J. Austral. Math. Soc. 19 (Series B), (1976), 343–357.

NONLINEAR LEAST SQUARES — THE LEVENBERG ALGORITHM REVISITED*

M. R. OSBORNE

(Received 7 March 1975)

(Revised 28 July 1975)

Abstract

One of the most successful algorithms for nonlinear least squares calculations is that associated with the names of Levenberg, Marquardt, and Morrison. This algorithm gives a method which depends nonlinearly on a parameter γ for computing the correction to the current point. In this paper an attempt is made to give a rule for choosing γ which (a) permits a satisfactory convergence theorem to be proved, and (b) is capable of satisfactory computer implementation. It is believed that the stated aims have been met with reasonable success. The convergence theorem is both simple and global in character, and a computer code is given which appears to be at least competitive with existing alternatives.

1. Introduction

In this paper we consider a class of descent methods for minimising the sum of squares

$$F(\boldsymbol{x}) = \|\boldsymbol{f}(\boldsymbol{x})\|^2 = \boldsymbol{f}^T(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}) \tag{1.1}$$

where $\boldsymbol{f}$ is an n vector of, in general nonlinear, suitably smooth functions f_i of the independent variables $x_1, x_2, \cdots, x_p$ with $p < n$. The descent vectors $\boldsymbol{h}_i(\gamma)$, $i = 1, 2, \cdots$ are chosen by solving at each step the linear least squares problem minimise $\|\boldsymbol{r}_i(\gamma)\|^2$ where

$$\begin{bmatrix} \boldsymbol{f}(\boldsymbol{x}_i) \\ 0 \end{bmatrix} + \begin{bmatrix} A_i \\ \gamma B_i \end{bmatrix} \boldsymbol{h}_i(\gamma) = \boldsymbol{r}_i(\gamma), \tag{1.2}$$

* Professor Levenberg was killed recently in a motor car accident. This paper is respectfully dedicated to his memory.

$A_i = \nabla_x \mathbf{f}(\mathbf{x}_i)$, B_i is a $p \times p$ matrix which is assumed to have full rank (it will be no restriction to assume $\| B_i \| = 1$, and $\| B_i^{-1} \| \leqq \alpha$ where the spectral norm is used), and $\gamma \geqq 0$ is a scale parameter which controls the magnitude and direction of $\boldsymbol{h}_i$.

Remark (i)

The case $B_i = I$ together with a suitable strategy for choosing γ gives the algorithms of Levenberg [4], Marquardt [5], and Morrison [6]. The use of $B_i \neq I$ permits, for example, taking some account of the relative sizes of the columns of A_i.

Remark (ii)

Equation (1.2) uniquely determines $\boldsymbol{h}_i(\gamma)$ unless $\gamma = 0$ and A_i has less than full rank. In this case we determine $\boldsymbol{h}_i$ by taking the unique solution of (1.2) of minimum norm (that is we use the generalised inverse solution to (1.2)).

Discussion of the convergence of the descent process requires the specification of the rule for obtaining $\boldsymbol{x}_{i+1}$ from $\boldsymbol{x}_i$. This has been done for various line search strategies in Osborne [7], but the correspondence between this kind of discussion and actual algorithmic implementations is not as close as might be desired. In this paper an attempt is made to close this gap. Thus an algorithm is given which introduces no auxiliary line search parameter so that all adjustment of step length is done by altering γ, and the actual test for accepting the predicted step requires only quantities that are readily available at the current stage. It is, in fact, only a minor modification of the frequently used test that $F(\boldsymbol{x})$ be reduced at each stage (without qualification this is known to be unsatisfactory). A bonus is that, apart from one explicitly testable exception, the algorithm is *globally convergent* from an arbitrary starting point to a stationary point (either a minimum or possibly a saddle point) of F.

2. The algorithm

The development of the algorithm will be given in three stages. First a characterisation of the stationary points of $F(\boldsymbol{x})$ is recalled. Then a convergence theorem is given under the assumption that the sequence of values $\{F(\boldsymbol{x}_i)\}$ satisfy a suitable criterion. Finally it is shown that for fixed $\boldsymbol{x}_i$ this criterion can always be satisfied by choosing γ large enough in (1.2).

