E04GYF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

E04GYF is an easy-to-use quasi-Newton algorithm for finding an unconstrained minimum of a sum of squares of m nonlinear functions in n variables ($m \geq n$). First derivatives are required.

It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

2 Specification

```
SUBROUTINE E04GYF(M, N, LSFUN2, X, FSUMSQ, W, LW, IUSER, USER,
1                 IFAIL)
INTEGER           M, N, LW, IUSER(*), IFAIL
real              X(N), FSUMSQ, W(LW), USER(*)
EXTERNAL          LSFUN2
```

3 Description

This routine is similar to the subroutine LSFDQ2 in the National Physical Laboratory Algorithms Library. It is applicable to problems of the form

$$\text{Minimize } F(x) = \sum_{i=1}^{m} [f_i(x)]^2$$

where $x = (x_1, x_2, \ldots, x_n)^T$ and $m \geq n$. (The functions $f_i(x)$ are often referred to as 'residuals'.) The user must supply a subroutine to evaluate the residuals and their first derivatives at any point x.

Before attempting to minimize the sum of squares, the algorithm checks the user-supplied routine for consistency. Then, from a starting point supplied by the user, a sequence of points is generated which is intended to converge to a local minimum of the sum of squares. These points are generated using estimates of the curvature of $F(x)$.

4 References

[1] Gill P E and Murray W (1978) Algorithms for the solution of the nonlinear least-squares problem *SIAM J. Numer. Anal.* **15** 977–992

5 Parameters

1: M — INTEGER *Input*

2: N — INTEGER *Input*

On entry: the number m of residuals, $f_i(x)$, and the number n of variables, x_j.

Constraint: $1 \leq \text{N} \leq \text{M}$.

3: LSFUN2 — SUBROUTINE, supplied by the user. *External Procedure*

This routine must be supplied by the user to calculate the vector of values $f_i(x)$ and the Jacobian matrix of first derivatives $\dfrac{\partial f_i}{\partial x_j}$ at any point x. It should be tested separately before being used in conjunction with E04GYF (see the the Chapter Introduction).

Its specification is:

```
SUBROUTINE LSFUN2(M, N, XC, FVECC, FJACC, LJC, IUSER, USER)
INTEGER           M, N, LJC, IUSER(*)
real              XC(N), FVECC(M), FJACC(LJC,N), USER(*)
```

Important: the dimension declaration for FJACC must contain the variable LJC, not an integer constant.

1: M — INTEGER *Input*
2: N — INTEGER *Input*

On entry: the numbers m and n of residuals and variables, respectively.

3: XC(N) — ***real*** array *Input*

On entry: the point x at which the values of the f_i and the $\frac{\partial f_i}{\partial x_j}$ are required.

4: FVECC(M) — ***real*** array *Output*

On exit: FVECC(i) must contain the value of f_i at the point x, for $i = 1, 2, \ldots, m$.

5: FJACC(LJC,N) — ***real*** array *Output*

On exit: FJACC(i, j) must contain the value of $\frac{\partial f_i}{\partial x_j}$ at the point x, for $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$.

6: LJC — INTEGER *Input*

On entry: the first dimension of the array FJACC.

7: IUSER(∗) — INTEGER array *User Workspace*
8: USER(∗) — ***real*** array *User Workspace*

LSFUN2 is called from E04GYF with the parameters IUSER and USER as supplied to E04GYF. The user is free to use the arrays IUSER and USER to supply information to LSFUN2 as an alternative to using COMMON.

LSFUN2 must be declared as EXTERNAL in the (sub)program from which E04GYF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

4: X(N) — ***real*** array *Input/Output*

On entry: X(j) must be set to a guess at the jth component of the position of the minimum, for $j = 1, 2, \ldots, n$. The routine checks the first derivatives calculated by LSFUN2 at the starting point, and so is more likely to detect an error in the user's routine if the initial X(j) are non-zero and mutually distinct.

On exit: the lowest point found during the calculations. Thus, if IFAIL = 0 on exit, X(j) is the jth component of the position of the minimum.

5: FSUMSQ — ***real*** *Output*

On exit: the value of the sum of squares, $F(x)$, corresponding to the final point stored in X.

