_k \qquad (88)$$

where we assume that the inverse of the Hessian matrix, $\mathbf{G}_k{}^{-1}$, exists and that the Newton direction

$$\mathbf{p}_k=-\mathbf{G}_k{}^{-1}\mathbf{g}_k \qquad (89)$$

is a direction of descent, i.e., $\mathbf{p}_k{}^{\mathrm{T}}\mathbf{g}_k < 0$. The Newton method involves the solution of the system of equations

$$\mathbf{G}_k\mathbf{p}_k=-\mathbf{g}_k \qquad (90)$$

to find the Newton direction, and it also requires, zero, first and second derivatives of $F(\mathbf{x})$ to be available at any point. Equation (90), solved by factoring $\mathbf{G}_k = \mathbf{LDL}^{\mathrm{T}}$, requires $\frac{1}{6}n^3 + O(n^2)$ multiplications per iteration.

The Newton method also needs $n^2 + n$ scalar function evaluations. Its storage requirements are to store a $(n \times n)$ symmetric Hessian matrix, i.e., $\frac{1}{2}n^2 + O(n)$ storage locations. Its rate of convergence is second order.

Quasi-Newton methods are descent methods of the form

$$\mathbf{x}_{k+1}=\mathbf{x}_k+\alpha_k\mathbf{d}_k \qquad (91)$$

$$\mathbf{B}_{k+1}\mathbf{d}_{k+1}=-\mathbf{g}_{k+1} \qquad (92)$$

where $\mathbf{B}_{k+1}$ is an approximation to the Hessian matrix. At each iteration, the approximate Hessian is updated by a matrix of rank two.

Quasi-Newton methods are based on the idea of building up curvature information as the iterations of a descent method proceed, using the observed behavior of F and $\mathbf{g}$; i.e., they are based on the idea that an approximation to the curvature of a nonlinear function can be computed without explicitly forming the Hessian matrix.

The quasi-Newton method requires only function and gradient evaluations together with only $O(n^2)$ multiplications per iteration as opposed to $O(n^3)$ multiplications per iteration of the Newton method.

The storage requirement is again for a symmetric approximate $(n \times n)$ Hessian matrix, i.e., $\frac{1}{2}n^2 + O(n)$ storage locations for the approximation to the Hessian matrix, which is the same as for the Newton method. It has, however, only a superlinear rate of convergence. The major advantages of the quasi-Newton method over the Newton method are that the quasi-Newton method requires only first derivatives, whereas for the Newton method the user must supply second derivatives, and that it has a better computational efficiency, i.e., $O(n^2)$ multiplications versus $O(n^3)$ multiplications for the Newton method.

h. *The Shanno (1978) limited-memory quasi-Newton method*

The most popular updating method is the Broyden (1970), Fletcher (1970), Goldfarb (1970), Shanno (1970) (BFGS) formula:

$$\mathbf{B}_{k+1}=\mathbf{B}_k-\frac{1}{\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k\mathbf{s}_k}\mathbf{B}_k\mathbf{s}_k\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k+\frac{1}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{s}_k}\mathbf{y}_k\mathbf{y}_k{}^{\mathrm{T}} \qquad (93)$$

where

$$\mathbf{s}_k=\mathbf{x}_{k+1}-\mathbf{x}_k=\alpha_k\mathbf{d}_k \qquad (94)$$

$$\mathbf{y}_k=\mathbf{g}_{k+1}-\mathbf{g}_k. \qquad (95)$$

Quasi-Newton methods can also be formulated by updating the *inverse* of the approximate Hessian. If $\mathbf{H}_k$ is a quasi-Newton approximation to the inverse Hessian, the search direction is defined by

$$\mathbf{d}_k=-\mathbf{H}_k\mathbf{g}_k \qquad (96)$$

and the quasi-Newton condition is

$$\mathbf{H}_{k+1}\mathbf{y}_k=\mathbf{s}_k. \qquad (97)$$

A quasi-Newton algorithm is then

$$\mathbf{x}_{k+1}=\mathbf{x}_k+\mathbf{p}_k \qquad (98)$$

$$\mathbf{s}_k\equiv\alpha_k\mathbf{p}_k$$

$$\mathbf{p}_k=-\alpha_k H_k\mathbf{g}_k \qquad (99)$$

and H_k is updated by the BFGS formula

$$\mathbf{H}_{k+1}=\mathbf{H}_k-\frac{\mathbf{H}_k\mathbf{y}_k\mathbf{p}_k{}^{\mathrm{T}}}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k}+\left(1+\mathbf{y}_k{}^{\mathrm{T}}\frac{\mathbf{H}_k\mathbf{y}_k}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k}\right)\frac{\mathbf{p}_k\mathbf{p}_k{}^{\mathrm{T}}}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k}. \qquad (100)$$

For a derivation of the BFGS updating methods for the Hessian and the inverse Hessian, see appendix C.

On the other hand, the conjugate-gradient algorithm can be written as

$$\mathbf{x}_{k+1}=\mathbf{x}_k+\alpha_k\mathbf{d}_k \quad \mathbf{d}_0=-\mathbf{H}\mathbf{g}_0 \qquad (101)$$

$$\mathbf{d}_{k+1}=-\mathbf{H}\mathbf{g}_{k+1}+\beta_k\mathbf{d}_k \qquad (102)$$

$$\beta_k = \mathbf{g}_{k+1}^{\mathrm{T}} \frac{\mathbf{H}\mathbf{y}_k}{\mathbf{d}_k{}^{\mathrm{T}}\mathbf{y}_k}. \qquad (103)$$

When $\mathbf{H} \equiv \mathbf{I}$ we regain the original conjugate-gradient algorithm of Fletcher and Reeves (1964).

Perry (1976, 1977, 1978) and Shanno (1978a, 1978b) propose a conjugate-gradient method with inexact line searches that can be interpreted as a memoryless quasi-Newton method with BFGS updating.

The memoryless (or limited-memory) quasi-Newton method is based upon the idea of computing the direction of search

$$\mathbf{p}_k = \alpha_k \mathbf{d}_k \qquad (104)$$

as

$$-\mathbf{H}_k \mathbf{g}_k \qquad (105)$$

where $\mathbf{H}_k$ is a matrix obtained by updating the identity matrix $\mathbf{I}$ with a limited number of quasi-Newton corrections.

Although the direction of search $\mathbf{d}_k$ is computed as the product of a matrix with a vector, the matrix $\mathbf{H}_k$ is never computed explicitly and the only additional storage is the vectors defining the updates. Different limited quasi-Newton methods can be defined, depending on the number of updating vectors stored (see Nazareth, 1979).

Shanno (1978) developed his method by using an observation of Perry (1976) that if

$$\beta_k = \frac{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{g}_{k+1}}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{d}_k} \quad \mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \qquad (106)$$

then

$$\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k = -\left(I - \frac{\mathbf{d}_k \mathbf{y}_k{}^{\mathrm{T}}}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{d}_k}\right)\mathbf{g}_{k+1}. \qquad (107)$$

If one denotes

$$\mathbf{p}_k = \alpha_k \mathbf{d}_k = \mathbf{x}_{k+1} - \mathbf{x}_k \qquad (108)$$

then one can obtain (see Shanno, 1978; Gill and Murray, 1979) the new direction with the "one-step" limited-memory BFGS quasi-Newton update using (95) with $\mathbf{H}_k$ equal to the identity matrix as

$$\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} - \left[\left(1 + \frac{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{y}_k}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k}\right)\frac{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{g}_{k+1}}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k} - \frac{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{g}_{k+1}}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k}\right]\mathbf{p}_k + \frac{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{g}_{k+1}}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k}\mathbf{y}_k. \qquad (109)$$

If this formula is applied with an exact line search, then $\mathbf{p}_{k+1} = \alpha_{k+1}\mathbf{d}_{k+1}$ is identical to the formula obtained from the conjugate-gradient method as all vectors multiplied by

$$\mathbf{s}_k{}^{\mathrm{T}}\mathbf{g}_{k+1} = \alpha_k \mathbf{d}_k{}^{\mathrm{T}}\mathbf{g}_{k+1} = 0 \qquad (110)$$

vanish. This means that the limited-memory quasi-Newton method of Shanno generates mutually conjugate directions if an exact linear search is made. However, the method of Shanno (1978) performs with inexact line searches and is relatively insensitive to line search errors, yielding descent directions under mild restrictions on line-search accuracy. Shanno (1978) included Beale restarts in the limited-memory quasi-Newton scheme (i.e., the Powell restart criteria) as well as a scaling of the conjugate-gradient suggested by Fletcher (1972):

$$\mathbf{d}_{k+1_{\text{scaled}}} = (2(f_{k+1} - f_k)/\mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{d}_{k+1})\mathbf{d}_{k+1}. \qquad (111)$$

The basic linear search used Davidon's (1959) cubic interpolation to find a step length α that satisfies

$$f(\mathbf{x} + \alpha\mathbf{d}) < f(\mathbf{x}) + \alpha\mathbf{d}^{\mathrm{T}}\mathbf{g}(\mathbf{x})0.0001 \qquad (112)$$

where $\mathbf{d}$ is the chosen direction, $\mathbf{g}(\mathbf{x}) = \nabla f(\mathbf{x})$, and $\mathbf{d}^{\mathrm{T}}\mathbf{g}(\mathbf{x})$, the directional derivative of $f(\mathbf{x})$ at $\mathbf{x}$ along $\mathbf{d}$, is negative. The value α must also satisfy

$$\mathbf{d}^{\mathrm{T}}\mathbf{g}(\mathbf{x} + \alpha\mathbf{d})/\mathbf{d}^{\mathrm{T}}\mathbf{g}(\mathbf{x}) < 0.9. \qquad (113)$$

The initial α at iteration step $k + 1$ is chosen to be

$$\alpha_{k+1} = \alpha_k \frac{\mathbf{d}_k{}^{\mathrm{T}}\mathbf{g}_k}{\mathbf{d}_{k+1}^{\mathrm{T}}\mathbf{g}_{k+1}}. \qquad (114)$$

Shanno's (1978) and Shanno and Phua's (1980) CONMIN algorithm implementing the quasi-Newton limited-memory conjugate-gradient requires seven vectors of length N for $\mathbf{x}_k$, $\mathbf{x}_{k+1}$, $\mathbf{g}_k$, $\mathbf{g}_{k+1}$, $\mathbf{d}_k$, $\mathbf{d}_t$ and $\mathbf{y}_t$ (this includes memory for Beale's restart).

4. Numerical tests: Comparison of different conjugate-gradient algorithms for two meteorological problems

a. The methods tested

The choice of the conjugate-gradient methods to be tested was dictated by the evolution of know-how in the unconstrained optimization field and by the availability of conjugate-gradient codes, i.e., software.

To test the Fletcher–Reeves and Polak–Ribiere methods, we used the NAG (1981) Fortran Library Subroutine E04DBF and slightly modified the same routine to implement the Polak–Ribiere method (hereafter denoted E04DBG).

When using E04DBF the user must supply a routine FUNCT to calculate the value of the function and the values of its first derivatives (the gradient vector) for any set of values of the variables. One iteration cycle is completed when N conjugate directions have been used. Another subroutine MONIT has to be supplied by the user to print out current values of parameters of interest and number of function calls (function and gradient normal values). Each conjugate-gradient iteration involves a linear minimization involving at least two calls of FUNCT.

For the Powell (1977) restart method we used the ZXCGR subroutine of the IMSL Fortran Scientific Library (1981). The user has to provide an initial value of the vector of variables x as well as a subroutine to calculate the objective function and its first derivatives $(\partial f/\partial x_i)$ $(i = 1, 2, \cdots, N)$. The same subroutine FUNCT was used. The criterion for termination (convergence criterion) was that

$$\|\mathbf{g}\|^2 < \epsilon \tag{115}$$

where ϵ is a prescribed accuracy requirement.

The subroutine ZXCGR implements the Powell (1977) algorithm with Beale restarts as described in section 3. To implement the Shanno (1978) Beale restarted memoryless quasi-Newton conjugate-gradient method, we used subroutine CONMIN described by Shanno and Phua (1980) and available from ACM Algorithms Distribution Service. The subroutine CONMIN allows the user to employ two nonlinear unconstrained minimization techniques, namely, the Shanno (1978) memoryless quasi-Newton conjugate-gradient algorithm or a quasi-Newton method with a BFGS update. The Shanno (1978) conjugate-gradient algorithm requires $7n$ double-precision words of working storage, while the quasi-Newton BFGS algorithm requires $n^2/2 + 11n/2$ double-precision words of working storage.

We only implemented the conjugate-gradient option, due to storage limitations. In our test version, the same termination criterion was applied to all four conjugate-gradient methods for comparison purposes.

b. The numerical tests

1) ENFORCING CONSERVATION OF INTEGRAL INVARIANTS

Our first application was toward unconstrained minimization of an augmented-Lagrangian applied to solving the problem of enforcing simultaneous conservation of integral invariants of the shallow-water equations on a limited-area domain (see Navon and de Villiers, 1983). In this case the vector **x** had 540 variables.

Using an accuracy criteria of

$$\|\mathbf{g}(\mathbf{x})\|^2 \leqslant \epsilon_a \tag{116}$$

the four aforementioned algorithms were tested with increased accuracy requirements for computational efficiency measured in number of iterations, number of functions calls and total CPU time. These runs were done in single precision on a CYBER 760.

We minimized the following augmented-Lagrangian:

$$L(\mathbf{x}, \lambda, \rho) = f(\mathbf{x}) + \lambda^{\mathrm{T}}\mathbf{e}(\mathbf{x}) + \frac{1}{2}\rho|\mathbf{e}(\mathbf{x})|^2 \tag{117}$$

where the objective function $f(\mathbf{x})$ had the form

$$f = \sum_{j=1}^{N_x}\sum_{k=1}^{N_y}[\alpha(u-u^0)^2 + \alpha(v-v^0)^2 + \beta(h-h^0)^2]_{jk} \tag{118}$$

where λ is the Lagrange multiplier vector, $\mathbf{e}(\mathbf{x})$ are equality constraints, ρ is a penalty parameter, $(u^0, v^0, h^0)_{jk}$ are the predicted variables of velocity and height, and $(u, v, h)_{jk}$ are the values of the same variables adjusted by augmented-Lagrangian nonlinear constrained optimization. The α and β are weights determined following Sasaki (1976).