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

LEMMA 2.1. [7] If γ is bounded in (1.2) then the following are equivalent

(i) $\begin{bmatrix} f_i \\ 0 \end{bmatrix} = r_i$,

(ii) $\| f_i \| = \| r_i \|$,

and

(iii) x_i is a stationary point of $F(x)$.

THEOREM 2.1. If in (1.2) a bounded sequence $\{\gamma_i\}$ can be chosen such that

$$0 < \sigma \leqq \psi(x_i, \gamma_i) \tag{2.1}$$

where σ is a prescribed constant independent of i,

$$x_{i+1} = x_i + h_i(\gamma_i), \tag{2.2}$$

and

$$\psi(x_i, \gamma_i) = \frac{F(x_i) - F(x_{i+1})}{2(F(x_i) - \| r_i \|^2)}, \tag{2.3}$$

then the sequence $\{F(x_i)\}$ is convergent, and the limit points of the sequence $\{x_i\}$ are stationary points of $F(x)$.

PROOF. Note that unless x_i is a stationary point of $F(x)$ then (1.2) implies $\| r_i \| < \| f_i \|$, and in this case (2.1) implies $F(x_{i+1}) < F(x_i)$. Now (2.1) can be rewritten

$$\begin{aligned} \sigma &\leqq \frac{(\| f(x_i) \| + \| f(x_{i+1} \|)(\| f(x_i) \| - \| f(x_{i-1}) \|)}{2(\| f(x_i) \| + \| r_i \|)(\| f(x_i) \| - \| r_i \|)} \\ &\leqq \frac{\| f(x_i) \| - \| f(x_{i+1}) \|}{\| f(x_i) \| - \| r_i \|} \end{aligned} \tag{2.4}$$

as $\| f(x_i) \| \leqq \| f(x_i) \| + \| r_i \|$, and $\| f(x_i) \| + \| f(x_{i+1}) \| \leqq 2 \| f(x_i) \|$. Thus, rearranging (2.4),

$$\| f(x_{i+1}) \| \leqq \| f(x_i) \| - \sigma(\| f(x_i) \| - \| r_i \|) \tag{2.5}$$

so that the non-negative sequence $\{\| f(x_i) \|\}$ is convergent. Rearranging again gives

$$\| f(x_i) \| - \| r_i \| \leqq \frac{1}{\sigma} (\| f(x_i) \| - \| f(x_{i+1}) \|). \tag{2.6}$$

An appeal to Lemma 2.1 establishes the second part of the theorem as the right hand side tends to zero.

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

THEOREM 2.2. For each $\sigma < 1$, and each x_i, there exists a γ_i such that (2.1) is satisfied.

PROOF. It will be shown that

$$\psi(x_i, \gamma) \to 1 + 0(1/\gamma^2) \quad \text{as} \quad \gamma \to \infty$$

so that the test can be satisfied for any fixed x_i by choosing γ large enough. First note that the normal equations determining $h_i(\gamma)$ can be written

$$[A_i^T A_i + \gamma^2 B_i^T B_i] h_i(\gamma) = -A_i^T f_i \tag{2.7}$$

so that, as B_i has full rank,

$$h_i(\gamma) = -\frac{1}{\gamma^2}(B_i^T B_i)^{-1} A_i^T f_i + 0\left(\frac{1}{\gamma^4}\right). \tag{2.8}$$

Thus $\| h_i(\gamma) \| \to 0$ as $\gamma \to \infty$.

Now a direct calculation shows that

$$\nabla F(x_i) = 2 f_i^T A_i \tag{2.9}$$

so that

$$\begin{aligned} \nabla F(x_i) h_i &= -2(\| f_i \|^2 - f_i^T r_i) \\ &= -2(\| f_i \|^2 - \| r_i \|^2) \end{aligned} \tag{2.10}$$

whence, using (2.8), (2.10), and the assumed smoothness of the f_j, $j = 1, \cdots, n$,

$$\frac{F(x_i) - F(x_{i+1})}{2(F(x_i) - \| r_i \|^2)} = 1 + 0\left(\frac{1}{\gamma^2}\right) \tag{2.11}$$

as $\gamma \to \infty$.