6: W(LW) — ***real*** array *Workspace*
7: LW — INTEGER *Input*

On entry: the length of W as declared in the (sub)program from which E04GYF is called.

Constraints:

LW $\geq 8 \times$ N $+ 2 \times$ N $\times$ N $+ 2 \times$ M $\times$ N $+ 3 \times$ M, if N > 1,
LW $\geq 11 + 5 \times$ M, if N $= 1$.

8: IUSER(∗) — INTEGER array *User Workspace*

Note: the dimension of the array IUSER must be at least 1.

IUSER is not used by E04GYF, but is passed directly to LSFUN2 and may be used to pass information to those routines.

9: USER(∗) — ***real*** array *User Workspace*

Note: the dimension of the array USER must be at least 1.

USER is not used by E04GYF, but is passed directly to LSFUN2 and may be used to pass information to those routines.

10: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to −1 before entry. **It is then essential to test the value of IFAIL on exit**.

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings specified by the routine:

IFAIL = 1

On entry, N < 1,
or M < N,
or LW < 8 × N + 2 × N × N + 2 × M × N + 3 × M, when N > 1,
or LW < 11 + 5 × M, when N = 1.

IFAIL = 2

There have been 50 × n calls of LSFUN2, yet the algorithm does not seem to have converged. This may be due to an awkward function or to a poor starting point, so it is worth restarting E04GYF from the final point held in X.

IFAIL = 3

The final point does not satisfy the conditions for acceptance as a minimum, but no lower point could be found.

IFAIL = 4

An auxiliary routine has been unable to complete a singular value decomposition in a reasonable number of sub-iterations.

IFAIL = 5, 6, 7 and 8

There is some doubt about whether the point X found by E04GYF is a minimum of $F(x)$. The degree of confidence in the result decreases as IFAIL increases. Thus, when IFAIL = 5, it is probable that the final x gives a good estimate of the position of a minimum, but when IFAIL = 8 it is very unlikely that the routine has found a minimum.

IFAIL = 9

It is very likely that the user has made an error in forming the derivatives $\frac{\partial f_i}{\partial x_j}$ in LSFUN2.

If the user is not satisfied with the result (e.g., because IFAIL lies between 3 and 8), it is worth restarting the calculations from a different starting point (not the point at which the failure occurred) in order to avoid the region which caused the failure. Repeated failure may indicate some defect in the formulation of the problem.

7 Accuracy

If the problem is reasonably well scaled and a successful exit is made then, for a computer with a mantissa of t decimals, one would expect to get $t/2-1$ decimals accuracy in the components of x and between $t-1$ (if $F(x)$ is of order 1 at the minimum) and $2t-2$ (if $F(x)$ is close to zero at the minimum) decimals accuracy in $F(x)$.

8 Further Comments

The number of iterations required depends on the number of variables, the number of residuals and their behaviour, and the distance of the starting point from the solution. The number of multiplications performed per iteration of E04GYF varies, but for $m \gg n$ is approximately $n \times m^2 + \mathrm{O}(n^3)$. In addition, each iteration makes at least one call of LSFUN2. So, unless the residuals and their derivatives can be evaluated very quickly, the run time will be dominated by the time spent in LSFUN2.

Ideally the problem should be scaled so that the minimum value of the sum of squares is in the range $(0, 1)$ and so that at points a unit distance away from the solution the sum of squares is approximately a unit value greater than at the minimum. It is unlikely that the user will be able to follow these recommendations very closely, but it is worth trying (by guesswork), as sensible scaling will reduce the difficulty of the minimization problem, so that E04GYF will take less computer time.

When the sum of squares represents the goodness of fit of a nonlinear model to observed data, elements of the variance-covariance matrix of the estimated regression coefficients can be computed by a subsequent call to E04YCF, using information returned in segments of the workspace array W. See E04YCF for further details.