The vector **x** is a $3N_xN_y$ component vector

$$\mathbf{x} = (u_{11}, \cdots, u_{N_xN_y}, v_{11}, \cdots, v_{N_xN_y}, h_{11}, \cdots, h_{N_xN_y})^{\mathrm{T}}. \tag{119}$$

The four conjugate-gradient methods, namely, Fletcher–Reeves, Polak–Ribiere, Powell restart and Shanno–Phua (1980) quasi-Newton memoryless method, were compared for the accuracy criteria

$$\|\mathbf{g}(\mathbf{x})\|^2 \leqslant \epsilon_a. \tag{120}$$

The results for the different methods in terms of conjugate-gradient iterations, number of function calls and CPU time are displayed in Table 1.

2) OBJECTIVE ANALYSIS OF PSEUDOSTRESS

The second problem solved concerns the objective analysis of wind-stress data over oceans. A variational method was utilized in an objective analysis scheme to provide monthly maps of pseudostress (pseudostress being defined as the magnitude of the wind times its component) winds on a $1^0 \times 1^0$ grid mesh from marine surface observations scattered throughout the Indian Ocean Basin (Fig. 1).

The function F to be minimized is expressed as a weighted sum of lack of fits to data and constraints

$$\begin{aligned} f = {} & \frac{1}{L^2}\rho\sum_x\sum_y[(u-u_0)^2 + (v-v_0)^2] \\ & + \frac{1}{L^2}\gamma\sum_x\sum_y[(u-u_c)^2 + (v-v_c)^2] \\ & + L^2\Gamma\sum_x\sum_y[[\nabla^2(u-u_c)]^2 + [\nabla^2(v-v_c)]^2] \\ & \quad + \beta\sum_x\sum_y[\nabla\cdot(\mathbf{v}-\mathbf{v}_c)]^2 + \alpha\sum_x\sum_y[\mathbf{k}\cdot\nabla\times(\mathbf{v}-\mathbf{v}_c)]^2 \end{aligned} \tag{121}$$

where u, v are the eastward and northward components of the resulting pseudostress wind **v** values, respectively; u_0, v_0 are the components of the initial values; and u_c, v_c are the components of an appropriate climatology. The coefficients ρ, γ, Γ, β and α are weights that control how closely the minimizing analysis fits each type of data or constraint, and L is a convenient length scale that allows all the bracketed expressions to be of the same order of magnitude.

The double sums are over 3133 points in the x–y mesh. Since there are two components of the pseudostress at each grid location (i.e., u and v are variables), the total number of variables in F is twice the number of grid points in the analysis domain, i.e., F is a function of 6266 variables. The full description of the method

TABLE 1. Enforcing conservation of integral invariances of the shallow water equation.

Method	Number of iterations	Number of function calls	Function value at minimum	CPU time (sec)
		Accuracy $\epsilon_a = 10^{-2}$		
E04DBF (Fletcher-Reeves)	12	25	0.4842×10^{-5}	3.3
E04DBG (Polak-Ribiere)	12	25	0.4842×10^{-5}	3.4
ZXCGR (Powell)	5 iterations 3 Beale restarts	21	0.4842×10^{-5}	2.7
CONMIN (Shanno-Phua)	5 iterations 1 Beale restart	15	0.4842×10^{-5}	2.7
		Accuracy $\epsilon_a = 10^{-4}$		
E04DBF (Fletcher-Reeves)	41	81	0.4842×10^{-5}	8.3
E04DBG (Polak-Ribiere)	51	102	0.4842×10^{-5}	9.7
ZXCGR (Powell)	12 iterations 8 Beale restarts	31	0.48419×10^{-5}	3.9
CONMIN (Shanno-Phua)	10 iterations 1 Beale restart	21	0.48417×10^{-5}	3.2
		Accuracy $\epsilon_a = 10^{-6}$		
E04DBF (Fletcher-Reeves)	Failed	Failed	—	—
E04DBG (Polak-Ribiere)	Failed	Failed	—	—
ZXCGR (Powell)	17 iterations 13 Beale restarts	42	0.48419×10^{-5}	5.2
CONMIN (Shanno-Phua)	20 iterations 10 Beale restarts	44	0.48415×10^{-5}	5.4
		Accuracy $\epsilon_a = 10^{-8}$		
ZXCGR (Powell)	Failed	—	—	—
CONMIN (Shanno-Phua)	26 iterations 12 Beale restarts		0.484136×10^{-5}	7.1

and its results will be given in Legler et al. [1987 (in preparation)].

Due to the large computational resources involved the second problem was tested for only two cases of accuracy requirements.

The behavior of the objective function scaled by its initial value $f/|f_0|$ as well as that of the gradient given by $\log_{10}|g/g_0|$ as a function of the conjugate-gradient iterations is displayed in Figs. 2–9 for the Shanno–Phua (1980) method and for accuracies $\epsilon_a = 10^{-2}$, 10^{-4}, 10^{-6} and 10^{-8} accordingly for test problem 1. Of note is the increase in accuracy of the gradient as the accuracy requirement ϵ_a increases from 10^{-2} to 10^{-8}.

For the second test problem we only display the behavior of the objective function scaled by its initial value $f/|f_0|$ as well as that of the gradient scaled by its initial value as a function of the conjugate-gradient iterations for an accuracy of $\epsilon_a = 10^{-2}$ for the Fletcher-Reeves, Polak–Ribiere, Powell restart and Shanno–Phua methods in Figs. 10–17.

c. Discussion of numerical results

Two unconstrained minimization cases typical of meteorological problems of interest were tested. Since the conjugate-gradient method requires "downhill" search directions, it is clear that we are finding a local minimum of the functions being minimized. Whether this minimum is a global minimum will depend on the nature of the function being minimized (convex or not) and on the initial estimate of (**x**). All of the methods were considered to have worked if they satisfied the stopping criterion (120) or to have failed if they did not satisfy the criteria (120) after 100 conjugate-gradient iterations.

The first test problem minimizing the augmented-Lagrangian (117) was convex and quasi-quadratic, and for a moderate accuracy ($\epsilon = 10^{-2}$) all the methods performed well, with ZXCGR and CONMIN being more efficient. The lead of ZXCGR and CONMIN become more evident for $\epsilon_a = 10^{-4}$ and fully prominent

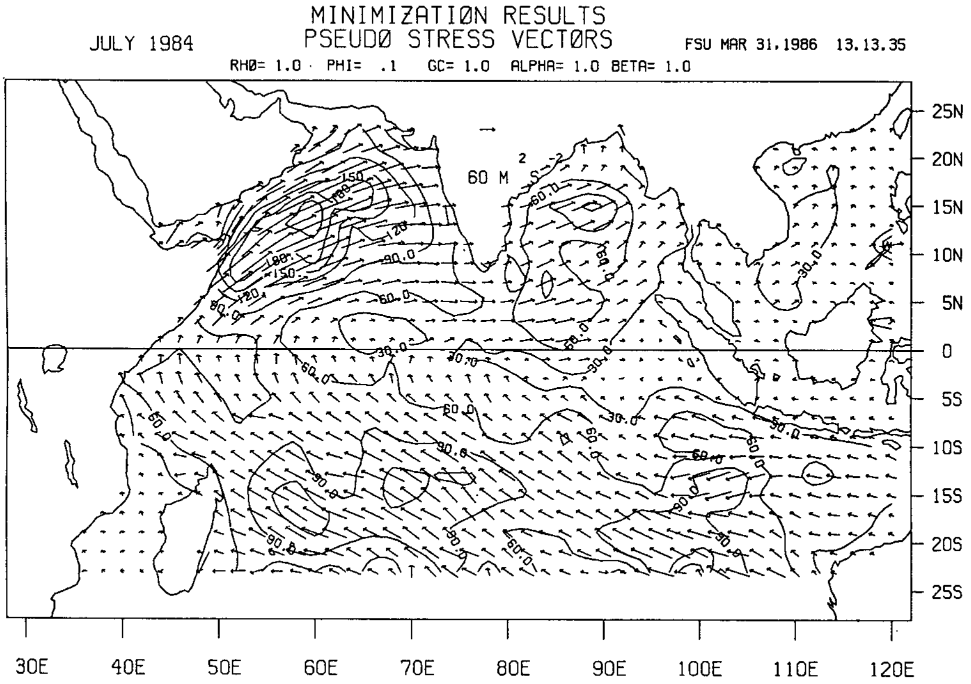

FIG. 1. Results of minimizing analyses for the pseudostress vectors shown on a 2° × 2° grid mesh from marine surface observations in the Indian Ocean Basin.

for a high accuracy of $\epsilon = 10^{-6}$ when both E04DBF and E04DBG failed. For a very high accuracy requirement of $\epsilon_a = 10^{-8}$ only CONMIN was able to perform.

For the large-scale problems with $n = 6266$ variables and for an accuracy of $\epsilon_a = 10^{-2}$ we note first a high increase in the computer time (CPU) required to achieve the minimization (see Table 2). This is due both to the increased number of variables and to the more complicated form of the gradient. It is common practice to weigh a gradient evaluation as equivalent to n function evaluations, and this is particularly representative when the gradients are obtained by finite differences (see Le, 1985).

The basic formula for CPU time consumption in an optimization is

$$T \approx t_f n_f + t_g n_g + t_i n_i = t_f(n_f + n n_g) + t_i n_i \qquad (122)$$

where t_f and t_g are the time required per function and

TABLE 2. Objective analysis of wind stress over oceans.

Method	Number of iterations	Number of function calls	Function value at minimum	CPU time (sec)
		Accuracy $\epsilon_a = 10^{-2}$		
E04DBF (Fletcher–Reeves)	Failed	—	—	—
E04DBG (Polak–Ribiere)	10 iterations	19	$0.51766726 \times 10^{-8}$	529
ZXCGR (Powell)	4 iterations 4 Beale restarts	9	$0.52277593 \times 10^{-8}$	254
CONMIN (Shanno–Phua)	4 iterations 3 Beale restarts	9	$0.51213279 \times 10^{-8}$	257
		Accuracy $\epsilon_a = 10^{-3}$		
E04DBG (Polak–Ribiere)	14 iterations	27	$0.48585018 \times 10^{-8}$	745
ZXCGR (Powell)	Failed	—	—	—
CONMIN (Shanno–Phua)	9 iterations 8 Beale restarts	22	$0.48774742 \times 10^{-8}$	618

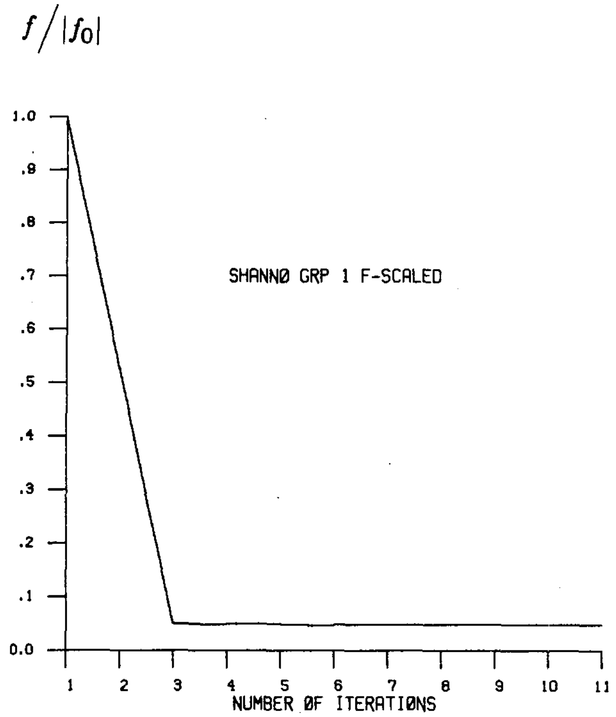

FIG. 2. Variation of the value of the scaled objective function $f/|f_0|$ with the number of conjugate-gradient iterations for the Shanno–Phua method with accuracy criteria $\|g(x)\|^2 < 10^{-2}$ for problem 1.

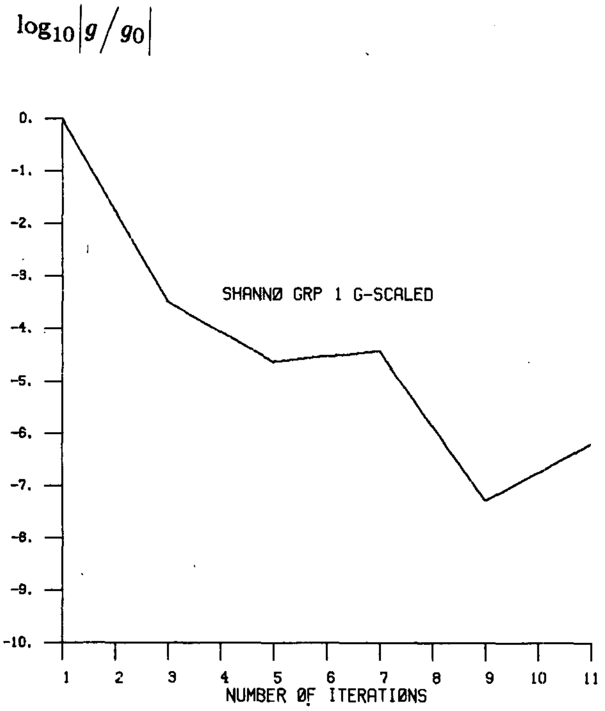

FIG. 3. Variation of the value of the scaled gradient ($\log_{10}|g/g_0|$) with the number of conjugate-gradient iterations. The Shanno–Phua method with accuracy criteria $\|g(x)\|^2 < 10^{-2}$ for problem 1.