Remark (i)

The inequality (2.1) gives a test which can be used to select an appropriate γ_i. It takes the decrease actually produced by the step h_i and compares this with that predicted by the linearised equation (1.2). The theorem says essentially that convergence is a consequence of these quantities being similar so that we do not depart too far from the region in which the linear approximation holds.

Remark (ii)

The particular case $\gamma = 0$ is equivalent to the well known Gauss–Newton method [7]. If this method is convergent (this is not guaranteed), and if $\| f \|$ is sufficiently small, then the rate of convergence can be very satisfactory. This suggests that there is an advantage in working with γ as small as possible.

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

Choosing σ small has the advantage that it makes (2.1) easier to satisfy, and such a choice probably also favours smaller values of γ.

Remark (iii)

The connection between (2.1) and the Goldstein–Armijo type tests (Daniel [2]) for standard line searches should be noted. In fact, by (2.10), $\psi(\boldsymbol{x}, \gamma)$ is the obvious extension of the Goldstein functional to allow for nonlinear dependence of the descent direction on a parameter. However, there are important differences. For example, $\|\boldsymbol{h}_i(\gamma)\|$ is bounded independent of γ. Thus arguments which depend on a large enough step being taken need not generalise (Goldstein [3]).

Remark (iv)

The gap in the global convergence result occurs because Theorem 2.2 holds for any given $\boldsymbol{x}_i$ but does not give information concerning $\sup_i \gamma_i$ which conceivably could be unbounded. Thus we have only that if $\{\gamma_i\}$ is bounded then the limit points of $\{\boldsymbol{x}_i\}$ are stationary values of $F(\boldsymbol{x})$.

On the basis of the above considerations the following algorithm is suggested.

Algorithm

(i) Set $\boldsymbol{x}_1$, $\gamma_1^{(1)}$, σ, DECR, EXPND, TOL, GMAX; $i = 1$
(ii) Set $j = 1$; compute $\boldsymbol{f}_i$, A_i, B_i
(iii) Determine $\boldsymbol{h}_i(\gamma_i^{(j)})$, $\psi(\boldsymbol{x}_i, \gamma_i^{(j)})$
(iv) If $\sigma \leqq \psi(\boldsymbol{x}_i, \gamma_i^{(j)})$ then go to (v)

else $j := j + 1$

$\gamma_i^{(j)} = \text{EXPND} * \gamma_i^{(j-1)}$

go to (iii)

(v) $\gamma_i = \gamma_i^{(j)}$; $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{h}_i(\gamma_i)$
(vi) If $\|\boldsymbol{f}_i\| - \|\boldsymbol{r}_i\| < \text{TOL}$ or $\gamma_i > \text{GMAX}$ then EXIT
(vii) If $j > 1$ then $\gamma_{i+1}^{(1)} = \gamma_i$; else $\gamma_{i+1}^{(1)} = \text{DECR} * \gamma_i$
(viii) $i := i + 1$; go to (ii)

The implementation of this algorithm is discussed in the next section. For our present purposes it is only necessary to note that σ is usually chosen small (say 10^{-4}), that TOL and GMAX test convergence and the boundedness of the $\{\gamma_i\}$ respectively, and that EXPND and DECR are fixed constants used for modifying γ. EXPND is used to increase in step (iv) to ensure that (2.1) is

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

satisfied, and DECR is used to decrease γ in step (vii) to attempt to increase the rate of convergence if the test in (iv) is satisfied with $j = 1$.

Now that the algorithm provides an explicit rule for choosing $\{\gamma_i\}$ it is possible to look at the unbounded case in more detail. In particular we can deduce information about $\nabla^2 F$. This situation can be compared with that in the standard proofs of the convergence of descent methods. For example, Ostrowski [8] assumes $\|\nabla^2 F\|$ is bounded, and Goldstein [3] assumes the existence of an equipotential spanning a bounded region.

THEOREM 2.3. Assume the sequence $\{\gamma_i\}$ determined by the algorithm is unbounded. Then the norm of the Hessian matrix $\nabla^2 F$ is also unbounded.

