E04YAF – 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

E04YAF checks that a user-supplied routine for evaluating a vector of functions and the matrix of their first derivatives produces derivative values which are consistent with the function values calculated.

2 Specification

```
 SUBROUTINE E04YAF(M, N, LSQFUN, X, FVEC, FJAC, LJ, IW, LIW, W, LW,
1                  IFAIL)
 INTEGER           M, N, LJ, IW(LIW), LIW, LW, IFAIL
 real              X(N), FVEC(M), FJAC(LJ,N), W(LW)
 EXTERNAL          LSQFUN
```

3 Description

Routines for minimizing a sum of squares of m nonlinear functions (or 'residuals'), $f_i(x_1, x_2, \ldots, x_n)$, for $i = 1, 2, \ldots, m$; $m \geq n$, may require the user to supply a subroutine to evaluate the f_i and their first derivatives. E04YAF checks the derivatives calculated by such user-supplied routines, e.g., routines of the form required for E04GBF, E04GDF and E04HEF. As well as the routine to be checked (LSQFUN), the user must supply a point $x = (x_1, x_2, \ldots, x_n)^T$ at which the check will be made. E04YAF is essentially identical to CHKLSJ in the NPL Algorithms Library.

E04YAF first calls LSQFUN to evaluate the $f_i(x)$ and their first derivatives, and uses these to calculate the sum of squares $F(x) = \sum_{i=1}^{m} [f_i(x)]^2$, and its first derivatives $g_j = \left.\frac{\partial F}{\partial x_j}\right|_x$, for $j = 1, 2, \ldots, n$. The components of g along two orthogonal directions (defined by unit vectors p_1 and p_2, say) are then calculated; these will be $g^T p_1$ and $g^T p_2$ respectively. The same components are also estimated by finite differences, giving quantities

$$v_k = \frac{F(x + hp_k) - F(x)}{h}, \quad k = 1, 2$$

where h is a small positive scalar. If the relative difference between v_1 and $g^T p_1$ or between v_2 and $g^T p_2$ is judged too large, an error indicator is set.

4 References

None.

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: $1 \leq N \leq M$.

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

LSQFUN must calculate the vector of values $f_i(x)$ and their first derivatives $\frac{\partial f_i}{\partial x_j}$ at any point x. (The minimization routines mentioned in Section 3 give the user the option of resetting a parameter to terminate immediately. E04YAF will also terminate immediately, without finishing the checking process, if the parameter in question is reset.)

Its specification is:

```
      SUBROUTINE LSQFUN(IFLAG, M, N, XC, FVECC, FJACC, LJC, IW, LIW, W,
     1                  LW)
      INTEGER           IFLAG, M, N, LJC, IW(LIW), LIW, LW
      real              XC(N), FVECC(M), FJACC(LJC,N), W(LW)
```

1: IFLAG — INTEGER *Input*

On entry: to LSQFUN, IFLAG will be set to 2. If the user resets IFLAG to some negative number in LSQFUN and returns control to E04YAF, the routine will terminate immediately with IFAIL set to the user's setting of IFLAG.

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

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

4: 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.

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

On exit: unless IFLAG is reset to a negative number, FVECC(i) must contain the value of f_i at the point in XC, for $i = 1, 2, \ldots, m$.

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

On exit: unless IFLAG is reset to a negative number, FJACC(i, j) must contain the value of $\frac{\partial f_i}{\partial x_j}$ at the point in XC, for $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$.

7: LJC — INTEGER *Input*

On entry: the first dimension of the array FJACC as declared in the (sub)program from which E04YAF is called.

8: IW(LIW) — INTEGER array *Workspace*
9: LIW — INTEGER *Input*
10: W(LW) — ***real*** array *Workspace*
11: LW — INTEGER *Input*

These parameters are present so that LSQFUN will be of the form required by the minimization routines mentioned in Section 3. LSQFUN is called with the same parameters IW, LIW, W, LW as in the call to E04YAF. If the recommendation in the minimization routine document is followed, the user will have no reason to examine or change the elements of IW or W. In any case, LSQFUN **must not change** the first $3 \times \mathrm{N} + \mathrm{M} + \mathrm{M} \times \mathrm{N}$ elements of W.