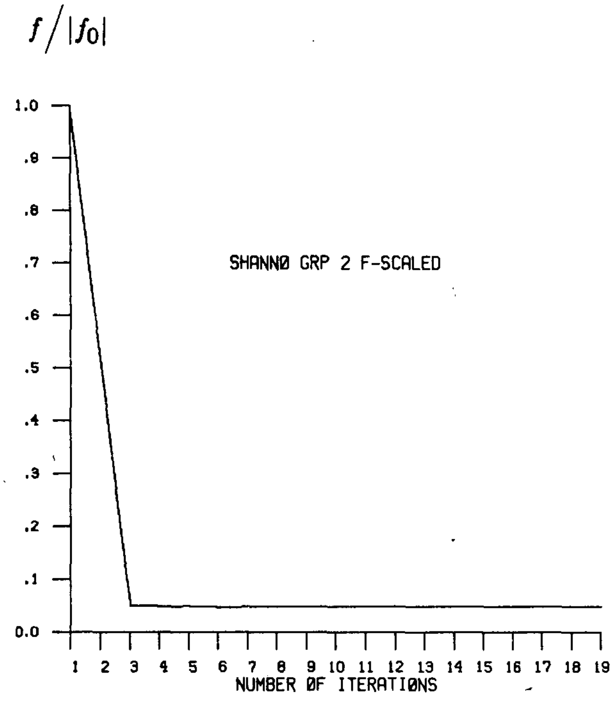

FIG. 4. As in Fig. 2 but for accuracy $\epsilon_a = 10^{-4}$.

gradient call, respectively, and t_i is the average overhead execution time per iteration. Here n_f is the number of function evaluations, n_g the number of gradient evaluations and n_i the number of iterations. For $\epsilon_a = 10^{-2}$

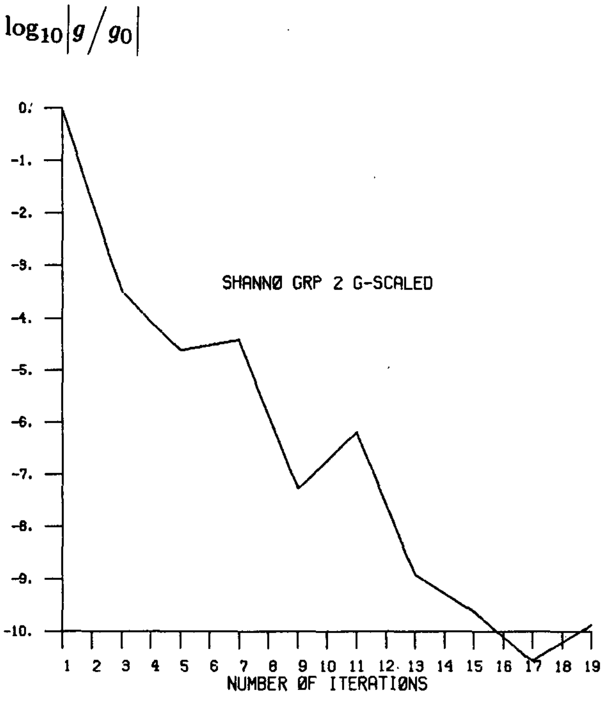

FIG. 5. As in Fig. 3 but for accuracy $\epsilon_a = 10^{-4}$.

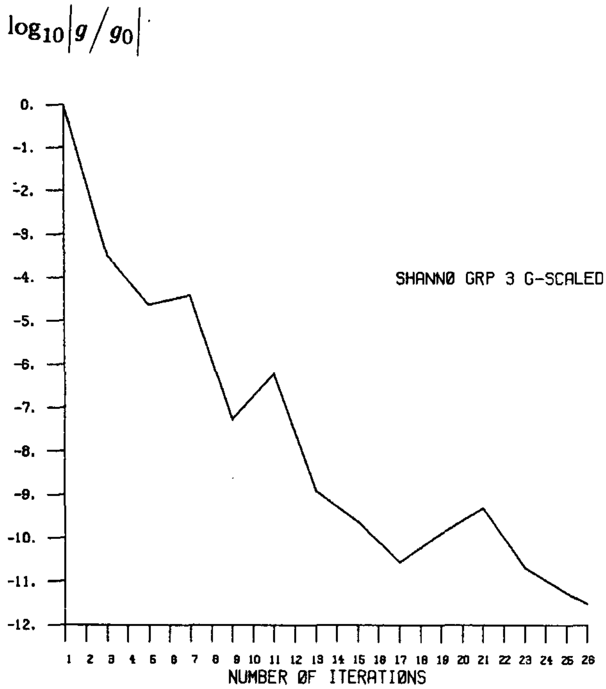

FIG. 6. As in Fig. 2 but for accuracy $\epsilon_a = 10^{-6}$.

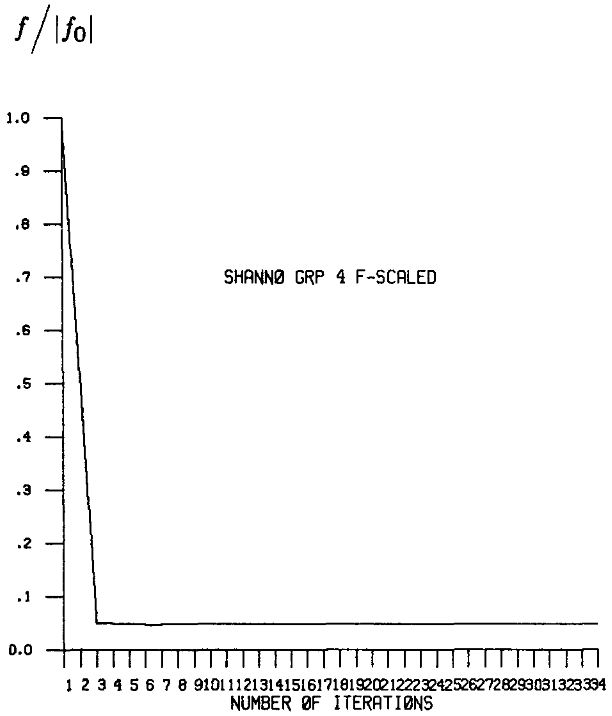

FIG. 8. As in Fig. 2 but for accuracy $\epsilon_a = 10^{-8}$.

the usual NAG library conjugate-gradient Fletcher–Reeves algorithm failed, but a modified version designed to implement the Polak–Ribiere version of section 3 (E04DBG) was quite successful, if not very efficient. ZXCGR and CONMIN performed equally well but CONMIN obtained a lower minimum value of the function. For $\epsilon_a = 10^{-3}$ the version ZXCGR of IMSL failed but E04DBG, the Polak–Ribiere modification

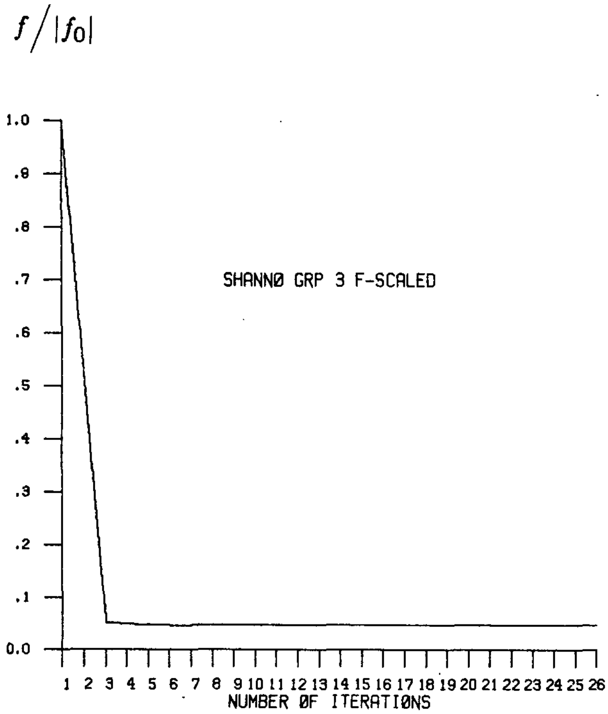

FIG. 7. As in Fig. 3 but for accuracy $\epsilon_a = 10^{-6}$.

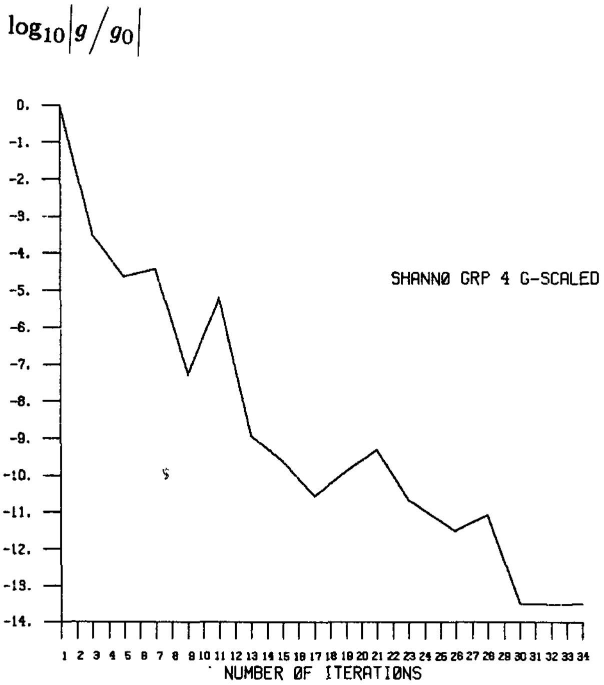

FIG. 9. As in Fig. 3 but for accuracy $\epsilon_a = 10^{-8}$.

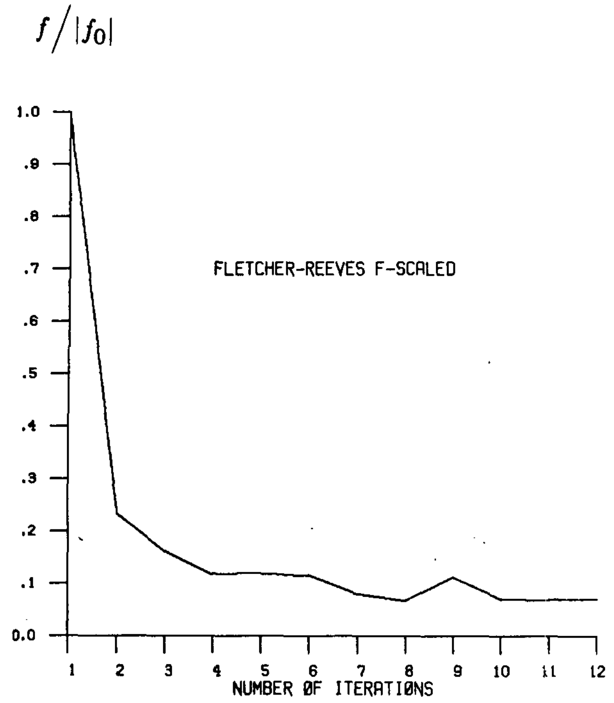

FIG. 10. The variation of the value of the scaled objective function $f/|f_0|$ with the number of conjugate-gradient iterations for problem 2, using the Fletcher-Reeves method with $\epsilon_a = 10^{-2}$.

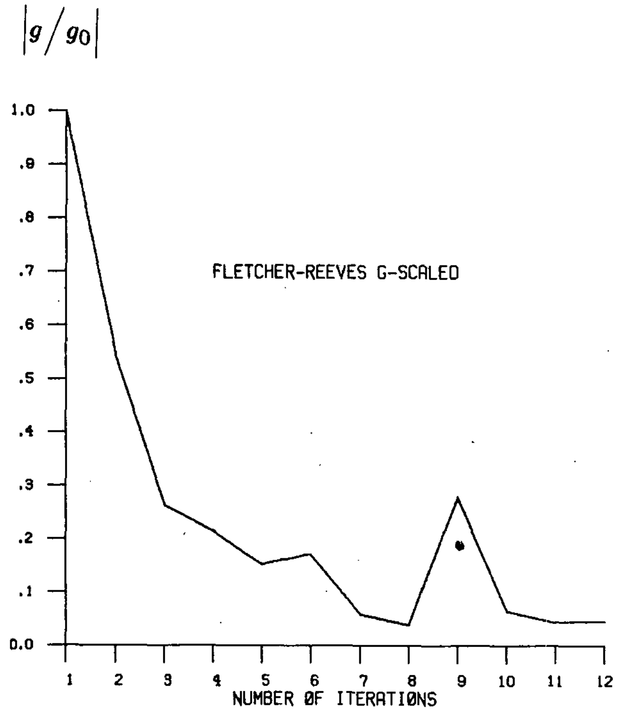

FIG. 11. Variation of the value of the scaled gradient $|g/g_0|$ with the number of conjugate-gradient iterations for problem 2 using the Fletcher-Reeves method with $\epsilon_a = 10^{-2}$.

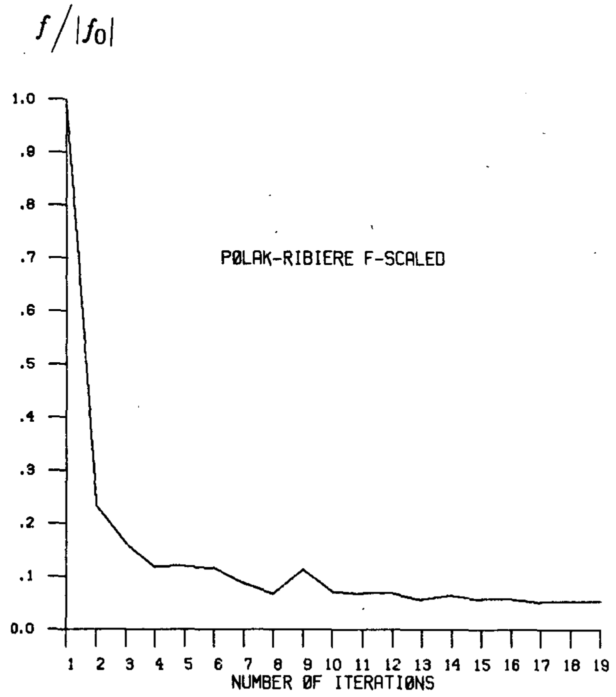

FIG. 12. As in Fig. 10 but for the Polak-Ribiere method.

of E04DBF of NAG (1981), performed very well. The CONMIN routine of Shanno and Phua (1980) was the more efficient of the two. The CONMIN algorithm appeared to require too many Beale restarts near

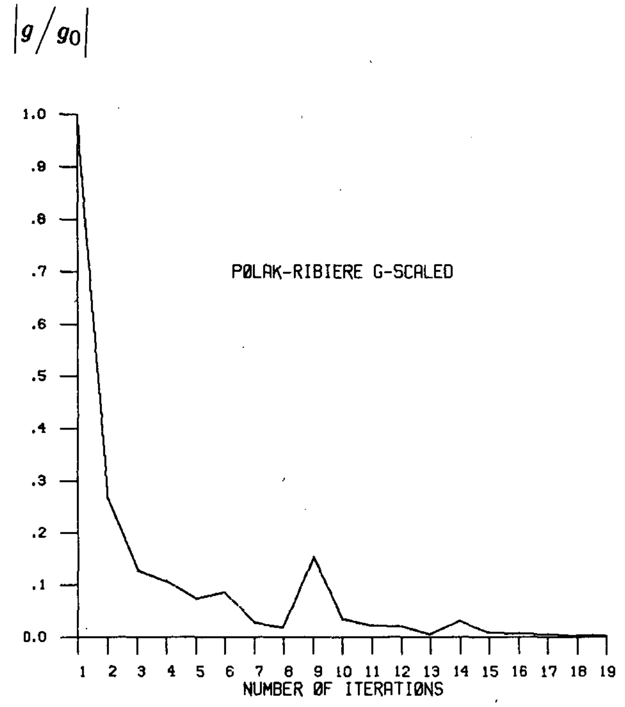

FIG. 13. As in Fig. 11 but for the Polak-Ribiere method.

