EUR 4777 e

COMMISSION OF THE EUROPEAN COMMUNITIES

A NEW ALGORITHM TO MINIMIZE FUNCTIONS

by

E. VAN DER VOORT and B. DORPEMA

1972

Joint Nuclear Research Center
Ispra Establishment - Italy
Scientific Data Processing Centre - CETIS
and
Materials Division

LEGAL NOTICE

This document was prepared under the sponsorship of the Commission of the European Communities.

Neither the Commission of the European Communities, its contractors nor any person acting on their behalf :

make any warranty or representation, express or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this document, or that the use of any information, apparatus, method or process disclosed in this document may not infringe privately owned rights ; or

assume any liability with respect to the use of, or for damages resulting from the use of any information, apparatus, method or process disclosed in this document.

This report is on sale at the addresses listed on cover page 4

at the price of B.Fr. 50.—

When ordering, please quote the EUR number and the title, which are indicated on the cover of each report.

Luxembourg, February 1972
Printed by Vanmelle, Gand

This document was reproduced on the basis of the best available copy.

EUR 4777 e

A NEW ALGORITHM TO MINIMIZE FUNCTIONS by E. VAN DER VOORT and B. DORPEMA

Commission of the European Communities
Joint Nuclear Research Center — Ispra Establishment (Italy)
Scientific Data Processing Centre — CETIS
and
Materials Division
Luxembourg, February 1972 — 32 pages — B.Fr. 50.—

The modified Newton-Raphson method is outlined and a general strategy is developed using at each iteration stage the steepest descent method, the modified Newton-Raphson method or the usual Newton-Raphson method. An optimized FORTRAN computer programme (MINIM) is described and some working examples are considered as tests.

EUR 4777 e

A NEW ALGORITHM TO MINIMIZE FUNCTIONS by E. VAN DER VOORT and B. DORPEMA

Commission of the European Communities
Joint Nuclear Research Center — Ispra Establishment (Italy)
Scientific Data Processing Centre — CETIS
and
Materials Division
Luxembourg, February 1972 — 32 pages — B.Fr. 50.—

The modified Newton-Raphson method is outlined and a general strategy is developed using at each iteration stage the steepest descent method, the modified Newton-Raphson method or the usual Newton-Raphson method. An optimized FORTRAN computer programme (MINIM) is described and some working examples are considered as tests.

EUR 4777 e

A NEW ALGORITHM TO MINIMIZE FUNCTIONS by E. VAN DER VOORT and B. DORPEMA

Commission of the European Communities
Joint Nuclear Research Center — Ispra Establishment (Italy)
Scientific Data Processing Centre — CETIS
and
Materials Division
Luxembourg, February 1972 — 32 pages — B.Fr. 50.—

The modified Newton-Raphson method is outlined and a general strategy is developed using at each iteration stage the steepest descent method, the modified Newton-Raphson method or the usual Newton-Raphson method. An optimized FORTRAN computer programme (MINIM) is described and some working examples are considered as tests.

EUR 4777 e

COMMISSION OF THE EUROPEAN COMMUNITIES

A NEW ALGORITHM TO MINIMIZE FUNCTIONS

by

E. VAN DER VOORT and B. DORPEMA

1972

Joint Nuclear Research Center
Ispra Establishment - Italy
Scientific Data Processing Centre - CETIS
and
Materials Division

ABSTRACT

The modified Newton-Raphson method is outlined and a general strategy is developed using at each iteration stage the steepest descent method, the modified Newton-Raphson method or the usual Newton-Raphson method. An optimized FORTRAN computer programme (MINIM) is described and some working examples are considered as tests.

KEYWORDS

NEWTON METHOD
FORTRAN
M-CODES
FUNCTIONS

CONTENTS *)

*) Manuscript received on January 5, 1972

A NEW ALGORITHM TO MINIMIZE FUNCTIONS

E. van der Voort, Materials Div.,
B. Dorpema, CETIS
EURATOM C.C.R., Ispra (Va.), Italy

INTRODUCTION

A great deal of problems occurring in applied mathematics may be reduced to the search of a local minimum.

Different methods have been developed that can be classified in three ways, each having its advantages and drawbacks.

a) The gradient methods in general converge very fast in the first iterations but fail in the precise location of the minimum. On the other hand, the fact that the time to compute the function and its gradient at each iteration, is less than with the other methods, where second-order information must be computed, permits a greater number of iterations.
Usually, the gradient methods become less efficient as the number of variables N increases and the number of iterations, in order to have some required precision, is proportional to N.

b) Relaxation or overrelaxation methods may be used in some minimization problems with the property that the matrix of the second-order derivatives (Hessian) is a positive definite band matrix. Being more precise, they have the same general features as the gradient methods.

c) The classical second-order method is the Newton-Raphson (N-R) method, which is much more powerful than the gradient methods. Its advantage is that the efficiency increases in the vicinity of the minimum and that convergence is obtained in a number of steps independent of N.

There are, however, strong limitations to its normal use:

1. As N is large, the computer volume to store the Hessian $\hat{H}$ and to use this information, must increase with N^2;
2. The inversion time of $\hat{H}$ (roughly proportional to N^3) may become too big with respect to the computation time of the function f, the gradient $\vec{g}$ and the Hessian $\hat{H}$.

These facts may counterbalance the advantage one hopes to derive from the N-R method.

The strongest limitation of the usual N-R method is that $\hat{H}$ needs to be positive definite, otherwise convergence may be achieved towards a stationary point, which not necessarily coincides with the searched local minimum. The aim of the new method presented here, is to by-pass this drawback using a modification of the N-R method.

The modified N-R method [1] , developed by Fiacco and McCormick and described in section A, solves this problem in that, using directions corresponding to negative eigenvalues, it converges to the zone of positive definiteness of the Hessian $\hat{H}$, where the usual N-R method very rapidly finds the local minimum. The general strategy set up in section B is the basis for a new computer procedure that is fully described in sections C, D, E and F. Some examples are given in section G.

A. The Modified Newton-Raphson Method

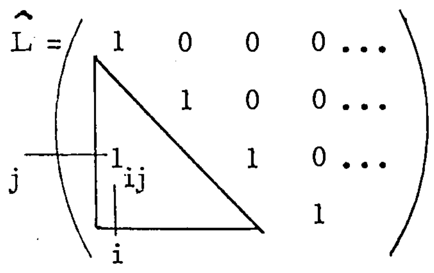
The method is based on the factorization of the Hessian $\hat{H}$ in:

$$\hat{H} = \hat{L} \cdot \hat{D} \cdot \hat{L}^T \tag{1}$$

where $\hat{L}$ is a lower triangular matrix with the elements on the main diagonal equal to 1, $\hat{D}$ is a pure diagonal matrix and $\hat{L}^T$ is the transpose of $\hat{L}$.

From (1) several elementary properties may be deduced:

(a) $\hat{H}$ is a symmetric matrix and the factorization is unambiguous in that there are as many unknowns as different elements in $\hat{H}$. Denoting the elements of $\hat{L}$ by l_{ij} and the diagonal elements of $\hat{D}$ by d_i, the first index denoting the column and the second the row, both running from 1 to N, one has a priori:



$$h_{ij} = h_{ji}$$

$$l_{ij} = 0 \qquad \text{for } i > j$$

$$l_{ij} = 1 \qquad \text{for } i = j$$

Formally it may easily be derived that:

$$d_i = h_{ii} - \sum_{k=1}^{i-1} d_k \cdot l_{ki}^2 \tag{2}$$

and supposing that none of the d_i's is zero:

$$l_{ij} = \frac{1}{d_i}\left(h_{ij} - \sum_{k=1}^{i-1} d_k \cdot l_{ki} \cdot l_{kj}\right) \quad i < j \tag{3}$$

Obviously, the sums have to be understood empty for i = 1. A necessary and sufficient condition that factorization (1) be possible is that none of the d_i's vanishes, i.e. none of the principal minors of $\hat{H}$ may be zero [2]. From now on it will be understood that this condition always is fulfilled.

(b) Denoting the eigenvalues of $\hat{H}$ by λ_i, one has:

$$\prod_{i=1}^{N} \lambda_i = \det(\hat{H}) = \det(\hat{L}) \cdot \det(\hat{D}) \cdot \det(\hat{L}^T) = \det(\hat{D}) = \prod_{i=1}^{N} d_i \tag{4}$$

Thus, if none of the λ_i's is zero, none of the d_i's will be zero and vice versa.