LSQFUN must be declared as EXTERNAL in the (sub)program from which E04YAF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

4: X(N) — ***real*** array *Input*

On entry: X(j) ($j = 1, 2, \ldots, n$) must be set to the co-ordinates of a suitable point at which to check the derivatives calculated by LSQFUN. 'Obvious' settings, such as 0 or 1, should not be used since, at such particular points, incorrect terms may take correct values (particularly zero), so that errors can go undetected. For a similar reason, it is preferable that no two elements of X should have the same value.

5: FVEC(M) — ***real*** array *Output*

On exit: unless the user sets IFLAG negative in the first call of LSQFUN, FVEC(i) contains the value of f_i at the point given by the user in X, for $i = 1, 2, \ldots, m$.

6: FJAC(LJ,N) — ***real*** array *Output*

On exit: unless the user sets IFLAG negative in the first call of LSQFUN, FJAC(i, j) contains the value of the first derivative $\frac{\partial f_i}{\partial x_j}$ at the point given in X, as calculated by LSQFUN, for $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$.

7: LJ — INTEGER *Input*

On entry: the first dimension of the array FJAC as declared in the (sub)program from which E04YAF is called.

Constraint: LJ $\geq$ M.

8: IW(LIW) — INTEGER array *Workspace*

This array appears in the parameter list purely so that, if E04YAF is called by another library routine, the library routine can pass quantities to LSQFUN via IW. IW is not examined or changed by E04YAF. The general user must provide an array IW, but is advised not to use it.

9: LIW — INTEGER *Input*

On entry: the length of IW as declared in the (sub)program from which E04YAF is called.

Constraint: LIW ≥ 1.

10: W(LW) — ***real*** array *Workspace*

11: LW — INTEGER *Input*

On entry: the length of W as declared in the (sub)program from which E04YAF is called.

Constraint: LW $\geq 3 \times \mathrm{N} + \mathrm{M} + \mathrm{M} \times \mathrm{N}$.

12: 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 $\neq 0$ 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 < 0

A negative value of IFAIL indicates an exit from E04YAF because the user has set IFLAG negative in LSQFUN. The setting of IFAIL will be the same as the user's setting of IFLAG. The check on LSQFUN will not have been completed.

IFAIL = 1

On entry, M < N,
or N < 1,
or LJ < M,
or LIW < 1,
or LW $< 3 \times \mathrm{N} + \mathrm{M} + \mathrm{M} \times \mathrm{N}$.

IFAIL = 2

The user should check carefully the derivation and programming of expressions for the $\frac{\partial f_i}{\partial x_j}$, because it is very unlikely that LSQFUN is calculating them correctly.

7 Accuracy

IFAIL is set to 2 if

$$(v_k - g^T p_k)^2 \geq h \times ((g^T p_k)^2 + 1)$$

for $k = 1$ or 2. (See Section 3 for definitions of the quantities involved.) The scalar h is set equal to $\sqrt{\epsilon}$, where ϵ is the ***machine precision*** as given by X02AJF.

8 Further Comments

E04YAF calls LSQFUN 3 times.

Before using E04YAF to check the calculation of the first derivatives, the user should be confident that LSQFUN is calculating the residuals correctly.

E04YAF only checks the derivatives calculated by a user-supplied routine when IFLAG = 2. So, if LSQFUN is intended for use in conjunction with a minimization routine which may set IFLAG to 1, the user must check that, for given settings of the XC(j), LSQFUN produces the same values for the $\frac{\partial f_i}{\partial x_j}$ when IFLAG is set to 1 as when IFLAG is set to 2.