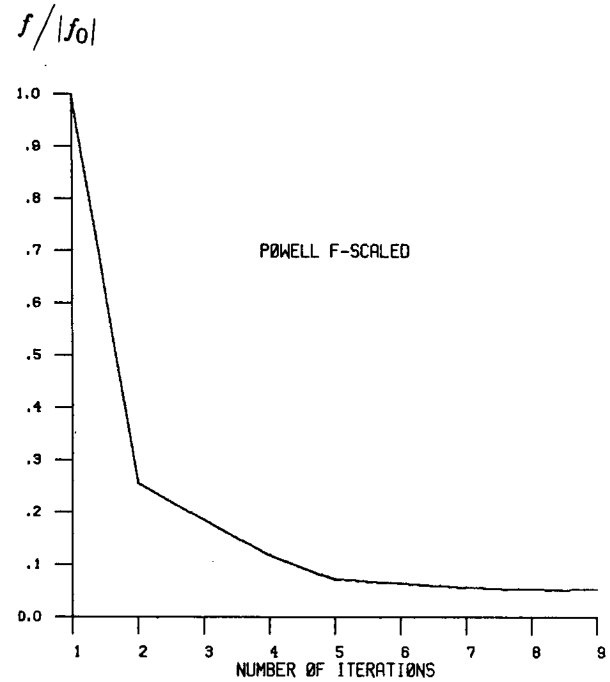

FIG. 14. As in Fig. 10 but for the Powell restart method.

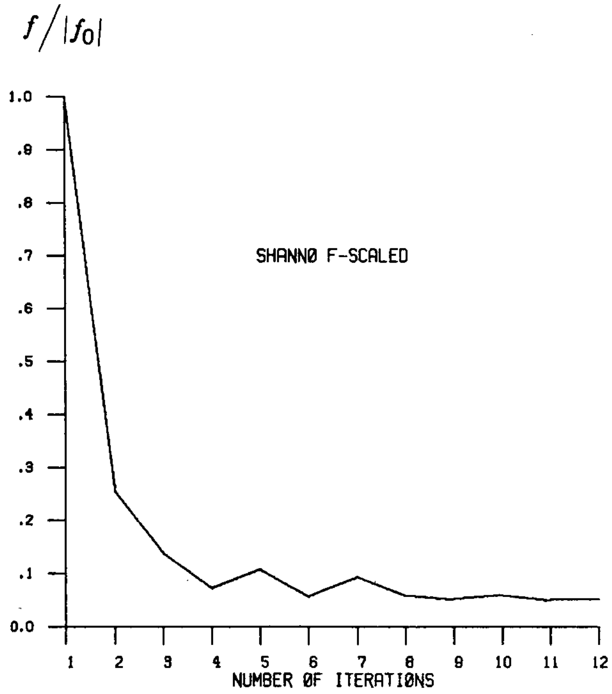

FIG. 16. As in Fig. 10 but for the Shanno–Phua method.

convergence. The good performance of the conjugate-gradient method, in particular, for large-scale minimization of $\sim 10^4$ variables, is very encouraging for meteorological users. The success is partly due to the quasi-quadratic quasi-convex nature of the problems arising in meteorological applications. The robustness of CONMIN for the cases tested in this paper proves that the method is successful for a given class of prob-

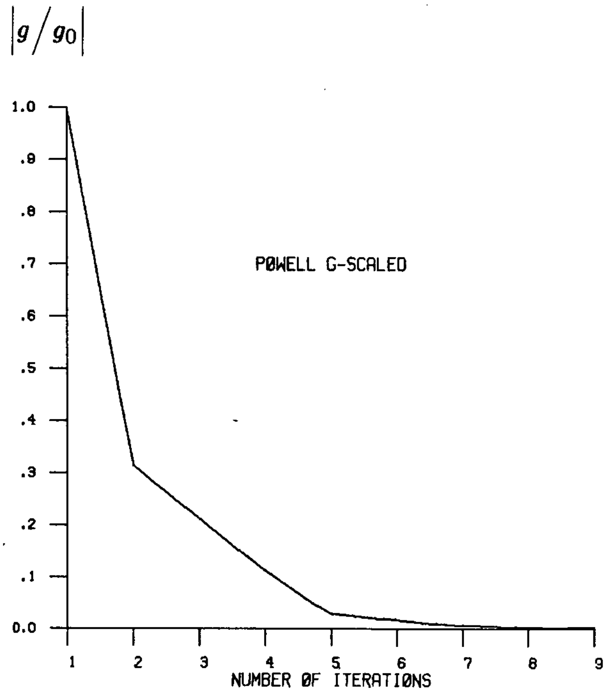

FIG. 15. As in Fig. 11 but for the Powell restart method.

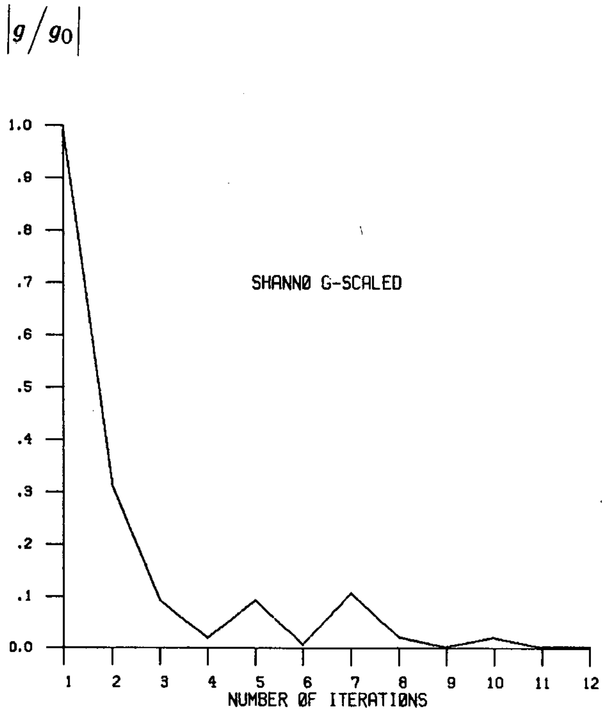

FIG. 17. As in Fig. 11 but for the Shanno–Phua method.

lems. However, one should qualify this statement, as it is well known in optimization theory that for different classes of problems, different conjugate-gradient algorithms prove to be optimal. As such, users should experiment with several conjugate-gradient codes mentioned in this paper to find the most suitable one for their problem. As we shall see in the next section, recent developments in unconstrained minimization cover the full range of possible functions.

5. New developments for large-scale minimization

a. The truncated Newton method

Dembo and Steihaug (1983) proposed a method called the truncated Newton method. Starting from the fact that the Newton method is based on approximating the function $f(\mathbf{x}_k + \mathbf{d}_k)$ by the quadratic model

$$\phi(\mathbf{d}_k) = f(\mathbf{x}_k) + \mathbf{d}_k{}^{\mathrm{T}}\mathbf{g}(\mathbf{x}_k) + \frac{1}{2}\mathbf{d}_k{}^{\mathrm{T}}\mathbf{H}_k\mathbf{d}_k \qquad (123)$$

and that the Newton direction is obtained from an exact solution to $\min_d \phi(\mathbf{d}_k)$, they use a Hestenes–Steiffel conjugate-gradient method to minimize the function

$$\phi(\mathbf{d}_k) = \frac{1}{2}\mathbf{d}_k{}^{\mathrm{T}}\mathbf{H}_k\mathbf{d}_k + \mathbf{d}_k{}^{\mathrm{T}}\mathbf{g}_k \qquad (124)$$

where $\mathbf{H}_k$ is the Hessian matrix of $f(\mathbf{x})$, i.e., the symmetric matrix of second-order derivatives.

Letting $r = \nabla\phi$, the gradient of ϕ, a sequence of approximations to the search direction $\mathbf{d}_k$ is generated by

$$\mathbf{d}_k{}^0 = -\mathbf{g}_k \qquad (125)$$

$$\mathbf{s}_0 = -\mathbf{r}_0 \qquad (126)$$

$$\mathbf{d}_k{}^{i+1} = \mathbf{d}_k{}^i + \alpha_i\mathbf{s}_i \qquad (127)$$

$$\mathbf{s}_{i+1} = -\mathbf{r}_{i+1} + \beta_i\mathbf{s}_i \qquad (128)$$

$$\alpha_i = -\mathbf{s}_i{}^{\mathrm{T}}\mathbf{r}_i/\mathbf{s}_i{}^{\mathrm{T}}\mathbf{H}_k\mathbf{s}_i \qquad (129)$$

$$\beta_i = \mathbf{r}_{i+1}^{\mathrm{T}}\mathbf{r}_{i+1}/\mathbf{r}_i{}^{\mathrm{T}}\mathbf{r}_i. \qquad (130)$$

Using this inaccurate solution to the Newton equations by the conjugate-gradient method, the sequence (125)–(130) will in no more than n steps solve the linear system. If the Hessian matrix $\mathbf{H}_k$ has directions of negative curvature, at some point the term $\mathbf{s}_i{}'\mathbf{H}_k\mathbf{s}_i$ in the denominator of (129) will become negative. At such a stage Dembo and Steihaug (1983) terminate the truncated-Newton conjugate-gradient iteration and use the approximation $\mathbf{d}_k{}^{(i)}$ computed up to this point. Additional sets of rules for terminating the sequence of approximations $\mathbf{d}_k{}^{(i)}$ before a full solution is obtained are recommended by Dembo and Steihaug, yielding substantial savings in computer time without reducing greatly the rate of convergence of the Newton method. For instance, they use the truncated-Newton termination

$$\text{if} \quad \|\mathbf{r}_{i+1}\|/\|\mathbf{g}\| \leqslant \eta \qquad (131)$$

$$\text{then} \quad \mathbf{d} = \mathbf{d}_k{}^{i+1} \qquad (132)$$

where the sequence $\{\eta_k\}$ is called a forcing sequence and condition (131) is referred to as a truncated-Newton termination. Usually $\{\eta_k\}$ is chosen as $\eta_k = \min\{1/k, \|\mathbf{g}(x_k)\|^t\}$ for some $0 \leqslant t \leqslant 1$, and Dembo and Steihaug show how to construct a truncated-Newton algorithm possessing any prescribed order of convergence between 1 and 2. One way to avoid storage of the Hessian matrix is the observation that the matrix–vector product $\mathbf{H}_k\mathbf{s}_i$, where $\mathbf{s}_i$ is a direction generated within the conjugate-gradient formula method, can be approximated using the finite difference formula

$$\mathbf{H}_k\mathbf{s}_i \sim \frac{1}{\sigma}[\nabla f(\mathbf{x}_k + \sigma\mathbf{s}_i) - \nabla f(\mathbf{x}_k)]. \qquad (133)$$

This eliminates the need for storing any matrix but requires an additional gradient evaluation. Then the method becomes equivalent to a conjugate-gradient in storage requirements. This method holds promise because of its Newton step, which near the optimum is a good predictor of both step size and step direction, whereas conjugate-gradient algorithms are only good predictors of step directions. As such they require more computations to determine a good step length α_k. (See appendix A.) A good choice for the finite-difference step size σ was chosen by Dembo and Steihaug (1983) to be

$$\sigma = \frac{\sqrt{\text{machine precision}}}{\|\mathbf{d}\|} = 10^{-8}/\|\mathbf{d}\| \qquad (134)$$

for their case.

b. The Buckley-Lenir (1983) quasi-Newton variable storage conjugate-gradient method

Using the Shanno (1978) BFGS update of the identity matrix applied to $\mathbf{g}_{k+1}$, Eq. (109) can be written as

$$\mathbf{d}_{k+1} = -\left[1 + \frac{\mathbf{p}_k y_k{}^{\mathrm{T}} + \mathbf{y}_k\mathbf{p}_k{}^{\mathrm{T}}}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k} + \left(1 + \frac{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{y}_k}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k}\right)\frac{\mathbf{p}_k\mathbf{p}_k{}^{\mathrm{T}}}{\mathbf{p}_k{}^{\mathrm{T}}\mathbf{y}_k}\right]\mathbf{g}_{k+1}. \qquad (135)$$

A preconditioned Fletcher–Reeves (1964) conjugate-gradient algorithm has the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k\mathbf{d}_k \qquad (136)$$

$$\mathbf{d}_{k+1} = \mathbf{A}\mathbf{g}_{k+1} - \beta_k\mathbf{d}_k \qquad (137)$$

$$\beta_k = \mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{A}\mathbf{y}_k/\mathbf{d}_k{}^{\mathrm{T}}\mathbf{y}_k \qquad (138)$$

and

$$\mathbf{d}_0 = \mathbf{A}\mathbf{g}_0,$$

$\mathbf{A}$ being the preconditioner matrix.