PROOF. If $\{\gamma_i\}$ is unbounded then there exists an unbounded sequence $\{\gamma_i^*\}$ with the property that

$$\sigma > \psi(x_i, \gamma_i^*/\text{EXPND})$$

as there must be infinitely many times that the test in step (iv) of the algorithm fails. Thus there exists an unbounded sequence $\{\hat{\gamma}_i\}$ with the property that $\sigma > \psi(x_i, \hat{\gamma}_i)$. In what follows $h_i(\hat{\gamma}_i)$ is denoted by h_i. We have, denoting mean values by a bar

$$\sigma > \frac{|\nabla F_i h_i| - \frac{1}{2}|h_i^T \overline{\nabla^2 F_i} h_i|}{|\nabla F_i h_i|}$$

so that

$$|h_i^T \overline{\nabla^2 F_i} h_i| > 2(1-\sigma)|\nabla F_i h_i|. \tag{2.12}$$

Thus

$$\|\overline{\nabla^2 F_i}\| > 2(1-\sigma)\frac{|\nabla F_i h_i|}{\|h_i\|^2} \tag{2.13}$$

From (2.7) and (2.9)

$$\begin{aligned}\nabla F_i h_i &= -\tfrac{1}{2}\nabla F_i (A_i^T A_i + \hat{\gamma}_i^2 B_i^T B_i)^{-1} \nabla F_i^T \\ &= -\tfrac{1}{2} h_i^T (A_i^T A_i + \hat{\gamma}_i^2 B_i^T B_i) h_i\end{aligned}$$

whence

$$|\nabla F_i h_i| \geqq \tfrac{1}{2}\|h_i\|^2 \frac{\hat{\gamma}_i^2}{\alpha^2} \tag{2.14}$$

where, by assumption, $1/\alpha^2$ is a lower bound for the smallest eigenvalue of $B_i^T B_i$.

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

It follows that

$$\|\overline{\nabla^2 F_i}\| > (1-\sigma)\frac{\hat{\gamma}^2}{\alpha^2}. \tag{2.15}$$

COROLLARY 2.1. If $\|\nabla^2 F\|$ is finite in any bounded region of x, then any finite limit point of $\{x_i\}$ is a stationary value of $F(x)$.

PROOF. Let x^* be a finite limit point, and let $\{x_{\pi_i}\}$ be a subsequence tending to x^*. Then $\{\gamma_{\pi_i}\}$ is bounded as the alternative implies that $\nabla^2 F$ is unbounded at a finite point as the $\{\|h_{\pi_i}\|\}$ are bounded. The result now follows by a minor modification of the argument of Theorem 2.1.

Remark (i)

In the examples considered in the next section $\|\nabla^2 F\|$ is certainly bounded in bounded regions of parameter space. An example where this may not be so is rational fitting.

3. Implementation notes

In this section we consider in more detail the nonlinear least squares algorithm given in section 2. The key steps in a computational implementation based on the use of orthogonal matrix factorisation techniques are as follows:

(i) Set initial parameters (iteration counter, initial γ etc.).
(ii) Evaluate f, $\nabla f = A$ at the current point.
(iii) Scale A to have column length 1 ($A := AD$).
(iv) Compute

$$A = Q\begin{bmatrix} U \\ 0 \end{bmatrix}, \quad \text{and} \quad g = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} = Q^T f$$

where U is upper triangular and Q orthogonal.
(v) Compute

$$\begin{bmatrix} U \\ 0 \\ \gamma I \end{bmatrix} = H \begin{bmatrix} R \\ 0 \\ 0 \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} y \\ \\ c \end{bmatrix} = H^T \begin{bmatrix} g \\ \\ 0 \end{bmatrix}$$

where H is orthogonal and R is upper triangular.
(vi) Compute

$$h(\gamma) = -DR^{-1}y_1, \quad \|r(\gamma)\|^2 = \|y_2\|^2 + \|c\|^2.$$

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

(vii) Evaluate $F(\boldsymbol{x}+\boldsymbol{h})$, $\psi(\boldsymbol{x}, \gamma)$ and test for accepting $\boldsymbol{h}$. Increase γ and go to (v) if $\boldsymbol{h}$ not accepted.

(viii) $$\boldsymbol{x} := \boldsymbol{x} + \boldsymbol{h}(\gamma)$$

If $\boldsymbol{h}$ is accepted at the first attempt decrease γ.