(c) A necessary and sufficient condition that $\hat{H}$ be positive definite is that all elements of $\hat{D}$ in factorization (1) be positive. Indeed, then and only then $\hat{D}$

may be written as $\widehat{D^{1/2}} \cdot \widehat{D^{1/2}}$ where $\widehat{D^{1/2}}$ is a diagonal matrix consisting of the elements $\sqrt{d_i}$. Factorization (1) takes the Choleski form:

$$\hat{H} = \hat{L}^{*} \cdot \hat{L}^{*\,T} ,$$

where $\hat{L}^{*} = \hat{L} \cdot \widehat{D^{1/2}}$, i.e. a necessary and sufficient condition that $\hat{H}$ be positive definite.

As said in the introduction, the modified N-R method is used when $\hat{H}$ is <u>not</u> positive definite. Some of the d_i's thus are less than zero, and it is always possible to build up a non-zero vector $\vec{a}$ consisting of elements zero or one depending on whether the corresponding element d_i is positive or not.

All variables of the function f may be gathered together into a vector $\vec{x}$. Let $\vec{x}_o$ be the initial point of the iteration step. The information of the previous iteration contains the function value $f(\vec{x}_o)$, the gradient vector $\vec{g}(\vec{x}_o)$ and the Hessian $\hat{H}(\vec{x}_o)$. In order to get a better guess of the local minimum, a direction $\vec{s}$ must be chosen in which the function is minimized. The one-dimensional problem to find the good λ that minimizes $f(\vec{x}_o + \lambda \vec{s})$ is treated in section B.

The modified N-R method consists in finding a direction $\vec{s}$ such that:

$$\frac{d}{d\lambda}\left[f(\vec{x}_o) + \lambda \vec{s}) \right]_{\lambda=0} = (\vec{g}(\vec{x}_o), \vec{s}) < 0 \tag{5}$$

and

$$\frac{d^2}{d\lambda^2}\left[f(\vec{x}_o + \lambda\vec{s}) \right]_{\lambda=0} = (\vec{s}, \hat{H}(\vec{x}_o)\vec{s}) < 0 \tag{6}$$

This may be achieved when not all d_i's are positive. Indeed, define the vector $\vec{a}$ when the elements a_i are 0 or 1 depending whether the corresponding d_i is positive or negative respectively; then construct the vector

$$\vec{t} = \widehat{(L^T)^{-1}} \cdot \vec{a} \tag{7}$$

and compute $(\vec{g}(\vec{x}_o), \vec{t})$. If this is positive, take $\vec{s} = -\vec{t}$, if negative, take

$\vec{s} = +\vec{t}$. So in any case $\vec{s}$ is found satisfying (5).

Regarding condition (6), an important lemma may be derived:

$$(\vec{s}, \hat{H}(\vec{x}_o)\vec{s}) \quad = \text{sum of the negative } d_i\text{'s} \tag{8}$$

and thus itself negative satisfying (6). Indeed:

$$\begin{aligned}(\vec{s}, \hat{H}(\vec{x}_o)\vec{s}) = (\vec{t}, \hat{H}(\vec{x}_o)\vec{t}) &= \left[\widehat{(L^T)^{-1}}\vec{a}, \hat{L}.\hat{D}.\hat{L}^T.\widehat{(L^T)^{-1}}.\vec{a}\right] \\ &= \left[\widehat{(L^{-1})^T}\vec{a}, \hat{L}.\hat{D}.\vec{a}\right] \\ &= \left[\vec{a}, \hat{L}^{-1}.\hat{L}.\hat{D}.\vec{a}\right] \\ &= \left[\vec{a}, \hat{D}\vec{a}\right] \quad = \text{sum of the negative } d_i\text{'s}\end{aligned}$$

From $\vec{x}_o$ on in the direction $\vec{s}$, the function not only decreases but the curvature is negative too. Unless this curvature does not change sign, the function value becomes $-\infty$. In this case the local minimum in the N-dimensional space is found within the iteration step itself. Normally, the curvature changes sign and there is a one-dimensional local minimum in the $\vec{s}$ direction. Each iteration with the modified N-R method finds out a direction along which the curvature is turned over from negative to positive. This method provides thus a tool to locate a region where the Hessian $\hat{H}$ is positive definite and where the usual N-R method finds the corresponding local minimum in an optimized way.

B. The Minimizing Strategy

Having a point $\vec{x}_o$, each iteration chooses a direction $\vec{s}$ minimizing the function $f(\vec{x}_o + \lambda\vec{s})$ for some positive λ_{min}. The problem is split up in two parts: (1) to choose $\vec{s}$ and (2) to find λ_{min}.

1. The Direction $\vec{s}$.

If $\hat{H}$ is positive definite at $\vec{x}_o$, the usual N-R method is used and

$$\vec{s} = -\hat{H}^{-1}(\vec{x}_o) \cdot \vec{g}(\vec{x}_o) \tag{9}$$

If $\hat{H}$ is not positive definite at $\vec{x}_o$, a choice is made between the steepest descent method and the modified N-R method. In the first few iterations, experience has shown that the steepest descent makes considerable progress in lowering the function value. In that case simply:

$$\vec{s} = -\vec{g}(\vec{x}_o) \tag{10}$$

When the number of iterations increases, the first order methods stop their efficiency and (always if $\hat{H}$ is not positive definite) the modified N-R method is used. $\vec{s}$ is then defined as in section A.

2. The Minimizing λ.

Roughly the same method is used as that of R. Fletcher and C. M. Reeves [3].

At first, an estimate for λ_{min} is to be found. This will be dependent on how the direction $\vec{s}$ has been chosen. It must be remarked that in each of the three cases:

$$\frac{d}{d\lambda}\left[f(\vec{x}_o + \lambda\vec{s}\,\right]_{\lambda=0} \equiv (\vec{g}(\vec{x}_o), \vec{s}) < 0 \tag{11}$$

Indeed: - for the modified N-R method, this is already proven (see (5));

- for the usual N-R method: $\hat{H}(\vec{x}_o)\vec{s} = -\vec{g}(\vec{x}_o)$ and $(\vec{g}(\vec{x}_o), \vec{s}) = -(\vec{s}, \hat{H}(\vec{x}_o)\vec{s}) < 0$ for then $\hat{H}(\vec{x}_o)$ is positive definite;
- for the steepest descent method obviously: $(\vec{g}(\vec{x}_o), \vec{s}) = -\left|\vec{g}(\vec{x}_o)\right|^2 < 0.$

This proves (11) in a general way with the conclusion that λ_{min} must be positive.

When $(\vec{s}, \hat{H}(\vec{x}_o)\,\vec{s})$ is positive, the function $f(\vec{x}_o + \lambda\,\vec{s})$ may be approximated by its Taylor series truncated after the λ^2-term. The minimum estimate is then:

$$\lambda_{est} = -\frac{(\vec{g}(\vec{x}_o), \vec{s})}{(\vec{s},\ \hat{H}(\vec{x}_o)\vec{s})}$$

When $(\vec{s},\ \hat{H}(\vec{x}_o)\vec{s})$ is negative, having no third order information of $f(\vec{x}_o + \lambda\vec{s})$

at $\lambda = 0$, the supposition is made that $f(\vec{x}_o + \lambda \vec{s})$ becomes $-\infty$ for some λ. The function is then approximated by the form: $A \log(\lambda_{est} - \lambda) + B$, having in second order contact at $\lambda = 0$.

It is easily proved that $\lambda_{est} = + \dfrac{(\vec{g}(\vec{x}_o), \vec{s})}{(\vec{s}, \hat{H}(\vec{x}_o)\vec{s})}$ °).

In both cases λ_{est} satisfies thus:

$$\lambda_{est} = - \frac{(\vec{g}(\vec{x}_o), \vec{s})}{\left|(\vec{s}, \hat{H}(\vec{x}_o)\vec{s})\right|} \tag{12}$$

In a second step the bounds λ_a and λ_b on λ_{min} are sought. The function and the gradient are calculated successively at the points $\vec{x}_o + i \lambda_{est} \vec{s}$ for $i = 1, 2, 4, 8, \ldots$ till a function value is found greater than that in the preceding point or, till the derivative of the function in the $\vec{s}$-direction no longer is negative. The upper bound λ_b is then taken to be this last $i\, \lambda_{est}$ - value while the least bound λ_a corresponds with the preceding $i\, \lambda_{est}$- value.