Buckley and Lenir (1982, 1983, 1985) generalize the

Shanno (1978) method for variable storage of $2j$ storage vectors of length n, where a j-step approximation to the BFGS update may be kept in storage. They update the matrix H_k until $2j$ storage locations are filled and then H_k is used as a preconditioning matrix in the conjugate-gradient sequence (136)–(138). At each subsequent step after the j initial steps the new search vector is a preconditioned conjugate-gradient vector version of Shanno's limited-memory form, i.e.,

$$\mathbf{d}_{k+1}=\left[\mathbf{H}_j-\frac{\mathbf{p}_k\mathbf{y}_k^{\mathrm{T}}\mathbf{H}_j+H_j\mathbf{y}_k\mathbf{p}_k^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{y}_k}+\left(1+\frac{\mathbf{y}_k^{\mathrm{T}}H_j\mathbf{y}_k}{\mathbf{p}_k^{\mathrm{T}}\mathbf{y}_k}\right)\frac{\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{y}_k}\right]\mathbf{g}_{k+1}. \quad (139)$$

The conjugate-gradient cycle continues until either n steps of the conjugate-gradient have been done or until a Powell (1977) restart requires a restart of the conjugate-gradient procedure. At this point a new approximation of H starts. When the number j of storage vectors available is just sufficient for one update, the method collapses into the Shanno (1978) algorithm. (See also Shanno, 1983.)

The implementation of the method is described in Buckley and Lenir (1985) and a code BBVSCG is fully documented. This routine adapts to the amount of user-available storage and will do automatic computation of finite-difference approximations for derivatives if requested.

Other early attempts at combining quasi-Newton and conjugate-gradient algorithms with variable storage have been done by Buckley (1978), Nazareth (1979), Nocedal (1980) and Nazareth and Nocedal (1982).

Recent work on conjugate-gradient methods involves work of Shanno (1985) on implementing an angle test to determine when to restart conjugate-gradient methods in a steepest descent direction. The test is based on guaranteeing that the cosine of the angle between the search direction and the negative gradient is within a constant multiple of the angle between the Fletcher–Reeves search direction and the negative gradient. Recent work by Le (1985) using an algorithm employing conjugate directions and requiring only three n-dimensional vectors seems to hold promise.

c. The bundle algorithm

This algorithm carries out unconstrained minimization in the case where the functions involved have discontinuous first derivatives. This causes classical methods to fail (see Zowe, 1985). Two new concepts, the subgradient concept and the bundle concept, are then applied, allowing a minimum to be found even then. The method has been initiated by Wolfe (1975) and developed by Lemarechal (1980) and Lemarechal et al. (1981). This method has been used by Le-Dimet and Talagrand (1986), and a code M1FC1 is available for interested users (see Schittowski, 1985).

d. Parallel computing of conjugate-gradient methods

In a recent survey on parallel computing in optimization, Schnabel (1985) mentions that most of the work done was for solving linear systems by conjugate-gradient methods (see Adams, 1983; Kowalik and Kumar, 1982). Some experiments with parallel nonlinear conjugate-gradient algorithms on the ICL DAP processor array were conducted by a group at the Hatfield Polytechnic led by L. Dixon (see Dixon et al., 1982). Housos and Wing (1980) and Van Laarhoven (1985) have also proposed parallel versions of unconstrained optimization algorithms, including the nonlinear conjugate-gradient algorithm.

6. Summary and concluding remarks

This paper has summarized the current status of knowledge and experience with conjugate-gradient methods applied to large-scale unconstrained minimization in meteorology. With the recent advances in variational analysis summarized in Le-Dimet and Talagrand (1986) and Navon (1985, 1986), it becomes evident that it is important to develop good algorithms for large-scale unconstrained optimization—not only because it occurs in many meteorological applications on its own (Hoffman, 1982, 1984) but more so because it must be solved in the inner loop of the solution of nonlinear programming problems (constrained nonlinear optimization). Such methods will be the order of the day in solving problems of variational analysis and data assimilation of meteorological observations by using the augmented-Lagrangian or the adjoint equations techniques (Le-Dimet and Talagrand, 1986). Experience with optimization techniques has shown that the choice of an appropriate descent algorithm can be very efficient in reducing the total cost of a constrained minimization process.

From the conjugate-gradient methods presented and tested in this paper, the most consistent and performing one for most applications turned out to be the Shanno–Phua (1980) quasi-Newton memoryless conjugate-gradient algorithm. For moderate accuracy requirements the implementation of Powell's (1977) restart conjugate-gradient algorithm (ZXCGR) as well as a modified version of the NAG (1981) library E04DBF routine, implementing the Polak–Ribiere (1969) method, performed reasonably well. For future large-scale minimization applications in meteorology the new methods presented in section 5 should be thoroughly investigated.

At present there are no vectorized versions of the conjugate-gradient minimization method, although versions adapted for parallel processors are available (Housos and Wing, 1984). Such methods can result in substantial economy of CPU time spent on large-scale unconstrained minimization, in particular, for high-resolution global analysis data where more than 10 000 grid points have to be considered.

The success of the different versions of conjugate-gradient methods applied to meteorological problems is also due to the particular form of the functions to be minimized. It turns out that for all of the problems where large-scale unconstrained minimization is required in meteorological applications, the functional is either quasi-convex or convex and almost quadratic. This, of course, facilitates the task of the conjugate-gradient algorithm.

Finally, a word of caution for practitioners. When using finite-difference approximations for estimating derivatives, one must be careful not to request too high an accuracy since differences can only be evaluated to a precision roughly half that of the machine in most cases (see also Hoffman, 1984). Each unconstrained minimization algorithm has a set of problems on which it performs optimally. As scientific libraries have a certain time lag for adopting new methods, researchers are highly encouraged to experiment with new unconstrained-minimization methods obtainable directly from their authors in order to find the conjugate-gradient algorithm best suited to their problem.

Acknowledgments. The first author should like to acknowledge the encouragement of Dr. E. Kalnay, Head of the Global Modelling and Simulation Branch, NASA/Goddard Space Flight Center, where part of this research has taken shape. The help of Dr. Brian Ford, Director of NAG, who supplied us the E04DBF routine, is graciously acknowledged. The help of Professor David Shanno, Graduate School of Administration, University of California, Davis, who supplied us the subroutine CONMIN, is also acknowledged. The programming help of Mrs. Rosalie de Villiers of the National Research Institute for Mathematical Sciences, CSIR, Pretoria, South Africa, who helped program the first test case, is acknowledged.

The authors would also like to acknowledge the constructive contribution of an anonymous referee.

The first author is supported by the Florida State University Supercomputer Computations Research Institute, which is partially funded by the U.S. Department of Energy through Contract DE-FC05-85ER250000. The second author was supported in part by NSF Grant OCE-84-15986 and by NOAA Grant NA84-AA-D-00049. Support from Dr. James J. O'Brien is also acknowledged.

The expert typing of Mrs. Mimi Burbank, which rendered a difficult manuscript readable, is acknowledged.

APPENDIX A

Computation of the Step Length

The conjugate-gradient method, being a descent method, proceeds from one iteration to the next iteration using the formula

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k \tag{A1}$$

where $\mathbf{d}_k$ is the search direction generated by the conjugate-gradient method and α_k is a positive step length. The value of α_k is generally computed by an iterative *step-length procedure.*

If convergence of the minimization algorithm is to be assured, the step length must produce a "sufficient decrease" in F at each iteration (see Ortega and Rheinboldt, 1970). The "sufficient decrease" requirement can be satisfied by various conditions on α_k. The exact line search would have given

$$\min_{\alpha}(F(\mathbf{x}_k + \alpha \mathbf{d}_k) \tag{A2}$$

but in general the solution of the nonlinear function $dF/\partial\alpha = 0$ cannot be implemented in a finite number of operations. In practice, the line-search (or step-length) algorithm is terminated when certain conditions for an approximate minimum along the line are satisfied.

For instance, a sufficient decrease in $F(\mathbf{x})$ is achieved when α_k, the step-length, satisfies the Goldstein–Armijo principle (See Goldstein and Price, 1967; Ortega and Rheinboldt, 1970):

$$0 < -\mu_1 \alpha_k \mathbf{g}_k{}^{\mathrm{T}} \leqslant F(\mathbf{x}_k) - F(\mathbf{x}_k + \alpha_k \mathbf{d}_k) \leqslant -\mu_2 \alpha_k \mathbf{g}_k{}^{\mathrm{T}} \mathbf{d}_k \tag{A3}$$

where μ_1 and μ_2 are scalars satisfying

$$0 < \mu_1 \leqslant \mu_2 < 1. \tag{A4}$$

The upper and lower bounds of (A3) ensure that α_k is neither "too large" nor "too small." When the gradient of $F(\mathbf{x})$ is available at trial step lengths, a practical criterion based on interpreting α_k in terms of univariate minimization requires that the magnitude of the directional derivative at $\mathbf{x}_k + \alpha_k \mathbf{d}_k$ be sufficiently reduced from that at $\mathbf{x}_k$

$$|\mathbf{g}(\mathbf{x}_k + \alpha_k \mathbf{d}_k)^{\mathrm{T}} \mathbf{d}_k| \leqslant -\eta \mathbf{g}_k{}^{\mathrm{T}} \mathbf{d}_k \tag{A5}$$

where

$$0 \leqslant \eta < 1. \tag{A6}$$

To guarantee a suitable reduction in F, step-length algorithms include the following condition (see Gill et al., 1982)

$$F(\mathbf{x}_k) - F(\mathbf{x}_k + \alpha_k \mathbf{d}_k) \geqslant -\mu \alpha_k \mathbf{g}_k{}^{\mathrm{T}} \mathbf{d}_k$$

where $0 < \mu \leqslant 1/2$.

Other methods for finding the step length use either quadratic interpolation (Powell, 1971) or cubic interpolation (Davidon, 1959). The Powell (1971) method uses function values only, while Davidon's method uses the function and its derivatives.

APPENDIX B

The Beale Restart Method

Beale (1972) showed that it was possible to derive a simple conjugate-gradient formula given any general starting direction. Let us start with a quadratic objective function

$$f(\mathbf{x})=\mathbf{c}^{\mathrm{T}}\mathbf{x}+\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{G}\mathbf{x} \tag{B1}$$

where $\mathbf{x}$ is a column vector, $\mathbf{c}$ a column vector representing the gradient of f at the origin, and $\mathbf{G}$ is a symmetric matrix representing the Hessian of the objective function.

Recalling the definition of conjugacy [Eq. (2)], a set of search directions $\mathbf{d}_i$ are said to be mutually conjugate with respect to $\mathbf{G}$ if

$$\mathbf{d}_i{}^{\mathrm{T}}\mathbf{G}\mathbf{d}_j=0 \quad \text{whenever} \quad i\neq j. \tag{B2}$$

If we knew the Hessian matrix $\mathbf{G}$, and knew it was positive definite, we could construct a set of mutually conjugate directions $\mathbf{d}_1, \ldots, \mathbf{d}_n$ from an arbitrary set of linearly independent directions $\eta_1, \ldots, \eta_n$ by the Gram–Schmidt process by putting

$$\mathbf{d}_1=\eta_1 \tag{B3}$$

$$\mathbf{d}_i=\eta_i+\sum_{j=1}^{i-1}a_{ij}\mathbf{d}_j \quad i=2,3,\cdots,n \tag{B4}$$

where the coefficients a_{ij} are chosen so as to make

$$\mathbf{d}_i{}^{\mathrm{T}}\mathbf{G}\mathbf{d}_k=0 \quad \text{for all} \quad k=1,\cdots,i-1; \tag{B5}$$

i.e., the coefficients a_{ij} must satisfy the equations

$$\eta_i{}^{\mathrm{T}}\mathbf{G}\mathbf{d}_k+\sum_{j=1}^{i-1}a_{ij}\mathbf{d}_j{}^{\mathrm{T}}\mathbf{G}\mathbf{d}_k=0. \tag{B6}$$

Equation (B6) defines $i-1$ simultaneous equations for the $i-1$ unknowns a_{ij}, but by using the fact that $\mathbf{d}_1, \ldots \mathbf{d}_{i-1}$ are mutually conjugate, Eq. (B6) reduces to $i-1$ simple equations

$$a_{ij}=-\eta_i{}^{\mathrm{T}}\mathbf{G}\mathbf{d}_j/\mathbf{d}_j{}^{\mathrm{T}}\mathbf{G}\mathbf{d}_j. \tag{B7}$$

The denominator in (B7) cannot vanish if $\mathbf{G}$ is positive definite since the η_j are assumed to be linearly independent. Hence the $\mathbf{d}_j$ cannot vanish.

If, however, we wish to minimize the function f without calculating the Hessian matrix $\mathbf{G}$ and only calculate the gradient $\mathbf{g}_i$ at each point $\mathbf{x}_i$ reached at the end of each linear search as well as at the initial point $\mathbf{x}_0$, we can use the general relationship [obtained from (B1)]

$$\mathbf{g}=\mathbf{c}+\mathbf{G}\mathbf{x} \tag{B8}$$

so that if

$$\mathbf{x}_i=\mathbf{x}_{i-1}+\lambda_i\mathbf{d}_i \tag{B9}$$

then we get by using (B8) that

$$\mathbf{g}_i-\mathbf{g}_{i-1}=\lambda_i\mathbf{G}\mathbf{d}_i. \tag{B10}$$

If we now take an arbitrary downhill direction d_1 (i.e., subject only to $\mathbf{d}_1{}^{\mathrm{T}}\mathbf{g}_0<0$), we can now use (B4) to define $\mathbf{d}_k$ for $k>1$ with

$$\eta_k=-\mathbf{g}_{k-1}. \tag{B11}$$

We see using (B10) that the coefficients a_{ij} defined by (B7) can be computed as

$$a_{kj}=\mathbf{g}_{k-1}^{\mathrm{T}}(\mathbf{g}_j-\mathbf{g}_{j-1})/\mathbf{d}_j{}^{\mathrm{T}}(\mathbf{g}_j-\mathbf{g}_{j-1}) \tag{B12}$$

without any explicit evaluation of the Hessian.

By making use of the fact that after j linear searches along the conjugate direction, f has been optimized in a hyperplane spanned by these directions, then $\mathbf{g}_j$ must be orthogonal to this hyperplane, hence to $\mathbf{d}_1, \ldots, \mathbf{d}_j$, and since these directions were constructed from $\mathbf{g}_1, \ldots, \mathbf{g}_{j-1}$ we see that

$$\mathbf{g}_k{}^{\mathrm{T}}\mathbf{g}_j=0 \quad 0<k<j. \tag{B13}$$

Hence $a_{ij}=0$ for $j=2, 3, \cdots, i-2$.