(ix) Test for convergencc. Go to (ii) if failure.

Note (i)

The factorisation of the augmented upper triangular matrix in step (v) is comparatively cheap if $n \gg p$ as the standard methods (Householder transformations, plane rotations) preserve the band of zeros introduced in step (iv). This observation is due to Golub. Note that a convenient expression is available for $\|\boldsymbol{r}(\gamma)\|^2$.

Note (ii)

The scaling used is equivalent to taking

$$B = \{\text{diag}\,(A^{\mathrm{T}}A)\}^{\frac{1}{2}}$$

and ensures that the terms added to the normal matrix are comparable with the original elements. With this scaling the choice $\gamma = 1$ for the initial γ is natural. The desirability of this scaling has been indicated by Marquardt [5] and Beale [1] for example. The author's experience with scaled and unscaled versions of the algorithm confirms their recommendations. It should also be noted that there is a presumption that the columns of A have finite lengths which are bounded away from zero. It is recommended that an explicit test be made to ensure that this is the case and that an error return be made if the test fails.

Note (iii)

The iteration has the familiar form of a descent calculation. Thus an inner iteration (steps (v) through (vii)) is used to satisfy a step acceptance criterion, and forms part of an outer iteration (steps (ii) through (ix)) which updates the current $\boldsymbol{x}$.

Note (iv)

Some care may be needed to avoid two calculations of $F(\boldsymbol{x})$ as the point which satisfies the acceptance criterion in the inner iteration is the point at which $\boldsymbol{f}$ and $\nabla \boldsymbol{f}$ are evaluated at the start of the next outer iteration.

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

However, this duplication can be avoided at the probable cost of some local storage in the function routine.

A FORTRAN program has been written to realize this implementation scheme and is given in detail in Appendix 1.

To illustrate the performance of the algorithm we consider the two examples given in [7], and used subsequently by several other authors. The first example is a five variable exponential fitting problem

$$f_i = -y_i + x_1 + x_2 e^{-x_3 t_i} + x_4 e^{-x_5 t_i}$$

with $1 \leqq i \leqq 33$ and $t_i = 10(i-1)$. The second example has eleven variables and is appropriate to stripping Gaussians in the presence of an exponentially decaying background. In this case we have

$$f_i = -y_i + x_1 e^{-x_5 t_i} + x_2 e^{-x_6 (t_i - x_9)^2} + x_3 e^{-x_7 (t_i - x_{10})^2} + x_4 e^{-x_8 (t_i - x_{11})^2}$$

with $1 \leqq i \leqq 65$, and $t_i = \cdot 1 * (i-1)$. The data values y_i and the initial conditions for both cases are given in the reference cited.

We present results for each example for two different initial choices of γ — both $\gamma = 1$ and $\gamma = \overline{A_{ij}}$, the average value of the moduli of the elements of A, and several different values for EXPND and DECR. The results are given in Tables 1 and 2 respectively, and include, in addition to the above data, both the number of outer iterations and the number of inner iterations. Apart from

TABLE 1.
RESULTS FOR THE EXPONENTIAL FITTING PROBLEM

EXPND	DECR	OUTER ITERATIONS	INNER ITERATIONS	INITIAL γ
1.5	.5	17	24	.0267
3.75	.2	9	12	.0267
7.5	.1	18	26	.0267
1.5	.5	20	25	1.
3.75	.2	24	33	1.
7.5	.1	27	40	1.

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

TABLE 2.
RESULTS FOR THE GAUSSIAN STRIPPING PROBLEM

EXPND	DECR	OUTER ITERATIONS	INNER ITERATIONS	INITIAL γ
1.5	.5	7	17	.0048
3.75	.2	16	25	.0048
7.5	.1	22	33	.0048
1.5	.5	8	8	1.
3.75	.2	8	9	1.
7.5	.1	14	19	1.

the first function call the number of inner iterations corresponds to the number of evaluations of the sum of squares. These results show that there is some scope for optimising the performance of the algorithm by carefully choosing the parameters; but the consistently good performance for the combination EXPND = 1.5, DECR = .5, INITIAL $\gamma = 1.$ suggests that these should be a satisfactory selection in general. Also it is not always clear how the parameters should be chosen to optimise performance. For example, in the results for the exponential fitting problem a step in which γ is multiplied by DECR is almost always followed by a step in which it is multiplied by EXPND, making this look, at first sight, like a poor strategy. However, although a steadily decreasing γ can be obtained by choosing DECR greater than about .7, the results are always worse than in the case DECR = .5, in terms of numbers of evaluations of the sum of squares. The results also indicate that DECR should not be chosen too small. Certainly DECR = .1 is always less efficient than the others.