In the third step, a cubic interpolation (Davidon) is made localizing λ_{min} between λ_a and λ_b. Denoting $f(\vec{x}_o + \lambda_a \vec{s})$ and $f(\vec{x}_o + \lambda_b \vec{s})$ respectively by f_a and f_b and the derivatives of f along $\vec{s}$ in these points by g_a and g_b, one defines:

$$z = 3 \frac{f_a - f_b}{\lambda_a - \lambda_b} + g_a + g_b \tag{14}$$

and

$$w = [\, z^2 - g_a \cdot g_b \,]^{1/2} \tag{15}$$

The minimum is then approximated by:

°) Taking e.g. a hyperbola $\dfrac{A}{\lambda - \lambda_{est}} + B$, one has

$\lambda_{est} = 2 \dfrac{(\vec{g}(\vec{x}_o), \vec{s})}{(\vec{s}, \hat{H}(\vec{x}_o)\vec{s})}$ finding the same order of magnitude.

$$\lambda_{min} = \lambda_b - \frac{(g_b + w - z)(\lambda_b - \lambda_a)}{(g_b - g_a + 2w)} \qquad (16)$$

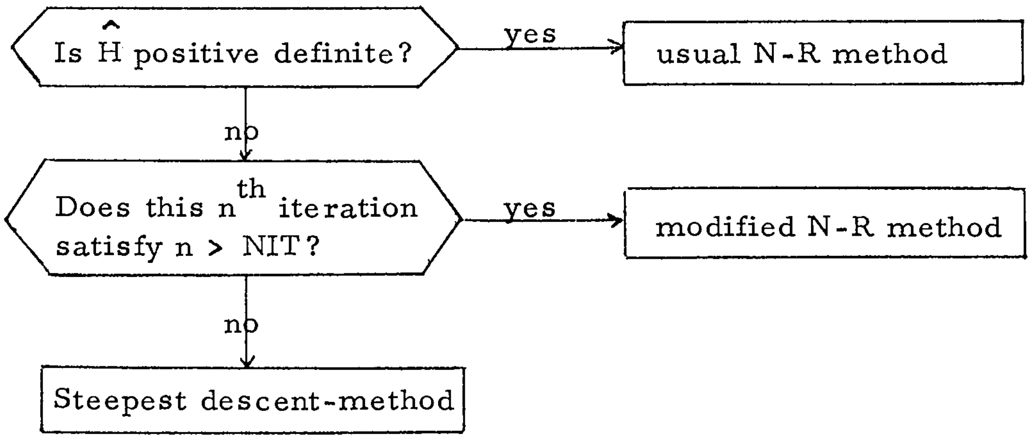
At the point $(\vec{x}_o + \lambda_{min}\vec{s})$, the function f_m is calculated as well as the gradient g_m and the Hessian $\hat{H}_m$.

If $f_m > f_a$ then λ_{min} is taken for λ_b and the interpolation is repeated; otherwise one asks if $f_m > f_b$. If true, λ_{min} is taken for λ_a and the interpolation is repeated. If not, λ_{min} is taken as the minimizing λ.
The point $(\vec{x}_o + \lambda_{min}\vec{s})$ and the information contained in f_m, $\vec{g}_m$ and $\hat{H}_m$ is used to start the next iteration. Usually only one interpolation has to be made, seldom two.

The remark must be made that when $\hat{H}(\vec{x}_o)$ is positive definite, $\lambda_{est} = +1$ for $\vec{s}$ satisfies (9). Unless $f(\vec{x}_o + \vec{s})$ is not greater than $f(\vec{x}_o)$, no interpolation must be made in this case, in order to insure convergence in the final iteration steps.

C. General Description of the MINIM-Subroutine

The programme, called MINIM, is an iterative subroutine that minimizes a function of many variables. At each iteration a guess of the minimum is available and a choice is made between 3 strategies, according to the following scheme:


Is $\hat{H}$ positive definite?
yes
usual N-R method
no
Does this nth iteration satisfy n > NIT?
yes
modified N-R method
no
Steepest descent-method

As mentioned in section B, the steepest descent method is usually very effi-

cient in localizing roughly the minimum but fails in its precise determination. The number NIT varies from problem to problem and must therefore be provided by the user. When $\hat{H}$ is positive definite, however, the most efficient usual N-R method is used at once.
Convergence is proposed if some of the criteria A, B, C or D are satisfied, where:

A means: the number of iterations exceeds IMAX

B means: $\Delta \vec{x} = \left| \vec{x}_{new} - \vec{x}_{old} \right| < \text{EPSX}$

C means: $\left| f(\vec{x}_{new}) - f(\vec{x}_{old}) \right| < \text{EPSF}$

D means: $\left| \vec{g}(\vec{x}_{new}) \right| \leqslant \text{EPSG}$

Obviously, only criterion D satisfies the definition of a local minimum (the extremum is a minimum because $\hat{H}$ is positive definite there) and the user should try to adjust the parameters IMAX, EPSX, EPSF and EPSG to his problem so that the RETURN statement is caused by criterion D. The other criteria are to be considered as security switches. It may happen that the user cannot supply one or more of these parameters. If so, the parameter should be given a value $\leqslant 0$, then it will be adjusted (only on input not on return) to a standard value. These standard values are 10^{-8} for EPSX, EPSF, EPSG and 40 for IMAX. Furthermore, if NIT < 0, it is adjusted to the standard NIT $= \sqrt[3]{2N}$, where N is the number of variables in the function to be minimized. These assigned standard values have been shown to be adequate for most problems.

The listing of MINIM is shown in Appendix 1.

D. Calling Sequence of MINIM with Some Remarks

SUBROUTINE MINIM (FUN, FM, X, G, H, N, M, IRIT, EPSF, EPSX, EPSG, IMAX, NIT)

FUN - Name of function to be minimized. In the calling programme this name must be defined by an EXTERNAL statement.

FM - Function value of the minimum estimate of the last iteration before return.

X - Vector of independent variables: initial minimum guess on input and final minimum estimate before return. X is input data to "FUN".

G - Vector containing the gradient at X. G must be calculated by "FUN".

H - Vector containing the second order partial derivatives (Hessian) at X. H must be calculated by "FUN". Since the symmetric matrix $\hat{H}$ is stored in lower triangular mode, care must be taken for correct indexing.
$H(1) = H_{1,1}$; $H(2) = H_{1,2} = H_{2,1}$; $H(3) = H_{2,2}$; $H(4) = H_{3,1} = H_{1,3}$; $H(5) = H_{3,2} = H_{2,3}$; $H(6) = H_{3,3}$; $H(7) = H_{1,4}$, etc.

N - Number of independent variables of the function (dimension of X and G).

M - Number of independent elements of $\hat{H}$. Dimension of H; this must always be equal to N*(N+1)/2 and calculated in the calling programme.

IRIT - Output printing option, must have values 0, 1, 2, or 3. See output description.

EPSF - Desired absolute accuracy in function values of succeeding minimum estimates.

EPSX - Desired absolute accuracy in the independent variables vector.

EPSG - Desired absolute accuracy in the gradient norm.

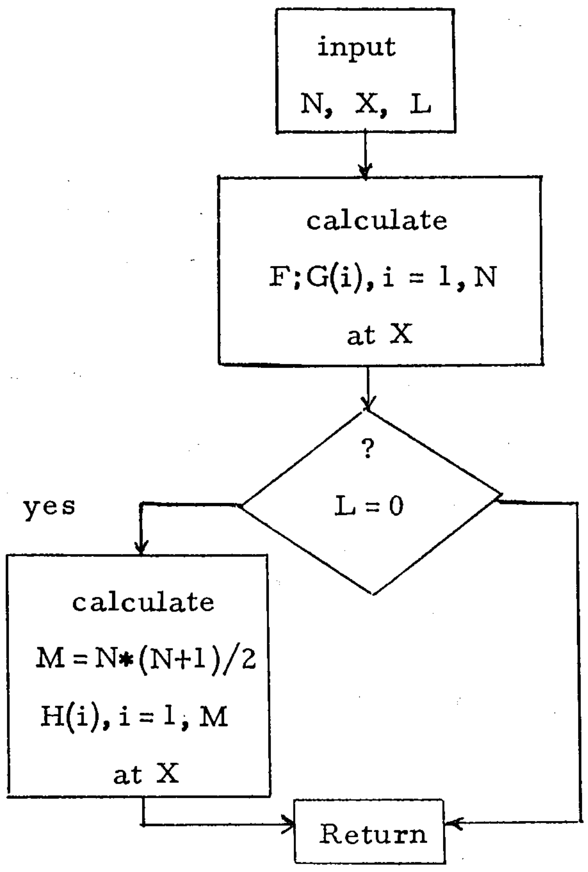
IMAX - Maximum number of iterations.