9 Example

To find the least-squares estimates of x_1, x_2 and x_3 in the model

$$y = x_1 + \frac{t_1}{x_2 t_2 + x_3 t_3}$$

using the 15 sets of data given in the following table.

y	t_1	t_2	t_3
0.14	1.0	15.0	1.0
0.18	2.0	14.0	2.0
0.22	3.0	13.0	3.0
0.25	4.0	12.0	4.0
0.29	5.0	11.0	5.0
0.32	6.0	10.0	6.0
0.35	7.0	9.0	7.0
0.39	8.0	8.0	8.0
0.37	9.0	7.0	7.0
0.58	10.0	6.0	6.0
0.73	11.0	5.0	5.0
0.96	12.0	4.0	4.0
1.34	13.0	3.0	3.0
2.10	14.0	2.0	2.0
4.39	15.0	1.0	1.0

The program uses (0.5, 1.0, 1.5) as the initial guess at the position of the minimum.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     E04GYF Example Program Text.
*     Mark 19 Revised. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          M, N, NT, LW
      PARAMETER        (M=15,N=3,NT=3,LW=8*N+2*N*N+2*M*N+3*M)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             FSUMSQ
      INTEGER          I, IFAIL, J, K
*     .. Local Arrays ..
      real             T(M,NT), USER(M+M*NT), W(LW), X(N), Y(M)
      INTEGER          IUSER(1)
*     .. External Subroutines ..
      EXTERNAL         E04GYF, LSFUN2
*     .. Executable Statements ..
      WRITE (NOUT,*) 'E04GYF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*
*     Observations of TJ (J = 1, 2, 3) are held in T(I, J)
*     (I = 1, 2, . . . , 15)
*
      IUSER(1) = NT
      K = M
      DO 40 I = 1, M
         READ (NIN,*) Y(I), (T(I,J),J=1,NT)
         USER(I) = Y(I)
         DO 20 J = 1, NT
            USER(K+J) = T(I,J)
   20    CONTINUE
         K = K + NT
   40 CONTINUE
*
      X(1) = 0.5e0
      X(2) = 1.0e0
      X(3) = 1.5e0
*
      IFAIL = 1
*
      CALL E04GYF(M,N,LSFUN2,X,FSUMSQ,W,LW,IUSER,USER,IFAIL)
*
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'Error exit type', IFAIL,
     +     ' - see routine document'
      END IF
      IF (IFAIL.NE.1) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'On exit, the sum of squares is', FSUMSQ
         WRITE (NOUT,99998) 'at the point', (X(J),J=1,N)
      END IF
      STOP
*
```

```
99999 FORMAT (1X,A,I3,A)
99998 FORMAT (1X,A,3F12.4)
      END
*
      SUBROUTINE LSFUN2(M,N,XC,FVECC,FJACC,LJC,IUSER,USER)
*     Routine to evaluate the residuals and their 1st derivatives.
*     .. Scalar Arguments ..
      INTEGER           LJC, M, N
*     .. Array Arguments ..
      real              FJACC(LJC,N), FVECC(M), USER(*), XC(N)
      INTEGER           IUSER(*)
*     .. Local Scalars ..
      real              DENOM, DUMMY
      INTEGER           I, K
*     .. Executable Statements ..
      K = M
      DO 20 I = 1, M
         DENOM = XC(2)*USER(K+2) + XC(3)*USER(K+3)
         FVECC(I) = XC(1) + USER(K+1)/DENOM - USER(I)
         FJACC(I,1) = 1.0e0
         DUMMY = -1.0e0/(DENOM*DENOM)
         FJACC(I,2) = USER(K+1)*USER(K+2)*DUMMY
         FJACC(I,3) = USER(K+1)*USER(K+3)*DUMMY
         K = K + IUSER(1)
   20 CONTINUE
      RETURN
      END
```

9.2 Program Data

```
E04GYF Example Program Data
 0.14  1.0 15.0  1.0
 0.18  2.0 14.0  2.0
 0.22  3.0 13.0  3.0
 0.25  4.0 12.0  4.0
 0.29  5.0 11.0  5.0
 0.32  6.0 10.0  6.0
 0.35  7.0  9.0  7.0
 0.39  8.0  8.0  8.0
 0.37  9.0  7.0  7.0
 0.58 10.0  6.0  6.0
 0.73 11.0  5.0  5.0
 0.96 12.0  4.0  4.0
 1.34 13.0  3.0  3.0
 2.10 14.0  2.0  2.0
 4.39 15.0  1.0  1.0
```

9.3 Program Results

```
E04GYF Example Program Results

On exit, the sum of squares is      0.0082
at the point      0.0824      1.1330      2.3437
```