References

[1] E. M. L. Beale, 'Numerical methods' in *Nonlinear Programming* (ed. J. Abadie), North Holland (1967) 135–205.

[2] J. W. Daniel, *The approximate minimisation of functionals*, Prentice Hall, (1971).

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

[3] A A. Goldstein, 'On steepest descent', *Siam J. Control* 3 (1965), 147–151.
[4] K. Levenberg, 'A method for the solution of certain nonlinear problems in least squares', *Quant. Appl. Math.* 2 (1944), 164–168.
[5] D W. Marquardt, 'An algorithm for least squares estimation of nonlinear parameters', *Siam J. Appl. Math.* 11 (1963), 431–441
[6] D. D Morrison, 'Methods for nonlinear least squares problems and convergence proofs', *JPL Seminar Proceedings*, (1960).
[7] M. R. Osborne, 'Some aspects of nonlinear least squares calculations in *Numerical Methods for Nonlinear Optimization* (ed. F. A. Lootsma), Academic Press, (1972).
[8] A. M. Ostrowski, '*Solutions of equations and systems of equations*', Academic Press, (1966).

Computer Centre,
Australian National University,
Canberra, A.C.T. 2601,
Australia.

Appendix 1.

FORTRAN subroutine for the nonlinear least squares algorithm. This subroutine has been implemented using the FORTRAN V compiler on a Univac 1108.

```
      SUBROUTINE LMM1(X,F,A,SUMSQ,N,NP,TOL,EXPND,DECR,ITS,IER)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 X(1),A(1),F(1),B(20,20),DA(20),DU(20),D(20),
     1C(20),DX(20),Y(20)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     LEVENBERG,MARQUART,MORRISON ALGORITHM IMPLEMENTED AFTER
C     SUGGESTION OF GOLUB (SEE OSBORNE 'SOME ASPECTS OF NON-
C     LINEAR LEAST SQUARES CALCULATIONS' EDITOR F.A. LOOTSMA
C     ACADEMIC PRESS).  MAIN FEATURE OF THIS ROUTINE IS IMPROVED
C     TEST FOR ACCEPTING PREDICTED CORRECTION AND ADJUSTING
C     LEVENBERG PARAMETER EPS
C     VARIABLES
C     X(1)      VECTOR OF INDEPENDENT VARIABLES (<=20)
C               INPUT:CONTAINS ESTIMATE OF SOLUTION
C               OUTPUT:CONTAINS SOLUTION VECTOR
C     A(1)      STORAGE OF GRAD F BY COLUMNS
C               OUTPUT:CONTAINS UPPER TRIANGULAR FACTOR IN
C               ORTHOGONAL FACTORIZATION OF GRAD F
C     F(1)      STORAGE FOR F VECTOR OF TERMS IN SUM OF SQUARES
C     SUMSQ     OUTPUT:CONTAINS SUM OF SQUARES
C     N         INPUT:DIMENSION OF F
C     NP        INPUT:DIMENSION OF X (DIM A=N*NP)
C     TOL       INPUT:TOLERANCE ON CALCULATION
C     EXPND     INPUT:FACTOR BY WHICH EPS INCREASED IF
C               TEST ON SUM SQUARE FAILS
C     DECR      INPUT:FACTOR BY WHICH EPS DECREASED IF TEST
C               QN SUM SQUARES SUCCEEDS ON FIRST ATTEMPT
C     ITS       INPUT:MAX NUMBER OF ITERATIONS
C               OUTPUT:ACTUAL NUMBER OF ITERATIONS
C     IER       INPUT:=0 NO PRINTING
C                     =1 PRINT DIAGNOSTIC INFORMATION
C               OUTPUT:=1 SUCCESSFUL TERMINATION
C                      =2 MAX ITS EXCEEDED
C                      =3 EPS EXCEEDS 1.D6
C                      =4 ATTAINABLE ACCURACY REACHED
C                         TOL TOO SMALL
C               IF IER=2,3, OR 4 THERE MAY BE ERRORS IN
C               GRADIENT CALCULATION
C                      =500+I I'TH COLUMN OF A HAS A SCALE
C                         WHICH IS SMALL COMPARED TO
C                         EUCLIDEAN NORM OF A BY A
C                         FACTOR < 1.D-6
C
C     USER SUPPLIED SUBROUTINE FUNVAL REQUIRED TO SET VALUES
C     OF SUMSQ,F,A DECLARATION MUST BE
C               SUBROUTINE FUNVAL(A,F,X,SUMSQ,IFL)
C               IF IFL=1 SETS ALL VALUES
C               IF IFL=2 SETS SUMSQ ONLY MUST NOT ALTER A,F
C
C     NOTE: A VERSION OF THIS PROGRAM INCORPORATING A NUMBER OF
C     ADDITIONAL FEATURES INCLUDING AUTOMATIC PLOTTING OF RESIDUALS
C     AND BASIC STATISTICAL TESTING HAS BEEN PREPARED BY
C     DR A.J. MILLER, CSIRO DIVISION OF MATHEMATICS AND STATISTICS.
C     VERSIONS OF THIS PROGRAM ARE AVAILABLE FOR THE CYBER 7600 AND
C     THE UNIVAC 1100/42.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