NIT - Maximum number of steepest descent method iterations.

E. Calling Sequence of "FUN"

SUBROUTINE NAME (N, X, F, G, H, L) where NAME is the name of the function assigned to FUN in MINIM and specified in the programme that calls MINIM. This subroutine must be made by the programmer; it should produce for a given X with dimension N: the function value F, the gradient G and the Hessian H. The matrix $\hat{H}$ must be stored in the same way as described in section D. Since the evaluation of $\hat{H}$ is the most time-consuming and not always required by the calling MINIM, a signal is transferred from MINIM to NAME. This is done by the switch L. If L = 0, $\hat{H}$ must be computed in NAME; if L $\neq$ 0, not.

NAME has thus to be constructed in the following way:



F. Output Given by MINIM

IRIT = 0	No output at all except error messages.
IRIT = 1	At every iteration a list is printed out of the essential data to control the flow of the procedure. This list contains:

1) EPSX, EPSF, EPSG, IMAX, NIT. These parameters are only printed as heading at every page.
2) EXTERNAL SUB ITR and INTERNAL SUB ITR are indications how λ_a and λ_b were found in minimizing $f(\vec{x}_o + \lambda\vec{s})$.
3) NEG-D. Number of negative values in $\hat{D}$-matrix. (If zero, $\hat{H}$ is positive definite).
4) F(OLD) and F(NEW) are respectively $f(\vec{x}_{old})$ and $f(\vec{x}_{new})$.

5) G$\ast$ S is $(\vec{g}(\vec{x}_{new}), \vec{s})$

S$\ast$H$\ast$S is $(\vec{s}, \hat{H}(\vec{x}_{new})\,\vec{s})$

G$\ast$H$\ast$G is $(\vec{g}(\vec{x}_{new}), \hat{H}\,\vec{g}(\vec{x}_{new}))$

GNØRM is $\left|\vec{g}(\vec{x}_{new})\right|$

SNØRM is $|\vec{s}|$

6) DAX is $\left|\vec{x}_{new} - \vec{x}_{old}\right| = \left|\Delta\vec{x}\right|$

7) D(LØW) is the most negative element in $\hat{D}$

D(SML) is the smallest absolute element in $\hat{D}$

8) LAMBDA is the minimizing λ.

IRIT = 2 In addition to the described output, a list is given of XOLD containing $\vec{x}_{old}$ and XNEW containing $\vec{x}_{new}$.

IRIT = 3 This option is mainly for testing purposes. It produces also a list of G, S and D containing respectively the elements of $g(\vec{x}_{new})$, $\vec{s}$ and the diagonal of $\hat{D}$.

An example of an output with IRIT = 2 may be found in Appendix 2.

G. Some Examples

1) Tests have been made with the function of R. Fletcher and M. J. D. Powell [4]:

$$f(\vec{\alpha}) = \sum_{i=1}^{n}\left[E_i - \sum_{j=1}^{n}(A_{ij}\sin\alpha_j + B_{ij}\cos\alpha_j)\right]^2 \qquad (17)$$

where the A_{ij}'s and the B_{ij}'s are fixed randomly between -100 and +100. Numbers α_j^M were then generated randomly between $-\pi$ and $+\pi$ after which the E_i's were computed by:

$$E_i = \sum_{j=1}^{n}(A_{ij}\sin\alpha_j^M + B_{ij}\cos\alpha_j^M) \qquad (18)$$

The minimum of $f(\vec{\alpha})$ lies at $\vec{\alpha}^M$ where the function is zero.

Starting with $\vec{\alpha} = \vec{\alpha}^M + 0.1\vec{\delta}$, where the numbers δ_i are randomly distributed between $-\pi$ and $+\pi$, the programme MINIM finds the minimum at $\vec{\alpha}^M$. This has been successfully tested for the cases n = 2, 5, 10, 40; the number of iterations required to get $\left|\vec{g}(\vec{\alpha})\right| < 10^{-8}$ being respectively 5, 7, 7, 16. This shows the main advantage of second order methods where the number of required iterations is almost independent of the number of variables. For an output example of MINIM one is referred to Appendix 2.

Other satisfactory tests have been made on the banana-shaped Rosenbrock's function [5]:

$$f(\vec{x}) = 100\,(x_2 - x_1^2)^2 + (1-x_1)^2 \tag{19}$$

Starting at $\vec{x} = (x_1, x_2) = (-1.2,\ 1)$. The minimum at point (1,1) is reached in 20 normal N-R iterations.

2) The function of C.F. Wood, cited in [1]:

$$f(\vec{x}) = 100\,(x_2 - x_1^2)^2 + (1-x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3^2) +$$

$$+ 10.1\left[(x_2-1)^2 + (x_1-1)^2\right] + 19.8(x_2-1)(x_4-1) \tag{20}$$

has been minimized starting with $\vec{x} = (x_1, x_2, x_3, x_4) = (-3,\ -1,\ -3,\ -1)$. This function shows the power of the modified N-R method where $\hat{H}$ is not "complete" positive definite. The results are summarized in Table I.

The same general behaviour for the convergence as described in [1] is observed. In the final steps the simple N-R method shows, however, a slower convergence than in [1] due to the special method used here to find the minimizing λ instead of imposing $\lambda = 1$ at each step. It must be said that for higher order functions with many variables, the estimate of the minimum in the $\vec{s}$-direction is often very bad by imposing $\lambda = 1$, especially far from the minimum neighbourhood, and that in these cases a new minimum estimate is found for which $\hat{H}$ is again not always positive definite.

3) This programme MINIM has been successfully used to minimize the potential energy of a system of interacting atoms in low symmetry crystal structures including a central point defect (e. g. a vacancy in monoclinic ZrO_2). The 70 nearest neighbours to this point defect were allowed to relax and MINIM had to minimize a function of 210 variables. The number of iterations to achieve physical zero gradient for this kind of problem lies between 30 and 40. For cubic structures the displacements are not so big as in low symmetry structures and usually the starting positions are at once in the convergence region of the usual N-R method. To have full relaxation of the atoms, only 4 or 5 iterations are needed. Techniques using only first order methods and relaxing all the atoms at a time showed to be unable to solve this kind of problem.

4) MINIM has also been used for curve fitting in the least squares sense. Curves could be fitted where the REEP-programme (using first order methods) failed. A general programme has been designed for this scope.

REFERENCES

[1] A. V. FIACCO and G. P. McCORMICK; "Nonlinear Programming: Sequential Unconstrained Minimization Techniques", J. Wiley & Sons, Inc., (1968)

[2] M. MARCUS; "Basic Theorems in Matrix Theory", Nat. Bur. of Standards. Appl. Math. Ser., 57 (1959)

[3] R. FLETCHER and C. M. REEVES; Comp. Journ. 7, 149 (1964)

[4] R. FLETCHER and M. J. D. POWELL; " A Rapidly Convergent Descent Method for Minimization", Comp. Journ. 6, 163 (1963)

[5] H. H. ROSENBROCK; "An Automatic Method for Finding the Greatest or Least Value of a Function", Comp. Journ. 3, 175 (1960)