So that (B4) reduces to

$$\mathbf{d}_2=-\mathbf{g}_1+a_{21}\mathbf{d}_1 \tag{B14}$$

$$\mathbf{d}_k=-\mathbf{g}_{k-1}+a_{k1}\mathbf{d}_1+a_{k,k-1}\mathbf{d}_{k-1} \quad (k>2). \tag{B15}$$

If we now identify

$$\mathbf{d}_1\equiv\mathbf{d}_t \tag{B16}$$

$$a_{k1}\equiv\gamma_k, \quad a_{k,k-1}\equiv\beta_k. \tag{B17}$$

We get Eq. (79)

$$\mathbf{d}_k=-\mathbf{g}_{k-1}+\beta_k\mathbf{d}_{k-1}+\gamma_k\mathbf{d}_t \tag{B18}$$

and by using (B12) we get

$$\beta_k=a_{k,k-1}=\frac{\mathbf{g}_k{}^{\mathrm{T}}(\mathbf{g}_k-\mathbf{g}_{k-1})}{(\mathbf{g}_k-\mathbf{g}_{k-1})\mathbf{d}_k{}^{\mathrm{T}}} \tag{B19}$$

$$\gamma_k=a_{k,1}=\frac{\mathbf{g}_k{}^{\mathrm{T}}(\mathbf{g}_{t+1}-\mathbf{g}_t)}{\mathbf{d}_t{}^{\mathrm{T}}(\mathbf{g}_{t+1}-\mathbf{g}_t)}. \tag{B20}$$

APPENDIX C

Quasi-Newton Methods

1. The derivation of quasi-Newton updating methods

The matrix of n^2 second partial derivatives is called the Hessian. If the partial derivatives $\partial F/\partial x_i$, $\partial F/\partial x_j$ and $\partial^2 F/\partial \mathrm{x}_i\partial x_j$ are continuous, then $\partial^2 F/\partial x_j\partial x_i$ exists and

$$\frac{\partial^2 F}{\partial x_i\partial x_j}=\frac{\partial^2 F}{\partial x_j\partial k_i}. \tag{C1}$$

The n^2 second partial derivatives are represented by a square symmetric matrix, the Hessian matrix of $F(\mathbf{x})$ denoted in optimization literature by $\nabla^2 F(\mathbf{x})$ or by $\mathbf{G}(\mathbf{x})$.

Quasi-Newton methods are based on the idea of building up curvature information as the iterations of a descent method proceed, using the observed behavior of the objective function F and of its gradient $\mathbf{g}$. The theory of quasi-Newton methods is based on the fact that an approximation to the curvature of a nonlinear function can be computed without explicitly forming the Hessian matrix. Instead, $\mathbf{G}(\mathbf{x})$, the Hessian matrix, is approximated by a symmetric positive definite matrix

$\mathbf{G}^{(k)}$, which is corrected or updated from iteration to iteration.

The initial Hessian approximation $\mathbf{G}^{(0)}$ can be any positive definite matrix, but is usually taken as the identity matrix $\mathbf{I}$ in the absence of any better estimate.

If first and second derivatives of F are available, a quadratic model of the objective function can be obtained by taking the first three terms of the Taylor series expansion about the current point

$$F(\mathbf{x}_k+\mathbf{d})=F_k+\mathbf{g}_k{}^{\mathrm{T}}\mathbf{d}+\frac{1}{2}\mathbf{d}^{\mathrm{T}}\mathbf{G}_k\mathbf{d} \qquad \text{(C2)}$$

where $\mathbf{d}$ is the step to the minimum. The minimum of the right-hand side of (C2) is achieved if $\mathbf{d}_k$ is a minimum of the quadratic function

$$\phi(\mathbf{d})=\mathbf{g}_k{}^{\mathrm{T}}\mathbf{d}+\frac{1}{2}\mathbf{d}^{\mathrm{T}}\mathbf{G}\mathbf{d} \qquad \text{(C3)}$$

and a stationary point of (C3) satisfies the linear system

$$\mathbf{G}_k\mathbf{d}_k=-\mathbf{g}_k. \qquad \text{(C4)}$$

Let $\mathbf{s}_k$ be the step taken from x_k:

$$\mathbf{s}_k=\mathbf{x}_{k+1}-\mathbf{x}_k. \qquad \text{(C5)}$$

If we expand the gradient function about $\mathbf{x}_k$ in a Taylor series around $\mathbf{s}_k$ we obtain

$$\mathbf{g}(\mathbf{x}_k+\mathbf{s}_k)=\mathbf{g}_k+\mathbf{G}_k\mathbf{s}_k+\cdots. \qquad \text{(C6)}$$

The curvature of F along $\mathbf{s}_k$ is given by $\mathbf{s}_k{}^{\mathrm{T}}\mathbf{G}_k\mathbf{s}_k$, which can be approximated by using only first-order derivative information [from (C6)]

$$\mathbf{s}_k{}^{\mathrm{T}}\mathbf{G}_k\mathbf{s}_k\approx(\mathbf{g}(\mathbf{x}_k+\mathbf{s}_k)-\mathbf{g}_k)^{\mathrm{T}}\mathbf{s}_k. \qquad \text{(C7)}$$

Let $\mathbf{B}_k$ denote the approximate Hessian at the beginning of the kth iteration of a quasi-Newton method. If $\mathbf{B}_k$ is taken as the Hessian matrix of a quadratic model function, the search direction $\mathbf{p}_k$ is the solution of a linear system:

$$\mathbf{B}_k\mathbf{p}_k=-\mathbf{g}_k \qquad \text{(C8)}$$

or

$$\mathbf{p}_k=-\mathbf{B}_k{}^{-1}\mathbf{g}_k=-\mathbf{B}_k{}^{-1}\nabla_xF(\mathbf{x}_k). \qquad \text{(C9)}$$

After $\mathbf{x}_{k+1}$ has been computed using a line search along $\mathbf{s}_k$

$$(\mathbf{s}_k=\mathbf{x}_{k+1}-\mathbf{x}_k\equiv\alpha_k\mathbf{p}_k) \quad \text{giving}$$

$$\mathbf{x}_{k+1}=\mathbf{x}_k+\alpha_{(k)}\mathbf{s}_k \qquad \text{(C10)}$$

a new Hessian approximation $\mathbf{B}_{k+1}$ is computed by updating $\mathbf{B}_k$ to take into account newly acquired curvature information.

If we denote by $\mathbf{y}_k$ the change in the gradient

$$\mathbf{y}_k=\mathbf{g}_{k+1}-\mathbf{g}_k=\Delta\mathbf{g}_k, \qquad \text{(C11)}$$

the standard condition required of the updated Hessian approximation is to approximate curvature of F along α_k, i.e., based on (C8) to satisfy the quasi-Newton condition

$$\mathbf{B}_{k+1}\mathbf{s}_k=\mathbf{y}_k. \qquad \text{(C12)}$$

During a single quasi-Newton iteration, we obtain information about the second-order behavior of F only along one direction. As such, the quasi-Newton update, $\mathbf{B}_{k+1}$, is expected to differ from $\mathbf{B}_k$ by a matrix of low rank. (The rank or dimension of a subspace is the smallest integer r such that the entire subspace can be generated by a set of r linearly independent vectors.) A matrix of the form $\mathbf{u}\mathbf{v}^{\mathrm{T}}$, where $\mathbf{u}$ and $\mathbf{v}$ are vectors, is termed a matrix of rank one. Every column of $\mathbf{u}\mathbf{v}^{\mathrm{T}}$ is a multiple of the vector $\mathbf{u}$ and every row is a multiple of the vector $\mathbf{v}^{\mathrm{T}}$. It can be shown that the quasi-Newton condition can be satisfied by updating $\mathbf{B}_k$ with a rank one matrix.

Assume

$$\mathbf{B}_{k+1}=\mathbf{B}_k+\mathbf{u}\mathbf{v}^{\mathrm{T}}. \qquad \text{(C13)}$$

From the quasi-Newton condition (C12) we get

$$\mathbf{B}_{k+1}\mathbf{s}_k=(\mathbf{B}_k+\mathbf{u}\mathbf{v}^{\mathrm{T}})\mathbf{s}_k=\mathbf{y}_k \qquad \text{(C14)}$$

or

$$\mathbf{u}(\mathbf{v}^{\mathrm{T}}\mathbf{s}_k)=\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k. \qquad \text{(C15)}$$

Therefore $\mathbf{u}$ is in the direction $\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k$.

For any $\mathbf{v}$ such that $\mathbf{v}^{\mathrm{T}}\mathbf{s}_k\neq0$, the vector $\mathbf{u}$ is given by

$$\frac{1}{\mathbf{v}^{\mathrm{T}}\mathbf{s}_k}(\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k) \qquad \text{(C16)}$$

and we get

$$\mathbf{B}_{k+1}=\mathbf{B}_k+\frac{1}{\mathbf{v}^{\mathrm{T}}\mathbf{s}_k}(\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k)\mathbf{v}^{\mathrm{T}}. \qquad \text{(C17)}$$

Given any vector $\mathbf{w}$ orthogonal to $\mathbf{s}_k$, the rank one matrix $\mathbf{z}\mathbf{w}^{\mathrm{T}}$ annihilates $\mathbf{s}_k$, and the quasi-Newton condition (C12) holds if further rank-one matrices of the form $\mathbf{z}\mathbf{w}^{\mathrm{T}}$ are added to $\mathbf{B}_{k+1}$.

The quasi-Newton methods have evolved by requiring updates to possess properties of *hereditary symmetry* and *positive definiteness,* i.e., since the Hessian matrix is symmetric and positive definite, one looks for updates such that if $\mathbf{B}_k$ is symmetric and positive definite so is the next approximation to the Hessian $\mathbf{B}_{k+1}$.

For rank-one updates, requirement of symmetry inheritance uniquely determines the update. In such a case, the vector $\mathbf{v}$ must be a multiple of $\mathbf{u}$ in Eq. (C17), and the rank-one update becomes

$$\mathbf{B}_{k+1}=\mathbf{B}_k+\frac{1}{(\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k)^{\mathrm{T}}\mathbf{s}_k}(\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k)(\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k)^{\mathrm{T}}. \qquad \text{(C18)}$$

It was found that a more flexible formulation is obtained by allowing the updates to be of rank two. Such an update with the property of hereditary symmetry resulted in (see Gill et al., 1981) the update:

$$\mathbf{B}_{k+1}=\mathbf{B}_k+\frac{1}{\mathbf{v}^{\mathrm{T}}\mathbf{s}_k}[(\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k)\mathbf{v}^{\mathrm{T}}(\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k)^{\mathrm{T}}]$$
$$-\frac{(\mathbf{y}_k-\mathbf{B}_k\mathbf{s}_k)^{\mathrm{T}}}{(\mathbf{v}^{\mathrm{T}}\mathbf{s}_k)^2}\mathbf{v}\mathbf{v}^{\mathrm{T}}. \quad \text{(C19)}$$

The quasi-Newton update matrix in (C19) is of rank two and is well defined for any $\mathbf{v}$ that is not orthogonal to $\mathbf{s}_k$. When $\mathbf{v}$ is taken as $\mathbf{y}_k$ the Davidon–Fletcher–Powell (DFP) update is obtained, i.e.,

$$\mathbf{B}_{k+1}=\mathbf{B}_k-\frac{1}{\mathbf{s}_k{}^{\mathrm{T}}B_k\mathbf{s}_k}\mathbf{B}_k\mathbf{s}_k\mathbf{s}_k{}^{\mathrm{T}}B_k+\frac{1}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{s}_k}\mathbf{y}_k\mathbf{y}_k{}^{\mathrm{T}}$$
$$+(\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k\mathbf{s}_k)\,\mathbf{w}_k\mathbf{w}_k{}^{\mathrm{T}}$$

where

$$\mathbf{w}_k{}^{\mathrm{T}}=\frac{1}{\mathbf{s}_k{}^{\mathrm{T}}\mathbf{s}_k}\mathbf{y}_k-\frac{1}{\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k\mathbf{s}_k}\mathbf{B}_k\mathbf{s}_k. \quad \text{(C20)}$$

One can verify by substitution that the vector $\mathbf{w}_k$ is orthogonal to $\mathbf{s}_k$, and as such, any multiple of the rank-one matrix $\mathbf{w}_k\mathbf{w}_k{}^{\mathrm{T}}$ can be added to $\mathbf{B}_{k+1}$ without affecting the quasi-Newton condition (C12).

This led to the one-parameter family of quasi-Newton updates

$$\mathbf{B}^{\phi}_{k+1}=\mathbf{B}_k-\frac{1}{\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k\mathbf{s}_k}\mathbf{B}_k\mathbf{s}_k\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k+\frac{1}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{s}_k}\mathbf{y}_k\mathbf{y}_k{}^{\mathrm{T}}$$
$$+\phi_k(\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k\mathbf{s}_k)\mathbf{w}_k\mathbf{w}_k{}^{\mathrm{T}} \quad \text{(C21)}$$

where ϕ_k is a scalar depending on $\mathbf{y}_k$ and $\mathbf{B}_k{}^{\phi}\mathbf{s}_k$.

It is now believed, after a considerable amount of research, that the most effective quasi-Newton update is the one corresponding to the choice $\phi_k \equiv 0$, called the Broyden–Fletcher–Goldfarb–Shanno (BFGS) update:

$$\mathbf{B}_{k+1}=\mathbf{B}_k-\frac{1}{\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k\mathbf{s}_k}\mathbf{B}_k\mathbf{s}_k\mathbf{s}_k{}^{\mathrm{T}}\mathbf{B}_k+\frac{1}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{s}_k}\mathbf{y}_k\mathbf{y}_k{}^{\mathrm{T}}. \quad \text{(C22)}$$

Hereditary positive definiteness can be proven for the BFGS quasi-Newton update.

The BFGS quasi-Newton update was found to be greatly superior to the DFP formula in most cases with rather crude linear searches. It is also less prone to loss of the positive-definiteness property through rounding errors, it has better theoretical convergence properties, and it is the simplest rank-two update for implementation. Currently the BFGS quasi-Newton update formula is well accepted.