```
      IPRINT=IER
      IF (IPRINT.EQ.0) GO TO 41
      WRITE(3,102)
   41 MAXITS=ITS
      ITS=0
   40 ITS=ITS+1
      NITS=0
      CALL FUNVAL(A,F,X,SSF,1)
C**** SCALE GRAD F
      W=0.D0
      DO 1 I=1,NP
      II=(I-1)*N
      S=0.D0
      DO 2 J=1,N
    2 S=S+A(II+J)**2
      W=W+S
    1 D(I)=DSQRT(S)
      W=DSQRT(W)
      DO 46 I=1,NP
      II=(I-1)*N
      IF (D(I)/W.LT.1.D-6) GO TO 47
      S=1.D0/D(I)
      DO 3 J=1,N
    3 A(II+J)=A(II+J)*S
   46 CONTINUE
      GO TO 48
   47 IER=500+I
      IF (IPRINT.EQ.0) GO TO 49
      WRITE(3,104) I
      WRITE(3,105) (D(I),I=1,NP)
   49 GO TO 45
   48 IF (ITS.EQ.1) EPS=1.D0
      IF (IPRINT.EQ.0) GO TO 42
      WRITE(3,100) ITS,EPS,SSF
C**** HOUSEHOLDER TRANSFORMATION OF GRAD F,F
   42 DO 4 I=1,NP
      II=(I-1)*N
      S=0.D0
      DO 5 J=I,N
    5 S=S+A(II+J)**2
      S=DSQRT(S)
      IF (A(II+I).GT.0.D0) S=-S
      DA(I)=S
      A(II+I)=A(II+I)-S
      IF (I.EQ.NP) GO TO 6
      IP1=I+1
      DO 7 K=IP1,NP
      KK=(K-1)*N
      S=0.D0
      DO 8 J=I,N
    8 S=S+A(II+J)*A(KK+J)
      S=-S/(DA(I)*A(II+I))
      DO 9 J=I,N
    9 A(KK+J)=A(KK+J)-S*A(II+J)
    7 CONTINUE
    6 S=0.D0
      DO 20 J=I,N
   20 S=S+A(II+J)*F(J)
      S=-S/(DA(I)*A(II+I))
      DO 21 J=I,N
   21 F(J)=F(J)-S*A(II+J)
    4 CONTINUE
```

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