Itera-tion	X_1	X_2	X_3	X_4	Method	Function Value
0	-3.0	-1.0	-3.0	-1.0	NR	19192.
1	-2.696	6.176	-2.663	5.864	NR	1291.4
2	-1.896	2.635	-1.874	2.564	NR	295.95
3	-1.524	2.004	-1.475	1.810	NR	67.895
4	-1.215	1.323	-1.183	1.252	NR	17.336
5	-1.065	1.103	-1.018	0.999	NR	8.689
6	-1.000	1.003	-0.957	0.921	NR	7.892
7	-0.996	1.003	-0.940	0.895	NR	7.876
8	-1.042	1.094	-0.891	0.804	MOD	7.874
9	-1.098	1.211	-0.825	0.686	NR	7.870
10	-1.154	1.338	-0.743	0.558	NR	7.854
11	-1.212	1.474	-0.648	0.421	NR	7.833
12	-1.273	1.627	-0.520	0.267	NR	7.780
13	-1.329	1.772	-0.364	0.121	NR	7.768
14	-1.361	1.862	-0.207	0.031	NR	7.749
15	-1.387	1.933	0.024	-0.039	NR	7.301
16	-1.366	1.874	0.151	0.020	NR	6.779
17	-1.331	1.780	0.447	0.115	NR	6.775
18	-1.262	1.597	0.545	0.300	NR	5.600
19	-1.158	1.336	0.789	0.567	NR	5.148
20	-1.034	1.063	0.929	0.853	NR	4.232
21	-0.880	0.751	1.101	1.180	NR	3.756
22	-0.723	0.504	1.206	1.449	NR	3.164
23	-0.515	0.226	1.322	1.738	NR	2.794
24	-0.398	0.149	1.348	1.818	NR	2.372
25	-0.231	0.024	1.393	1.940	NR	2.136
26	-0.088	-0.007	1.404	1.973	NR	1.775
27	0.157	-0.031	1.411	1.995	NR	1.602
28	0.240	0.056	1.382	1.911	NR	1.077
29	0.479	0.173	1.341	1.798	NR	0.9838
30	0.551	0.302	1.293	1.673	NR	0.4821
31	0.725	0.495	1.221	1.487	NR	0.3236
32	0.801	0.638	1.163	1.350	NR	0.1206
33	0.954	0.887	1.058	1.110	NR	0.738×10^{-1}
34	0.967	0.936	1.031	1.063	NR	0.374×10^{-2}
35	0.999	0.997	1.001	1.002	NR	0.182×10^{-3}
36	0.999	0.999	1.000	1.000	NR	0.473×10^{-7}
37	1.000	1.000	1.000	1.000	NR	0.340×10^{-13}
38	1.000	1.000	1.000	1.000	NR	0.371×10^{-26}

TABLE 1

APPENDIX 1 - LISTING OF MINIM.

```
FORTRAN IV G LEVEL  18                MINIM             DATE = 71103          11/23/10                PAGE 0001
 0001              SUBROUTINE MINIM (FUN,FM,X,G,H,N,M,IRIT,EPSF,EPSX,EPSG,IMAX,NIT)
             C
             C     PROGRAM CALCULATES MINIMUM OF FUNCTION FM=FM(X(I)) I=1,N
             C
             C     DESCRIPTION CALLING SEQUENCE OF MINIM
             C     FUN   NAME OF FUNCTION TO BE MINIMISED
             C     X(I)   VECTOR OF INDEPENDENT VARIABLES
             C     G(I)   VECTOR OF FIRST DERIVATIVES (GRADIENTS)
             C     H(J)      MATRIX ( IN VECTOR FORM ) OF SECOND DERIVATIVES (HESSIAN)
             C     N      NUMBER OF ELEMENTS X(I),G(I)
             C     M      NUMBER OF ELEMENTS H(J),MUST BEPUT EQUAL N*(N+1)/2
             C     IRIT=0   NO PRINTING IN MINIM
             C     IRIT=1   PRINTING OF ESSENTIAL DATA OF EACH ITERATION
             C     IRIT=2   XOLD AND XNEW ARE ALSO PRINTED
             C     IRIT=3   G,S AND D  VECTORS ARE ALSO PRINTED
             C     EPSF DESIRED ACCURACY IN FM (FUNCTION)
             C     EPSX DESIRED ACCURACY IN X(I) (INDEPENDENT VARIABLES)
             C     EPSG DESIRED ACCURACY IN G (NORM OF GRADIENT)
             C     IMAX DESIRED MAXIMUM NUMBER OF ITERATIONS
             C     NIT  ITERATION NUMBER AFTER WHICH GRADIENT METHOD IS AVOIDED
             C     WHEN EPSF,EPSX,EPSG,IMAX OR NIT ARE PUT ZERO,MINIM SET THESE VALUES
             C     ON    1.-8 1.-8 1.-8  40      2*N**(1/3)
             C
 0002                                                                    CONTINUE
             C     THE MAIN PROGRAM THAT CALLS MINIM MUST PROVIDE THE SUBROUTINE WITH
             C     THESE PARAMETERS AND MUST DEFINE THE DIMENSIONS OF X,G AND H
             C
             C     SPECIFICATION OF FUNCTION TO BE MINIMISED
             C     CALLING SUBROUTINE FUN(N,X,FM,G,H,L)
             C     FUN IS NAME
             C     N NUMBER OF VARIABLES                   DELIVERED BY MINIM
             C     X(I) VARIABLES                          DELIVERED BY MINIM
             C     FM FUNCTION                             CALCULATED IN FUN
             C     G(I) GRADIENTS                          CALCULATED IN FUN
             C     H(J) HESSIAN                            CALCULATED IN FUN
             C     L SWITCH DELIVERED BY MINIM
             C     FOR L=0 FUN MUST CALCULATE H(J)
             C     FOR L=1 FUN MUST NOT CALCULATE H(J)
             C     H(J) MATRIX IS STORED AS A VECTOR IN FOLLOWING MANNER,
             C     H(1) = H(1,1)
             C     H(2) = H(2,1)
             C     H(3) = H(2,2)
             C     H(4) = H(3,1)
             C     ETC
             C
             C
             C
 0003              IMPLICIT REAL*8 (A-H,O-Z)
```

```
FORTRAN IV G LEVEL  18                    MINIM              DATE = 71103           11/23/10                PAGE 0002

            DIMENSION D(200),G1(200),S(200),XOLD(200),A(200)
            DIMENSION ALFA(4),BETA(3),X(N),G(N),H(M)
            DATA ALFA /8HMODIFIED,8H NEWTON ,8HRAPHSON ,8HGRADIENT/
            DATA BLANK /8H        /
      C
      C     INITIALIZING,START VALUE,LENGTH OF GRADIENT
      C
            FPSF=EPSF
            FPSX=EPSX
            FPSG=EPSG
            IMAC=IMAX
            NYT=NIT
            IF (FPSX.LE.0.0D0) FPSX=1.0D-8
            IF (FPSF.LE.0.0D0) FPSF=1.0D-8
            IF (FPSG.LE.0.0D0) FPSG=1.0D-8
            IF (IMAC.LE.0) IMAC=40
            FN=N
            IF (NYT.LE.0) NYT=2.0*FN**0.333333
      C
            CALL FUN(N,X,FM,G,H,0)
            GG=0.0D0
            DO 12 I=1,N
         12 GG=GG+G(I)**2
            GNORM=DSQRT(GG)
            ITR=0
      C
      C     BEGIN NEW ITERATION
      C
         14 ITR=ITR+1
            I5=1
            I6=1
      C
      C     SAFE X AND COMPUTE GHG
      C
            FL=FM
            GHG=0.0D0
            II=0
            DO 22 I=1,N
            XOLD(I)=X(I)
            HGI=0.0D0
            II=II+I-1
            JJ=II
            DO 20 J=1,N
            IF (J-I)16,16,18
         16 IJ=II+J
            GO TO 20
         18 JJ=JJ+J-1
            IJ=JJ+I
```

APPENDIX 1 (continued)

```
FORTRAN IV G LEVEL  18                MINIM              DATE = 71103          11/23/10            PAGE 0003
   20 HGI=HGI+H(IJ)*G(J)
   22 GHG=GHG+G(I)*HGI
C
C     DEVELOP H IN L*D*LT,STORE L IN H
C     SEARCH FOR LOWEST AND SMALLEST ELEMENT OF D
C
      II=0
      DO 34 I=1,N
      IME=I-1
      JJ=II
      II=II+IME
      DO 32 J=I,N
      JJ=JJ+J-1
      IJ=JJ+I
      AIJ=0.0D0
      IF (I.EQ.1) GO TO 26
      DO 24 K=1,IME
      KI=II+K
      KJ=JJ+K
   24 AIJ=AIJ+H(KI)*H(KJ)*D(K)
   26 IF (J-I)32,28,30
   28 D(I)=H(IJ)-AIJ
      H(IJ)=1.0D0
      GO TO 32
   30 H(IJ)=(H(IJ)-AIJ)/D(I)
   32 CONTINUE
   34 CONTINUE
C
      ILW=1
      ISM=1
      DO 36 I=2,N
      IF (D(I).LT.D(ILW)) ILW=I
      IF (DABS(D(I)).LT.DABS(D(ISM))) ISM=I
   36 CONTINUE
C
C     DETERMINE WHETHER H IS POSITIVE DEFINITE OR NOT
C     DEFINE VECTOR A,SEARCH NUMBER OF NEGATIVE ELEMENTS IN D
C
      NNEG=0
      DO 38 I=1,N
      A(I)=0.0D0
      IF (D(I).GT.0.0D0) GO TO 38
      NNEG=NNEG+1
      A(I)=1.0D0
   38 CONTINUE
      I7=0
      IF (NNEG.GT.0) I7=1
C
```