2. The connection between direct and inverse updates of the Hessian

Earlier quasi-Newton methods were formulated in terms of maintaining an approximation to the inverse Hessian matrix since solving

$$\mathbf{B}_k\mathbf{p}_k=-\mathbf{g}_k \quad \text{(C23)}$$

for the direct approximation from scratch would require order n^3 operations compared with order n^2 operations to form the matrix product

$$\mathbf{p}_k=\mathbf{H}_k\mathbf{g}_k \quad \text{(C24)}$$

where $\mathbf{H}_k=(\mathbf{B}_k)^{-1}$ is an approximation to the inverse Hessian. The quasi-Newton condition for the inverse Hessian is

$$\mathbf{H}_{k+1}\mathbf{y}_k=\mathbf{s}_k. \quad \text{(C25)}$$

Starting with Eq. (93), which is the direct BFGS quasi-Newton update, we can obtain Eq. (100) for the inverse update $\mathbf{H}_{k+1}$ (if $\mathbf{H}_k$ is equal to $\mathbf{B}_k{}^{-1}$) by the following interchanges:

$$\left.\begin{array}{c}\mathbf{B}_k\leftrightarrow\mathbf{H}_k\\ \mathbf{B}_{k+1}\leftrightarrow\mathbf{H}_{k+1}\\ \mathbf{s}_k=\Delta\mathbf{x}_k\leftrightarrow\mathbf{y}_k=\Delta\mathbf{g}_k\\ \alpha_k\leftrightarrow\alpha_k{}^{-1}\end{array}\right\}. \quad \text{(C26)}$$

Now the product of $\mathbf{B}_{k+1}$ in Eq. (93) and $\mathbf{H}_{k+1}$ in Eq. (100) can be shown by straightforward algebraic manipulations to be equal to the unit matrix $\mathbf{I}$, i.e., by establishing

$$\mathbf{B}^{(k+1)}_{\mathrm{BFGS}}\mathbf{H}^{k+1}_{\mathrm{BFGS}}=\mathbf{I}. \quad \text{(C27)}$$

To better see this, one can cast Eqs. (93) and (100) for the direct and inverse BFGS quasi-Newton updates in the forms:

$$\mathbf{B}_{k+1}=\left[I-\frac{\mathbf{y}_k\mathbf{s}_k{}^{\mathrm{T}}}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{s}_k}\right]B_k\left[I-\frac{\mathbf{y}_k\mathbf{s}_k{}^{\mathrm{T}}}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{s}_k}\right]^{\mathrm{T}}+\frac{\mathbf{y}_k\mathbf{y}_k{}^{\mathrm{T}}}{\mathbf{y}_k{}^{\mathrm{T}}\mathbf{s}_k} \quad \text{(C28)}$$

$$\mathbf{H}_{k+1}=\left[I-\frac{\mathbf{s}_k\mathbf{y}_k{}^{\mathrm{T}}}{\mathbf{s}_k{}^{\mathrm{T}}\mathbf{y}_k}\right]H_k\left[I-\frac{\mathbf{s}_k\mathbf{y}_k{}^{\mathrm{T}}}{\mathbf{s}_k{}^{\mathrm{T}}\mathbf{y}_k}\right]^{\mathrm{T}}+\frac{\mathbf{s}_k\mathbf{s}_k{}^{\mathrm{T}}}{\mathbf{s}_k{}^{\mathrm{T}}\mathbf{y}_k}. \quad \text{(C29)}$$

Note that a product $\mathbf{x}^{\mathrm{T}}\mathbf{y}$ gives the usual inner product

$$\langle\mathbf{x},\mathbf{y}\rangle=\mathbf{x}^{\mathrm{T}}\mathbf{y} \quad \text{(C30)}$$

whereas a product $\mathbf{x}\mathbf{y}^{\mathrm{T}}$ is the outer product giving a matrix of rank one.

For further information on quasi-Newton updates, see reviews by Dennis and More (1977) and Gill and Murray (1972) as well as the book by Fletcher (1980).

APPENDIX D

Conjugate-Gradient Minimization of a Quadratic Function

Theorem 4. *If conjugate search directions are employed sequentially, the minimum of a quadratic function of n variables will be found in at most n iterations.*

Proof: Assume the quadratic function

$$F(\mathbf{x})=a+\mathbf{b}^{\mathrm{T}}\mathbf{x}+\frac{1}{2}\mathbf{x}^{\mathrm{T}}G\mathbf{x} \quad \text{(D1)}$$

where

$$G=\left(\frac{\partial^2F}{\partial\mathbf{x}_i\partial\mathbf{x}_j}\right)\equiv\nabla^2F(\mathbf{x}) \quad \text{(D2)}$$

is the positive-definite symmetric Hessian matrix, so that

$$\nabla F(\mathbf{x}) = \mathbf{b} + \mathbf{G}\mathbf{x} \qquad \text{(D3)}$$

and at the minimum, $\mathbf{x}^*$,

$$\nabla F(\mathbf{x}^*) = 0 \qquad \text{(D4)}$$

that is, from (D3),

$$\mathbf{x}^* = -\mathbf{G}^{-1}\mathbf{b}. \qquad \text{(D5)}$$

At the end of the nth iteration of the conjugate-gradient method we have

$$\mathbf{x}_n = \mathbf{x}_0 + \sum_{k=0}^{n-1} \alpha_k \mathbf{d}_k. \qquad \text{(D6)}$$

For each search we determine α_k by a line minimization of $F(\mathbf{x}_k + \alpha_k \mathbf{d}_k)$, that is,

$$\frac{dF(\mathbf{x}_k + \alpha_k \mathbf{d}_k)}{d\alpha_k} = \nabla^{\mathrm{T}} F(\mathbf{x}_k)\mathbf{d}_k + (\mathbf{d}_k)^{\mathrm{T}} \nabla^2 F(\mathbf{x}_k)\alpha_k \mathbf{d}_k = 0 \qquad \text{(D7)}$$

obtained by a Taylor series expansion around $\mathbf{x}_k$.

This yields

$$\alpha_k = \frac{\nabla^{\mathrm{T}} F(\mathbf{x}_k)\mathbf{d}_k}{\mathbf{d}_k^{\mathrm{T}} \nabla^2 F(\mathbf{x}_k)\mathbf{d}_k}. \qquad \text{(D8)}$$

[This relation is used in Eq. (42)]. By substituting α_k into (D6) we obtain

$$\mathbf{x}_n = \mathbf{x}_0 - \sum_{k=0}^{n-1} \left[\frac{(\mathbf{d}_k)^{\mathrm{T}} \nabla F(\mathbf{x}_k)}{\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_k}\right]\mathbf{d}_k. \qquad \text{(D9)}$$

By using Eqs. (D3) and (D6) we get

$$\mathbf{d}_k^{\mathrm{T}} \nabla F(\mathbf{x}) = \mathbf{d}_k^{\mathrm{T}}(\mathbf{G}\mathbf{x} + \mathbf{b}) = \mathbf{d}_k^{\mathrm{T}}\left\{\mathbf{G}\left(\mathbf{x}_0 + \sum_{i=1}^{k-1} \alpha_i \mathbf{d}_i\right) + \mathbf{b}\right\}. \qquad \text{(D10)}$$

We remember that $\mathbf{d}_k$ and $\mathbf{d}_i$ are conjugate directions, i.e.,

$$\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_i = 0 \quad \text{for all } i. \qquad \text{(D11)}$$

Hence Eq. (D10) reduces to

$$\mathbf{d}_k^{\mathrm{T}} \nabla F(\mathbf{x}) = \mathbf{d}_k^{\mathrm{T}}(\mathbf{G}\mathbf{x}^0 + \mathbf{b}) \qquad \text{(D12)}$$

and substituting (D12) into (D9)

$$\mathbf{x}_n = \mathbf{x}_0 - \sum_{k=0}^{n-1} \frac{(\mathbf{d}_k)^{\mathrm{T}}(\mathbf{G}\mathbf{x}_0 + \mathbf{b})}{\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_k}\mathbf{d}_k$$

$$= \mathbf{x}_0 - \sum_{k=0}^{n-1} \left\{\frac{\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_k}{\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_k}\right\}\mathbf{x}_0 - \sum_{k=0}^{n-1} \frac{\mathbf{d}_k^{\mathrm{T}} \mathbf{b}\mathbf{d}_k}{\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_k}. \qquad \text{(D13)}$$

The second term on the right-hand side of Eq. (D13) is just $-\mathbf{x}_0$, and the third and last term, when multiplied by $\mathbf{G} \cdot \mathbf{G}^{-1} = \mathbf{I}$, leads to

$$\mathbf{x}^n = \mathbf{x}_0 - \mathbf{x}_0 - \mathbf{G}\mathbf{G}^{-1} \sum_{k=0}^{n-1} \frac{\mathbf{d}_k^{\mathrm{T}} \mathbf{b}\mathbf{d}_k}{\mathbf{d}_k^{\mathrm{T}} G\mathbf{d}_k}$$

$$= -\sum_{k=0}^{n-1} \frac{\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_k}{\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_k}\mathbf{G}^{-1}\mathbf{b} = -\mathbf{G}^{-1}\mathbf{b} \qquad \text{(D14)}$$

which by Eq. (D5) is $\mathbf{x}^*$, i.e.,

$$\mathbf{x}^n = \mathbf{x}^* \qquad \text{(D15)}$$

i.e., the minimum was found in, at most, n conjugate-gradient searches.

APPENDIX E

Proof of Orthogonality of Conjugate Directions and Gradients for a Quadratic Function Minimization

Starting from Eq. (1), the gradient of $F(\mathbf{x})$ is

$$\mathbf{g}(\mathbf{x}) = \mathbf{b} + \mathbf{G}\mathbf{x} \qquad \text{(E1)}$$

and for another point $\mathbf{x}_k$

$$\mathbf{g}(\mathbf{x}_k) = \mathbf{b} + \mathbf{G}\mathbf{x}_k. \qquad \text{(E2)}$$

From Eq. (8), the line minimization is

$$F(\mathbf{x}_k + \alpha_k \mathbf{d}_k) = \min_{\alpha} F(\mathbf{x}_k + \alpha \mathbf{d}_k) = \frac{dF(\mathbf{x}_k + \alpha \mathbf{d}_k)}{d\alpha} = 0. \qquad \text{(E3)}$$

Taking a quadratic approximation by Taylor series about the point $\mathbf{x}_k$

$$f(\mathbf{x}) = F(\mathbf{x}_k) + \nabla^{\mathrm{T}} F(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_k)^{\mathrm{T}} \nabla^2 F(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) \qquad \text{(E4)}$$

and taking $\mathbf{x} - \mathbf{x}_k = \alpha_k \mathbf{d}_k$ and substituting in (E4) we obtain for Eq. (E5)

$$\frac{dF(\mathbf{x}_k + \alpha \mathbf{d}_k)}{d\alpha} = \nabla^{\mathrm{T}} F(\mathbf{x}_k)\mathbf{d}_k + \mathbf{d}_k^{\mathrm{T}} \mathbf{G}\alpha \mathbf{d}_k = 0 \qquad \text{(E5)}$$

where $G = \nabla^2 F(\mathbf{x}_k)$, the Hessian of F, yielding

$$\alpha = \frac{\nabla^{\mathrm{T}} F(\mathbf{x}_k)\mathbf{d}_k}{\mathbf{d}_k^{\mathrm{T}} \mathbf{G}\mathbf{d}_k} \qquad \text{(E6)}$$

writing

$$\nabla^{\mathrm{T}} F(\mathbf{x}_k) = \mathbf{g}(\mathbf{x}_k) = \mathbf{b} + \mathbf{G}\mathbf{x}_k. \qquad \text{(E7)}$$

We obtain from (E5)

$$(\mathbf{b} + \mathbf{G}\mathbf{x}_k)\mathbf{d}_k + \mathbf{d}_k^{\mathrm{T}} \mathbf{G}\alpha_k \mathbf{d}_k = 0. \qquad \text{(E8)}$$

Introducing

$$\mathbf{x}_{k+1} - \mathbf{x}_k = \alpha_k \mathbf{d}_k \qquad \text{(E9)}$$

we obtain from (E8)

$$\mathbf{d}_k^{\mathrm{T}}(\mathbf{b} + \mathbf{G}\mathbf{x}_k) + \mathbf{d}_k^{\mathrm{T}} \mathbf{G}(\mathbf{x}_{k+1} - \mathbf{x}_k) = 0 \qquad \text{(E10)}$$

or

$$\mathbf{d}_k^{\mathrm{T}}(\mathbf{b} + \mathbf{G}\mathbf{x}_{k+1}) = \mathbf{d}_k^{\mathrm{T}} \mathbf{g}_{k+1} = 0; \qquad \text{(E11)}$$

i.e., the gradient at the point $\mathbf{x}_{k+1}$ in the conjugate-gradient minimization of the quadratic function $F(\mathbf{x})$ is orthogonal to the previous search direction $\mathbf{d}_k$.

To prove now that

$$\mathbf{g}_{l+1}\mathbf{d}_i = 0 \quad \text{for each} \quad \mathbf{i} = 0, 1, \cdots, l-1 \qquad \text{(E12)}$$

we start from

$$\mathbf{g}_l = \mathbf{b} + \mathbf{G}\mathbf{x}_l \qquad \text{(E13)}$$

and from the conjugate directions search we obtain

$$\mathbf{g}_l = \mathbf{b} + \mathbf{G}(\mathbf{x}_k + \sum_{i=k}^{l-1} \alpha_i \mathbf{d}_i) = \mathbf{b} + \mathbf{G}\mathbf{x}_k + \mathbf{G}\sum_{i=k}^{l-1} \alpha_i \mathbf{d}_i$$

$$(\text{as } \mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k) \qquad \text{(E14)}$$

where $\mathbf{x}_k$ is the arbitrary point from which conjugate searches start.

Since

$$\mathbf{g}_k = \mathbf{b} + \mathbf{G}\mathbf{x}_k \qquad \text{(E15)}$$

$$\mathbf{g}_l = \mathbf{g}_k + \sum_{i=k}^{l-1} \alpha_i \mathbf{G}\mathbf{d}_i \qquad \text{(E16)}$$

premultiplication of this Eq. (E16) by $\mathbf{d}_{k-1}^{\mathrm{T}}$ yields

$$\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_l = \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_k + \sum_{i=k}^{l-1} \alpha_i \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{G}\mathbf{d}_i. \qquad \text{(E17)}$$

The first term on the right-hand side of (E17) vanishes since we have already proven that the gradient at a point is orthogonal to the previous search direction if the quadratic function is minimized in that search direction [Eq. (E11)]. All of the terms in the sum of the second term on the right-hand side of Eq. (E17) vanish because of conjugacy. Hence

$$\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_l = 0 \qquad \text{(E18)}$$

and because the analysis above is still valid for k having any index number between 1 and l

$$\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_l = 0 \quad 0 \leqslant i \leqslant l-1. \qquad \text{(E19)}$$

REFERENCES

Adams, L., 1983: An M-step preconditioned conjugate-gradient method for parallel computation. *Proc. Int. Conf. on Parallel Processing,* Silver Spring, IEEE Computer Society Press and Association for Computing Machinery, 36–43.