```
C**** COMPUTE SUM OF SQUARES OF RESIDUALS
      NP1=NP+1
      SSR=0.D0
      DO 22 I=NP1,N
   22 SSR=SSR+F(I)**2
C**** FACTOR EPS APENDAGE,TRANSFORM RHS
C**** UPPER TRIANGLE OF TRANSFORMED MATRIX STORED IN UPPER
C**** TRIANGLE OF B . FILL IN B STORED COLUMNWISE IN ROWS
C**** IN LOWER TRIANGLE OF B
   19 DO 30 I=1,NP
      DO 31 J=1,NP
   31 B(I,J)=0.D0
      C(I)=0.D0
   30 B(I,I)=EPS
      DO 10 I=1,NP
      II=(I-1)*N
      S=DA(I)**2
      IP1=I+1
      IL1=I-1
      DO 12 J=1,I
   12 S=S+B(I,J)**2
      S=DSQRT(S)
      IF (DA(I).GT.0.D0) S=-S
      DU(I)=S
      W=DA(I)-S
      IF (I.EQ.NP) GO TO 18
      DO 13 K=IP1,NP
      KK=(K-1)*N+I
      S=A(KK)*W
      IF (I.EQ.1) GO TO 11
      DO 14 J=1,IL1
   14 S=S+B(I,J)*B(K,J)
   11 S=-S/(DU(I)*W)
      B(I,K)=A(KK)-S*W
      DO 15 J=1,I
   15 B(K,J)=B(K,J)-S*B(I,J)
   13 CONTINUE
   18 S=F(I)*W
      DO 16 J=1,I
   16 S=S+B(I,J)*C(J)
      S=-S/(DU(I)*W)
      DX(I)=F(I)-S*W
      DO 17 J=1,I
   17 C(J)=C(J)-S*B(I,J)
   10 CONTINUE
C**** BACK SUBSTITUTION
      DX(NP)=DX(NP)/DU(NP)
      DO 25 I=2,NP
      K=NP-I+1
      KP1=K+1
      S=0.D0
      DO 26 J=KP1,NP
   26 S=S+B(K,J)*DX(J)
   25 DX(K)=(DX(K)-S)/DU(K)
      SSS=SSR
      DO 32 I=1,NP
      SSS=SSS+C(I)**2
      DX(I)=DX(I)/D(I)
   32 Y(I)=X(I)-DX(I)
      NITS=NITS+1
```

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press

```
C**** CHECK CONVERGENCE
      IER=4
      IF (SSS.GE.SSF) GO TO 45
      IER=1
      CALL FUNVAL(A,F,Y,SSN,2)
      S=.5D0*(SSF-SSN)/(SSF-SSS)
      IF (IPRINT.EQ.0) GO TO 43
      WRITE(3,103) NITS,EPS,SSN,SSS,S
   43 IF (S.GE.1.D-4) GO TO 28
      EPS=EXPND*EPS
      IER=3
      IF (EPS.GT.1.D6) GO TO 45
      GO TO 19
   28 DO 29 I=1,NP
   29 X(I)=Y(I)
      IF (IPRINT.EQ.0) GO TO 44
      WRITE(3,101) ((I,X(I)),I=1,NP)
   44 IF ((DSQRT(SSF)-DSQRT(SSS))/(1.D0+DSQRT(SSF)).GE.TOL)
     1GO TO 35
   45 SUMSQ=SSN
      DO 33 I=1,NP
      II=(I-1)*N
      A(II+I)=DA(I)
      S=D(I)
      DO 34 J=1,I
   34 A(II+J)=A(II+J)*S
   33 CONTINUE
      RETURN
   35 IER=2
      IF (ITS.GE.MAXITS) GO TO 45
      IF (NITS.EQ.1) EPS=EPS*DECR
      GO TO 40
  100 FORMAT('  ITS=',I3,' EPS=',D14.6,' SUMSQ=',D14.6)
  101 FORMAT(4(' X(',I2,')=',D14.6))
  102 FORMAT('1   NONLINEAR LEAST SQUARES BY LEVENBERG ',
     1'ALGORITHM')
  103 FORMAT(' NITS=',I3,' EPS=',D14.6,' SUMSQ=',D14.6,
     1' RES SUMSQ=',D14.6,' PSI=',D14.6)
  104 FORMAT('  SCALING ERROR NO. OF COLUMN =',I3)
  105 FORMAT(4(' D(',I2,')=',D14.6))
      END
```

https://doi.org/10.1017/S033427000000120X Published online by Cambridge University Press