APPENDIX 1 (continued)

```
FORTRAN IV G LEVEL  18                 MINIM              DATE = 71103          11/23/10              PAGE 0004
C
C      WHEN HESSLAN NOT POS-DEF USE STEEPEST DESCENT METHOD FOR
C      FIRST 2*N**(1/3) ITERATIONS
C      THE S-DIRECTION IS GIVEN BY    S=-G
C
       IF (I7.EQ.0.OR.ITR.GT.NYT) GO TO 44
       DO 40 I=1,N
  40   S(I)=-G(I)
       I6=0
       GO TO 70
C
C      INVERT LOWER TRIANGULAR MATRIX L AND STORE IN H
C
  44   JJ=0
       DO 48 J=2,N
       JME=J-1
       JJ=JJ+JME
       II=0
       DO 48 I=1,JME
       IME=I+1
       II=II+I-1
       IJ=JJ+I
       AIJ=0.0D0
       IF ((J-I).EQ.1) GO TO 48
       KK=II
       DO 46 K=IME,JME
       KK=KK+K-1
       IK=KK+I
       KJ=JJ+K
  46   AIJ=AIJ+H(IK)*H(KJ)
  48   H(IJ)=-H(IJ)-AIJ
C
C      IF H NOT POS-DEF DETERMINE S-DIRECTION SATISFYING
C      THE EQUATION    LT.S=A
C
       IF (I7.EQ.0) GO TO 56
       I5=1
       II=0
       DO 54 I=1,N
       AI=0.0D0
       II=II+I-I
       IF (I.EQ.N) GO TO 54
       IPE=I+1
       JJ=II
       DO 52 J=IPE.N
       JJ=JJ+J-1
       IJ=JJ+I
  52   AI=AI+H(IJ)*A(J)
  54   S(I)=A(I)+AI
```

```
FORTRAN IV G LEVEL  18                MINIM             DATE = 71103           11/23/10            PAGE 0005
            GO TO 70
C
C           IF H IS POS-DEF COMPUTE INVERTED HESSIAN AND STORE IN H
C           H(-1)=L(-1)*D*LT(-1)
C
       56   I5=0
            II=0
            DO 60 I=1,N
            JJ=II
            II=II+I-1
            DO 60 J=I,N
            JJ=JJ+J-1
            IJ=JJ+I
            AIJ=0.0D0
            IF (J.EQ.N) GO TO 60
            JPE=J+1
            KK=JJ
            DO 58 K=JPE,N
            KK=KK+K-1
            IK=KK+I
            JK=KK+J
       58   AIJ=AIJ+H(IK)*H(JK)/D(K)
       60   H(IJ)=H(IJ)/D(J)+AIJ
C
C           DETERMINE S-DIRECTION SATISFYING THE EQUATION  H*S=-G
C
            II=0
            DO 66 I=1,N
            S(I)=0.0D0
            II=II+I-1
            JJ=II
            DO 66 J=1,N
            IF (J-I)62,62,64
       62   IJ=II+J
            GO TO 66
       64   JJ=JJ+J-1
            IJ=JJ+I
       66   S(I)=S(I)-H(IJ)*G(J)
C
C           COMPUTE GS,SHS,NORM OF S
C
       70   GS=0.0D0
            SS=0.0D0
            SHS=0.0D0
            DO 72 I=1,N
            IF (D(I).LT.0.0D0) SHS=SHS+D(I)
            GS=GS+G(I)*S(I)
       72   SS=SS+S(I)**2
```

APPENDIX I (continued)

```
FORTRAN IV G LEVEL  18              MINIM              DATE = 71103           11/23/10            PAGE 0006
            SNORM=DSQRT(SS)
C
            IF (I5.EQ.0) SHS=-GS
            IF (I6.EQ.0) SHS=GHG
            IF (GS.LT.0.0D0) GO TO 80
            DO 74 I=1,N
   74       S(I)=-S(I)
            GS=-GS
C
C
C           APPROXIMATE NEW FUNCTION MINIMUM IN THE S-DIRECTION
C           LAMBDA IS DETERMINED BY DAVIDON METHOD
C
   80       K1=0
            K2=0
            YA=0.0D0
            FA=FL
            GSA=GS
            YB=-GS/DABS(SHS)
            IF (I5.NE.0) YB=DMIN1(YB,1.0D0/SNORM)
            YM=YB
   82       DO 84 I=1,N
   84       X(I)=XOLD(I)+YB*S(I)
            K1=K1+1
   86       CALL FUN(N,X,FB,G1,H,I7)
            FM=FB
            GSB=0.0D0
            DO 88 I=1,N
   88       GSB=GSB+G1(I)*S(I)
            IF (FB.GT.FA+FPSF) GO TO 90
            IF (I5.EQ.0) GO TO 110
            IF (GSB.GT.0.0D0) GO TO 90
            YA=YB
            FA=FB
            GSA=GSB
            YB=YB+YB
            GO TO 82
C
   90       IF (YA.EQ.YB)    WRITE (6,330)
            Z=3.0D0*(FA-FB)/(YB-YA)+GSA+GSB
            W=DSQRT(Z*Z-GSA*GSB)
            YM=YB-(GSB+W-Z)*(YB-YA)/(GSB-GSA+W+W)
            YMS=YA+(YB-YA)*0.25D0
            ISAF=0
            IF (YM.LT.YMS) ISAF=1
            IF (ISAF.EQ.1) YM=YMS
            DO 92 I=1,N
   92       X(I)=XOLD(I)+YM*S(I)
```

```
FORTRAN IV G LEVEL  18                MINIM               DATE = 71103          11/23/10            PAGE 0007
      K2=K2+1
      CALL FUN(N,X,FM,G1,H,0)
      GIS=0.0D0
      DO 94 I=1,N
   94 GIS=GIS+G1(I)*S(I)
C
C     TEST ON CORRECT LAMBDA
C
      IF (ISAF.EQ.1) GO TO 110
      IF (FM-FA-FPSF) 98,98,96
   96 YB=YM
      FB=FM
      GSB=GIS
      GO TO 90
   98 IF (FM-FB-FPSF) 110,110,100
  100 YA=YM
      FA=FM
      GSA=GIS
      GO TO 90
C
  110 YL=YM
      DAX=YL*SNORM
      GG=0.0D0
      DO 112 I=1,N
      G(I)=G1(I)
  112 GG=GG+G(I)**2
      GNORM=DSQRT(GG)
C
C     PRINTING
C
      IF (IRIT.EQ.0) GO TO 116
      IPAG=50
      NVF=(N+4)/5
      IBL=7
      IF (IRIT.EQ.2) IBL=8+NVF*2
      IF (IRIT.EQ.3) IBL=9+NVF*5
      ILN=MAX0 (1,(IPAG-2)/IBL)
      IF (MOD(ITR,ILN).NE.1.AND.ILN.NE.1)  GO TO 114
      WRITE (6,320) FPSX,FPSF,FPSG,IMAC,NYT
  114 BETA(1)=BLANK
      BETA(2)=BLANK
      BETA(3)=ALFA(4)
      IF (I6.EQ.0) GO TO 118
      BETA(2)=ALFA(2)
      BETA(3)=ALFA(3)
      IF (I7.EQ.1)  BETA(1)=ALFA(1)
  118 WRITE (6,302) ITR,BETA,K1,K2,NNEG,I6,I7
      WRITE (6,304) FL,GS,SHS,GHG
      WRITE (6,306) FM,GNORM,SNORM,DAX
      WRITE (6,308) D(ILW),D(ISM),YL
      IF (IRIT.EQ.1) GO TO 116
      WRITE (6,310) (XOLD(I),I=1,N)
      WRITE (6,312) (X(I),I=1,N)
      IF (IRIT.EQ.2) GO TO 116
      WRITE (6,314) (G(I),I=1,N)
      WRITE (6,316) (S(I),I=1,N)
      WRITE (6,318) (D(I),I=1,N)
  116 CONTINUE
C
C     TEST QUALITY OF ITERATION
C
      IF (ITR.GT.IMAC) GO TO 120
      IF (DAX.LT.FPSX) GO TO 120
      IF (DABS(FL-FM).LT.FPSF) GO TO 120
      IF (GNORM.GT.FPSG) GO TO 14
  120 RETURN
C
  302 FORMAT (1H0/1H ,10HITERATION=I3,2X3A8,8H METHOD ,6X17HEXTERNAL SUB
     1 ITR=I2,4X17HINTERNAL SUB ITR=I2,5X6HNEG-D=I3,10X2I2)
  304 FORMAT (1H0,8H F(OLD)=D22.15,5X4HG*S=D22.15,4X6HS*H*S=D22.15,3X6HG
     1*H*G=D22.15)
  306 FORMAT (1H ,8H F(NEW)=D22.15,3X6HGNORM=D22.15,4X6HSNORM=D22.15,4X5
     1HDAX =D22.15)
  308 FORMAT (1H ,8H D(LOW)=D22.15,2X7HD(SML)=D22.15,3X7HLAMBDA=D22.15)
  310 FORMAT (1H0,8HXOLD(I)=D17.10,4D20.10/(D26.10,4D20.10))
  312 FORMAT (1H ,8HXNEW(I)=D17.10,4D20.10/(D26.10,4D20.10))
  314 FORMAT (1H0,6X2HG=D17.10,4D20.10/(D26.10,4D20.10))
  316 FORMAT (1H ,6X2HS=D17.10,4D20.10/(D26.10,4D20.10))
  318 FORMAT (1H ,6X2HD=D17.10,4D20.10/(D26.10,4D20.10))
  320 FORMAT (1H1,5HMINIM,9X5HEPSX=D11.4,5X5HEPSF=D11.4,5X5HEPSG=D11.4,
     15X5HIMAX=I3,6X4HNIT=I2,9X5HMINIM)
  330 FORMAT (1H0,45HERROR MESSAGE FROM MINIM,CHECK SUBROUTINE FUN)
      END
```