Beale, E. M., 1972: A derivation of conjugate-gradients. *Numerical Methods for Non-linear Optimization.* F. A. Lootsma, Ed., Academic Press, 39–43.

Bertocchi, M., and E. Spedicato, 1979: Computational experience with conjugate-gradient algorithms. *Calcolo,* **15,** 255–269.

Bertsekas, D. P., 1982: *Constrained Optimization and Lagrange Multiplier Methods,* Academic Press, 395 pp.

Broyden, C. G., 1970: The convergence of a class of double-rank minimization algorithms. *J. Inst. Math. Applic.,* **6,** 76–90.

Buckley, A. G., 1978: A combined conjugate-gradient quasi-Newton minimization algorithm. *Math. Program.,* **15,** 343–348.

——, 1982: Conjugate-gradient methods. *Nonlinear Optimization,* M. J. D. Powell, Ed., *Proc. NATO Advanced Research Institute on Nonlinear Optimization,* Academic Press, 17–22.

——, and A. Lenir, 1982: QN-like variable storage conjugate-gradients. MIS. Tech. Rep. 82-2, University of Arizona.

——, and ——, 1983: QN-like variable-storage conjugate gradients. *Math. Program.,* **27,** 155–175.

——, and ——, 1985: Algorithm 630-BBVSCG. A variable storage algorithm for function minimization. *ACM. Trans. Math. Software,* **11**(2), 103–119.

Concus, P., G. H. Golub and D. P. O'Leary, 1976: A generalized conjugate-gradient method for the numerical solution of elliptic partial differential equations. *Sparse Matrix Computations,* J. R. Bunch and D. J. Rose, Eds., Academic Press.

Courtier, P., 1985: Use of adjoint equations for assimilation of meteorological observations by barotropic models. *Int. Symp. on Variational Methods in Geosciences,* Norman, OK, CIMMS, 5–6, 128 pp.

Davidon, W. C., 1959: Variable metric method for minimization. A.E.C., R. and D. Rep. ANL-5990(Rev), Argonne National Laboratory.

Dembo, R. S., and T. Steihaug, 1983: Truncated-Newton algorithms for large-scale unconstrained minimization. *Math. Program.,* **26,** 190–212.

Dennis, J. E., and J. R. More, 1977: Quasi-Newton methods, motivation and theory. *SIAM Rev.,* **19**(1), 46–89.

Derber, J. C., 1985: The variational four-dimensional assimilation of analyses using filtered models as constraints. Ph.D. thesis, University of Wisconsin, 142 pp.

Dixon, L. C. W., P. G. Ducksbury and P. Singh, 1982: A parallel version of the conjugate-gradient algorithm for finite-element problems. Tech Rep. 127, Numerical Optimizational Centre, The Hatfield Polytechnic.

Fletcher, R., 1970: A new approach to variable metric algorithms. *Computer J.,* **13,** 317–322.

——, 1972: Conjugate direction methods. *Numerical Methods for Unconstrained Optimization,* W. Murray, Ed., Academic Press, 144 pp.

——, 1980: *Practical Methods of Optimization, Vol. 1. Unconstrained Optimization,* Wiley and Sons, 120 pp.

——, and C. M. Reeves, 1964: Function minimization by conjugate-gradients. *Computer J.,* **7,** 149–153.

Gill, P. E., and W. Murray, 1972: Quasi-Newton methods for unconstrained optimization. *J. Inst. Math. Applic.,* **9,** 91–108.

——, and ——, 1979: Conjugate-gradient methods for large-scale optimization. Tech. Rep. SOL 79-15, Systems Optimization Lab., Dept. of Operations Research, Stanford, 60 pp.

——, —— and M. H. Wright, 1981: *Practical Optimization,* Academic Press, 401 pp.

——, ——, M. A. Saunders and M. S. Wright, 1982: A note on a sufficient decrease criterion for a non-derivative step-length procedure. *Math. Program.,* **23,** 349–352.

Goldfarb, D., 1970: A family of variable metric methods derived by variational means. *Math. Comput.,* **24,** 23–26.

Goldstein, A. A., and J. F. Price, 1967: An effective algorithm for minimization. *Numer. Math.,* **10,** 184–189.

Golub, G. N., and C. F. Van Loan, 1983: *Matrix Computations.* The Johns Hopkins University Press, 476 pp.

Hestenes, M. R., 1980: Conjugate directions methods in optimization. *Applications of Mathematics,* **12,** Springer-Verlag, 325 pp.

——, and E. Stieffel, 1952: Methods of conjugate-gradients for solving linear systems. *J. Res. Natl. Bur. Stand.,* **48,** 409–436.

Himmelblau, D. M., 1972: *Applied Nonlinear Programming.* McGraw-Hill, 498 pp.

Hoffmann, R. N., 1982: SASS wind ambiguity removal by direct minimization. *Mon. Wea. Rev.,* **110,** 434–445.

——, 1984: SASS wind ambiguity removal by direct minimization. Part II: Use of smoothness and dynamic constraints. *Mon. Wea. Rev.,* **112,** 1829–1852.

Housos, E. C., and O. Wing, 1980: Parallel nonlinear minimization by conjugate-directions. *Proc. Conf. on Parallel Processing,* Los Alamitos, IEEE Computer Society and Association for Computer Machinery, 157–158. [IEEE Cat. No. 80CH1569.]

——, and ——, 1984: Pseudo-conjugate directions for the solution of the nonlinear unconstrained optimization problem on a parallel computer. *J. Optimization Theory Appl.,* **42**(2), 169–180.

IMSL, 1977: IMSL Library Reference Manual. International mathematical and statistical libraries, Houston, pp. ZXCGR-1–ZXCGR-4.

Kontarev, G., 1980: The adjoint equation technique applied to meteorological problems. ECMWF Tech. Rep. No. 21, 21 pp.

Kowalik, J., and S. P. Kumar, 1982: An efficient parallel block conjugate-gradient method for linear equations. *Proc. Int. Conf. on Parallel Processing,* 47–52.

Le, D., 1985: A fast and robust unconstrained minimization method requiring minimum storage. *Math. Program.,* **32,** 41–68.

Le-Dimet, F. X., 1982: A general formalism of variational analysis. CIMMS Rep. No. 22, Cooperative Institute for Mesoscale Meteorological Studies, 34 pp.

——, and J. Segot, 1985: Variational analysis of wind field and geopotential at 500 mb. *Proc. Int. Symp. on Variational Methods in Geosciences,* Norman, CIMMS, 13–15, 128 pp.

——, and O. Talagrand, 1986: Variational algorithms for analysis and assimilation of meteorological observations: Theoretical aspects. *Tellus,* **38A,** 97–110.

Lemarechal, C., 1980: Nondifferentiable optimization. *Nonlinear Optimization Theory and Algorithms,* L. C. W. Dixon, E. Spedicato and G. P. Szegö, Eds., Birkhäuser, 149–199.

——, J. J. Strodiot and A. Bihain, 1981: On a bundle-algorithm for non smooth optimization, *Nonlinear Programming,* Vol. 4, O. L. Mangasarian, R. R. Meyer and S. M. Robinson, Eds., Academic Press, 245–282.

Lewis, J. M., and J. C. Derber, 1985: The use of adjoint equations to solve variational adjustment problems with advective constraints. *Tellus,* **37A,** 309–322.

Manteuffel, T. A., 1979: Shifted incomplete Choleski factorization, *Sparse Matrix Proceedings, 1978,* I. S. Duff and G. W. Stewart, Eds., SIAM Publications, 41–62, 334 pp.

Marchuk, G. I., 1974: *Numerical Solution of the Problems of the Dynamics of the Atmosphere and Ocean,* Leningrad, Gidrometeoizdat, 303 pp (in Russian).

NAG 1984: Fortran Library Reference Manual (Mark 11), Numerical Algorithms Group, 3(E04), 1–6, 370 pp. [Available from NAG Limited, Oxford OX7-2DE, England, or NAG Inc., Downers Grove, IL 70515.]

Navon, I. M., 1982: A Numerov–Galerkin technique applied to a shallow-water equation model with exact conservation of integral invariants, *Finite-Element Flow-Analysis,* Tadahiko Kawai, Ed., North-Holland, 57–67, 1096 pp.

——, 1983: A Numerov–Galerkin technique applied to a finite-element shallow-water equations model with enforced conservation of integral invariants and selective lumping. *J. Comput. Phys.,* **52**(2), 313–339.

——, 1984: Applications of augmented-Lagrangian methods in meteorology—comparative study of different conjugate-gradient routines for a large-scale problem. NASA Research Review 1983, NASA/TM 86053, Global Modeling and Simulation Branch, NASA/Goddard Space Flight Center, 150–156.

——, 1985: A review of variational and optimization methods in meteorology. *Proc. Int. Symp. on Variational Methods in Geosciences,* Norman, OK CIMMS, 1–4, 128 pp.

——, 1986: A review of variational and optimization methods in meteorology, *Variational Methods in the Geosciences,* Y. K. Sasaki, Ed., Elsevier Science Publishing, Amsterdam, 309 pp.

——, and R. de Villiers, 1983: Combined penalty-multiplier optimization methods to enforce integral invariants conservation. *Mon. Wea. Rev.,* **111,** 1228–1243.

——, and ——, 1986: GUSTAF: A quasi-Newton nonlinear ADI Fortran IV program for solving the shallow-water equations with augmented-Lagrangians. *Comput. Geosci.* **12**(2), 151–170.

Nazareth, L., 1979: A relationship between BFGS and conjugate-gradient algorithms and its implications for new algorithms. *SIAM J. Numer. Anal.,* **16,** 794–800.

——, and J. Nocedal, 1982: Conjugate direction methods with variable storage. *Math. Program.,* **23,** 326–340.

Neethling, J. D., 1977: The incomplete-Choleski conjugate-gradient method for the iterative solution of large sparse linear systems. Atomic Energy Board, Pelindaba, Pretoria, Per-19-1, 15 pp.

Nocedal, J., 1980: Updating quasi-Newton matrices with limited storage. *Math. Comput.,* **35,** 773–782.

Ortega, J. M., and W. C. Rheinboldt, 1976: *Iterative Solution of Nonlinear Equations in Several Variables,* Academic Press.

Perry, A., 1976: A modified conjugate-gradient algorithm. Discussion Pap. No. 229, Center for Mathematical Studies in Economics and Management Sciences, Northwestern University.

——, 1977: A class of conjugate-gradient algorithms with a two-step variable metric memory. Discussion Pap. No. 269, Center for Mathematical Studies in Economics and Management Sciences, Northwestern University.

——, 1978: A modified conjugate-gradient algorithm. *Operations Res.,* **26,** 1073–1078.

Polak, E., 1971: *Computational Methods in Optimization, A Unified Approach,* Academic Press, 321 pp.

——, and G. Ribiere, 1969: Note sur la convergence de methodes de directions conjuguées. *Rev. Franç. Informat. Rech. Operationnelle,* **16,** 35–43.

Powell, M. J. D., 1971: On the convergence of a variable metric algorithm. *J. Inst. Math. Applic.,* **7,** 21–36.

——, 1976: Some convergence properties of the conjugate-gradient method. *Math. Prog.,* **11,** 42–49.

——, 1977: Restart procedures for the conjugate-gradient method. *Math. Program.,* **12,** 241–254.

Sasaki, Y. K., 1976: Variational design of finite-difference schemes for initial value problems with an integral invariant. *J. Comput. Phys.,* **21,** 270–278.

Schittkowshi, K., 1985: Software for mathematical programming. *Computational Mathematical Programming,* Vol. 15, NATO ASI Series F: Computer and System Sciences, K. Schittkowshi, Ed., Springer-Verlag, 384–451, 451 pp.

Schnabel, R. B., 1985: Parallel computing in optimization. *Computational Mathematical Programming,* Vol. 15, NATO ASI Series F: Computer and System Sciences, K. Schittkowshi Ed., Springer-Verlag, 357–381, 451 pp.

Shanno, D. F., 1970: Conditioning of quasi-Newton methods for function minimization. *Math. Comp.,* **24,** 647–657.

——, 1978a: Conjugate gradient methods with inexact searches. *Math. Operations Res.,* **3,** 244–256.

——, 1978b: On the convergence of a new conjugate-gradient method. *SIAM J. Numer. Anal.,* **15,** 1247–1257.

——, 1983: Large-scale unconstrained optimization. *Computers Chem. Eng.,* **7**(5), 569–574.

——, 1985: Globally convergent conjugate-gradient algorithms. *Math. Program.,* **33,** 61–67.

——, and K. H. Phua, 1980: Remark on algorithm 500—a variable method subroutine for unconstrained nonlinear minimization. *ACM Trans. on Mathematical Software,* **6,** 618–622.

——, and R. E. Marsten, 1982: Conjugate-gradient methods for linearly constrained nonlinear programming. *Math. Program. Study,* **16,** 149–161.

Talagrand, O., 1985: The adjoint model technique and meteorological applications, Proc. Workshop on High-Resolution Analysis, ECMWF, U.K., 277–297.

Testud, J., and M. Chong, 1983: Three-dimensional wind field analysis from dual Doppler radar data. Part I: Filtering, interpolating and differentiating the raw data. *J. Climate Appl. Meteor.,* **22,** 1204–1215.

Van Laarhoven, P., 1985: Parallel algorithms for unconstrained optimization. *Math. Program.,* **33,** 68–81.

Walsh, G. R., 1975: *Methods of Optimization,* Wiley and Sons, 200 pp.

Wolfe, P., 1975: A method of conjugate subgradients for minimizing non-differentiable functions. *Math. Program. Study,* **3,** 145–173.

Zowe, J., 1985: Nondifferentiable optimization. *Computational Mathematical Programming,* Vol. 15, Klaus Schittkowski, Ed., NATO ASI Series F. Computer and System Sciences, Springer-Verlag, 323–356.