APPENDIX 2 - OUT PUT OF MINIM.

```
MINIM          EPSX= 0.1000D-07     EPSF= 0.1000D-07     EPSG= 0.1000D-07     IMAX= 40      NIT= 4          MINIM

ITERATION=  1                     GRADIENT METHOD        EXTERNAL SUB ITR= 2    INTERNAL SUB ITR= 1     NEG-D=  1              0 1

 F(OLD)= 0.920131191039437D 04      G*S=-0.295975336245618D 10     S*H*S= 0.540162401805974D 15    G*H*G= 0.540162401805974D 15
 F(NEW)= 0.109562119078805D 04    GNORM= 0.150606797100281D 05     SNORM= 0.544036153436165D 05      DAX = 0.313002806285515D 00
 D(LOW)=-0.879308391565555D 05   D(SML)= 0.852739872359131D 04    LAMBDA= 0.575334569786530D-05

XOLD(I)=-0.2715732694D 01     -0.9384514093D 00      0.2133195102D 01     -0.1194156587D 01      0.2133292198D 01
         0.1306303084D 01     -0.1228957713D 01     -0.4710443020D 00      0.2868844509D 01      0.7667501569D 00
XNEW(I)=-0.2807152166D 01     -0.9885407119D 00      0.2177076081D 01     -0.1170148935D 01      0.2107523608D 01
         0.1238246353D 01     -0.1055851138D 01     -0.3944265102D 00      0.2710969926D 01      0.9029342088D 00

ITERATION=  2              NEWTON RAPHSON  METHOD        EXTERNAL SUB ITR= 1    INTERNAL SUB ITR= 1     NEG-D=  0              1 0

 F(OLD)= 0.109562119078805D 04      G*S=-0.260247923083261D 04     S*H*S= 0.260247923083261D 04    G*H*G= 0.418438315900140D 14
 F(NEW)= 0.399313174511913D 03    GNORM= 0.881237364611571D 04     SNORM= 0.119517270875289D 01      DAX = 0.525586943858797D 00
 D(LOW)= 0.174868138299792D 04   D(SML)= 0.174868138299792D 04    LAMBDA= 0.439758153787853D 00

XOLD(I)=-0.2807152166D 01     -0.9885407119D 00      0.2177076081D 01     -0.1170148935D 01      0.2107523608D 01
         0.1238246353D 01     -0.1055851138D 01     -0.3944265102D 00      0.2710969926D 01      0.9029342088D 00
XNEW(I)=-0.2788243441D 01     -0.6922301156D 00      0.2292366921D 01     -0.9958197948D 00      0.2022642688D 01
         0.1085200106D 01     -0.1051593344D 01     -0.2682476221D 00      0.2956040033D 01      0.1097292067D 01

ITERATION=  3              NEWTON RAPHSON  METHOD        EXTERNAL SUB ITR= 1    INTERNAL SUB ITR= 0     NEG-D=  0              1 0

 F(OLD)= 0.399313174511913D 03      G*S=-0.781138429391217D 03     S*H*S= 0.781138429391217D 03    G*H*G= 0.114976914521688D 14
 F(NEW)= 0.252879981555465D 01    GNORM= 0.403942286560755D 03     SNORM= 0.190428169388600D 00      DAX = 0.190428169388600D 00
 D(LOW)= 0.943354994915657D 04   D(SML)= 0.943354994915657D 04    LAMBDA= 0.100000000000000D 01

XOLD(I)=-0.2788243441D 01     -0.6922301156D 00      0.2292366921D 01     -0.9958197948D 00      0.2022642688D 01
         0.1085200106D 01     -0.1051593344D 01     -0.2682476221D 00      0.2956040033D 01      0.1097292067D 01
XNEW(I)=-0.2741130049D 01     -0.7017278200D 00      0.2263506895D 01     -0.1115887654D 01      0.2030424007D 01
         0.1043178078D 01     -0.9561656945D 00     -0.2508367204D 00      0.2947238683D 01      0.1033165265D 01

ITERATION=  4              NEWTON RAPHSON  METHOD        EXTERNAL SUB ITR= 1    INTERNAL SUB ITR= 0     NEG-D=  0              1 0

 F(OLD)= 0.252879981555465D 01      G*S=-0.503703825660757D 01     S*H*S= 0.503703825660757D 01    G*H*G= 0.198209466545032D 11
 F(NEW)= 0.978967181188882D-02    GNORM= 0.368066211932719D 02     SNORM= 0.494066585828914D-01      DAX = 0.494066585828914D-01
 D(LOW)= 0.273596573683033D 04   D(SML)= 0.273596573683033D 04    LAMBDA= 0.100000000000000D 01

XOLD(I)=-0.2741130049D 01     -0.7017278200D 00      0.2263506895D 01     -0.1115887654D 01      0.2030424007D 01
         0.1043178078D 01     -0.9561656945D 00     -0.2508367204D 00      0.2947238683D 01      0.1033165265D 01
XNEW(I)=-0.2728281934D 01     -0.7284499439D 00      0.2247938177D 01     -0.1136991097D 01      0.2070361471D 01
         0.1049740685D 01     -0.9488668812D 00     -0.2497115542D 00      0.2928187643D 01      0.1015492450D 01
```

APPENDIX 2 (Continued)

```
MINIM          EPSX= 0.1000D-07     EPSF= 0.1000D-07     EPSG= 0.1000D-07     IMAX= 40      NIT= 4          MINIM

ITERATION=  5            NEWTON RAPHSON  METHOD        EXTERNAL SUB ITR= 1    INTERNAL SUB ITR= 0     NEG-D=  0            1 0

 F(OLD)= 0.978967181188882D-02     G*S=-0.195695043204457D-01     S*H*S= 0.195695043204457D-01    G*H*G= 0.154341022788877D 09
 F(NEW)= 0.693849414753930D-07   GNORM= 0.228392555222613D-01     SNORM= 0.124480113488103D-02     DAX = 0.124480113488103D-02
 D(LOW)= 0.241165026931730D 04  D(SML)= 0.241165026931730D 04    LAMBDA= 0.100000000000000D 01

XOLD(I)=-0.2728281934D 01    -0.7284499439D 00     0.2247938177D 01    -0.1136991097D 01     0.2070361471D 01
         0.1049740685D 01    -0.9488668812D 00    -0.2497115542D 00     0.2928187643D 01     0.1015492450D 01
XNEW(I)=-0.2728295441D 01    -0.7288705718D 00     0.2247657425D 01    -0.1137542904D 01     0.2070822748D 01
         0.1049997893D 01    -0.9485775620D 00    -0.2499698076D 00     0.2927801648D 01     0.1014851527D 01

ITERATION=  6            NEWTON RAPHSON  METHOD        EXTERNAL SUB ITR= 1    INTERNAL SUB ITR= 0     NEG-D=  0            1 0

 F(OLD)= 0.693849414753930D-07     G*S=-0.138767543141909D-06     S*H*S= 0.138767543141909D-06    G*H*G= 0.508783936389675D 02
 F(NEW)= 0.107670887974281D-15   GNORM= 0.376598942272516D-05     SNORM= 0.155618471501742D-04     DAX = 0.155618471501742D-04
 D(LOW)= 0.235572063965228D 04  D(SML)= 0.235572063965228D 04    LAMBDA= 0.100000000000000D 01

XOLD(I)=-0.2728295441D 01    -0.7288705718D 00     0.2247657425D 01    -0.1137542904D 01     0.2070822748D 01
         0.1049997893D 01    -0.9485775620D 00    -0.2499698076D 00     0.2927801648D 01     0.1014851527D 01
XNEW(I)=-0.2728291512D 01    -0.7288799284D 00     0.2247653961D 01    -0.1137546539D 01     0.2070822716D 01
         0.1049996376D 01    -0.9485731126D 00    -0.2499752043D 00     0.2927797318D 01     0.1014844895D 01

ITERATION=  7            NEWTON RAPHSON  METHOD        EXTERNAL SUB ITR= 1    INTERNAL SUB ITR= 0     NEG-D=  0            1 0

 F(OLD)= 0.107670887974281D-15     G*S=-0.215341775787039D-15     S*H*S= 0.215341775787039D-15    G*P*G= 0.183931047166054D-05
 F(NEW)= 0.829635153259623D-26   GNORM= 0.415991148045312D-10     SNORM= 0.370052427931621D-09     DAX = 0.370052427931621D-09
 D(LOW)= 0.235565139601204D 04  D(SML)= 0.235565139601204D 04    LAMBDA= 0.100000000000000D 01

XOLD(I)=-0.2728291512D 01    -0.7288799284D 00     0.2247653961D 01    -0.1137546539D 01     0.2070822716D 01
         0.1049996376D 01    -0.9485731126D 00    -0.2499752043D 00     0.2927797318D 01     0.1014844895D 01
XNEW(I)=-0.2728291512D 01    -0.7288799286D 00     0.2247653961D 01    -0.1137546539D 01     0.2070822716D 01
         0.1049996376D 01    -0.9485731125D 00    -0.2499752045D 00     0.2927797318D 01     0.1014844894D 01
```

```
FORTRAN IV G LEVEL  18                   MAIN                 DATE = 71033          16/01/05            PAGE 0001

              IMPLICIT REAL*8(A-H,X)
              EXTERNAL FCOS
              COMMON A(40,40),B(40,40),E(40)
              DIMENSION X(40),G(40),H(2000)
          400 READ (5,100) N,K,Z
              CALL SETRND(Z)
              DO 4 I=1,N
              DO 2 J=1,N
              A(I,J)=RANDOM(-1.0E2,1.0E2)
            2 B(I,J)=RANDOM(-1.0E2,1.0E2)
            4 X(I)=RANDOM(-3.1415E0,3.1415E0)
        C
              DO 6 I=1,N
              E(I)=0.0D0
              DO 8 J=1,N
            8 E(I)=E(I)+A(I,J)*DSIN(X(J))+B(I,J)*DCOS(X(J))
            6 CONTINUE
        C
              WRITE (6,102) (X(I),I=1,N)
              WRITE (6,108) (E(I),I=1,N)
              DO 20 I=1,N
           20 X(I)=X(I)+RANDOM(-3.14E-1,3.14E-1)
              M=(N*(N+1))/2
              CALL MINIM (FCOS,FM,X,G,H,N,M,K,0.0D0,0.0D0,0.0D0,0,0)
              GO TO 400
          100 FORMAT(2I2,F6.0)
          102 FORMAT (1H1,12F10.3/(1H0,12F10.3))
          108 FORMAT (1H0,12F10.3/(1H0,12F10.3))
              END
```

APPENDIX 2 (Continued)

```
FORTRAN IV G LEVEL  18                  FCOS                DATE = 71033        16/01/05              PAGE 0001

            SUBROUTINE FCOS(N,X,FM,G,H,L)
            IMPLICIT REAL*8 (A-H,O-Z)
            DIMENSION X(1),G(1),H(1),D(40)
            DATA T/2.0D0/
            COMMON A(40,40),B(40,40),E(40)
      C
      C     FLETCHER AND POWELL MULTI-VARIABLE COS-SIN FUNCTION
      C
            DO 4 I=1,N
            D(I)=-E(I)
            DO 4 J=1,N
          4 D(I)=D(I)+A(I,J)*DSIN(X(J))+B(I,J)*DCOS(X(J))
            FS=0.0D0
            DO 6 I=1,N
          6 FS=FS+D(I)**2
            FM=FS
      C
      C     GRADIENTS
      C
            DO 10 K=1,N
            G(K)=0.0D0
            DO 10 I=1,N
         10 G(K)=G(K)+T*D(I)*(A(I,K)*DCOS(X(K))-B(I,K)*DSIN(X(K)))
      C
      C     HESSIAN ONLY IF L=0
      C
            IF (L.NE.0) GO TO 80
            IN=0
            DO 20 I=1,N
            DO 20 J=1,I
            IN=IN+1
            S=0.0D0
            IF (I.EQ.J) GO TO 14
            DO 12 K=1,N
         12 S=S+(A(K,I)*DCOS(X(I))-B(K,I)*DSIN(X(I)))*(A(K,J)*DCOS(X(J))-B(K,J
           1)*DSIN(X(J)))
            GO TO 18
         14 DO 16 K=1,N
         16 S=S-D(K)*(A(K,I)*DSIN(X(I))+B(K,I)*DCOS(X(I)))+(A(K,I)*DCOS(X(I))-
           1B(K,I)*DSIN(X(I)))**2
         18 H(IN)=T*S
         20 CONTINUE
         80 RETURN
            END
```

APPENDIX 2 (Continued)

```
FORTRAN IV G LEVEL  18              SETRND          DATE = 71033        16/01/05              PAGE 0001

            SUBROUTINE SETRND(ARG)
C           G.GAGGERO 20-8-68 ONLY FO- SOSTEM /360
            INTEGER IARG/221/
            IARG=ARG+0.5
            IF(MOD(IARG,2).EQ.0) IARG=IARG+1
            RETURN
C
            ENTRY RANDOM(ARG1,ARG2)
            VAL=ARG2-ARG1
            IARG=IARG*65539
            IF(IARG)5,6,6
          5 IARG=IARG+2147483647+1
          6 RANDOM=IARG*0.4656613E-9*VAL+ARG1
            RETURN
            END
```

NOTICE TO THE READER

All scientific and technical reports published by the Commission of the European Communities are announced in the monthly periodical "**euro-abstracts**". For subscription (1 year : B.Fr. 1025) or free specimen copies please write to :

Sales Office for Official Publications
of the European Communities

P.O. Box 1003

Luxembourg 1
(Grand-Duchy of Luxembourg)

To disseminate knowledge is to disseminate prosperity — I mean general prosperity and not individual riches — and with prosperity disappears the greater part of the evil which is our heritage from darker times.

Alfred Nobel

SALES OFFICES

All reports published by the Commission of the European Communities are on sale at the offices listed below, at the prices given on the back of the front cover. When ordering, specify clearly the EUR number and the title of the report which are shown on the front cover.

OFFICE FOR OFFICIAL PUBLICATIONS OF THE EUROPEAN COMMUNITIES
P.O. Box 1003 - Luxembourg 1
(Compte chèque postal N° 191-90)

BELGIQUE — BELGIË
MONITEUR BELGE
Rue de Louvain, 40-42 - B-1000 Bruxelles
BELGISCH STAATSBLAD
Leuvenseweg 40-42 - B-1000 Brussel

DEUTSCHLAND
VERLAG BUNDESANZEIGER
Postfach 108 006 - D-5 Köln 1

FRANCE
SERVICE DE VENTE EN FRANCE DES PUBLICATIONS DES COMMUNAUTÉS EUROPÉENNES
rue Desaix, 26 - F-75 Paris 15e

ITALIA
LIBRERIA DELLO STATO
Piazza G. Verdi, 10 - I-00198 Roma

LUXEMBOURG
OFFICE DES PUBLICATIONS OFFICIELLES DES COMMUNAUTÉS EUROPÉENNES
Case Postale 1003 - Luxembourg 1

NEDERLAND
STAATSDRUKKERIJ en UITGEVERIJBEDRIJF
Christoffel Plantijnstraat - Den Haag

UNITED KINGDOM
H. M. STATIONERY OFFICE
P.O. Box 569 - London S.E.1

Commission of the European Communities
D.G. XIII - C.I.D.
29, rue Aldringen
Luxembourg

CDNA04777ENC