9 Example

Suppose that it is intended to use E04GBF or E04GDF to find 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 following program could be used to check the first derivatives calculated by the routine LSQFUN required. (The tests of whether IFLAG = 0 or 1 in LSQFUN are present ready for when LSQFUN is called by E04GBF or E04GDF. E04YAF will always call LSQFUN with IFLAG set to 2.)

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.

```
*     E04YAF Example Program Text.
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          MDEC, NDEC, LJ, LIW, LW
      PARAMETER        (MDEC=15,NDEC=3,LJ=MDEC,LIW=1,
     +                  LW=3*NDEC+MDEC+MDEC*NDEC)
      INTEGER           NIN, NOUT
      PARAMETER         (NIN=5,NOUT=6)
*     .. Arrays in Common ..
      real              T(MDEC,NDEC), Y(MDEC)
*     .. Local Scalars ..
      INTEGER           I, IFAIL, J, M, N
*     .. Local Arrays ..
      real              FJAC(LJ,NDEC), FVEC(MDEC), W(LW), X(NDEC)
      INTEGER           IW(LIW)
*     .. External Subroutines ..
      EXTERNAL          E04YAF, LSQFUN
*     .. Common blocks ..
      COMMON            Y, T
*     .. Executable Statements ..
      WRITE (NOUT,*) 'E04YAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      N = NDEC
      M = MDEC
*     Observations of TJ (J = 1, 2, 3) are held in T(I, J)
*     (I = 1, 2, . . . , 15)
      DO 20 I = 1, M
         READ (NIN,*) Y(I), (T(I,J),J=1,N)
   20 CONTINUE
*     Set up an arbitrary point at which to check the 1st
*     derivatives
      X(1) = 0.19e0
      X(2) = -1.34e0
      X(3) = 0.88e0
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'The test point is'
      WRITE (NOUT,99999) (X(J),J=1,N)
      IFAIL = 1
*
      CALL E04YAF(M,N,LSQFUN,X,FVEC,FJAC,LJ,IW,LIW,W,LW,IFAIL)
*
      WRITE (NOUT,*)
      IF (IFAIL.LT.0) THEN
         WRITE (NOUT,99998) 'IFLAG was set to ', IFAIL, 'in LSQFUN'
      ELSE IF (IFAIL.EQ.1) THEN
         WRITE (NOUT,*) 'A parameter is outside its expected range'
      ELSE
         IF (IFAIL.EQ.0) THEN
            WRITE (NOUT,*)
     +        '1st derivatives are consistent with residual values'
         ELSE IF (IFAIL.EQ.2) THEN
            WRITE (NOUT,*)
     +        'Probable error in calculation of 1st derivatives'
         END IF
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'At the test point, LSQFUN gives'
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +     '      Residuals                   1st derivatives'
         WRITE (NOUT,99997) (FVEC(I),(FJAC(I,J),J=1,N),I=1,M)
      END IF
      STOP
```

```
*
99999 FORMAT (1X,4F10.5)
99998 FORMAT (1X,A,I3,A)
99997 FORMAT (1X,1P,4e15.3)
      END
*
      SUBROUTINE LSQFUN(IFLAG,M,N,XC,FVECC,FJACC,LJC,IW,LIW,W,LW)
*     Routine to evaluate the residuals and their 1st derivatives
*     .. Parameters ..
      INTEGER           MDEC, NDEC
      PARAMETER         (MDEC=15,NDEC=3)
*     .. Scalar Arguments ..
      INTEGER           IFLAG, LIW, LJC, LW, M, N
*     .. Array Arguments ..
      real              FJACC(LJC,N), FVECC(M), W(LW), XC(N)
      INTEGER           IW(LIW)
*     .. Arrays in Common ..
      real              T(MDEC,NDEC), Y(MDEC)
*     .. Local Scalars ..
      real              DENOM, DUMMY
      INTEGER           I
*     .. Common blocks ..
      COMMON            Y, T
*     .. Executable Statements ..
      DO 20 I = 1, M
         DENOM = XC(2)*T(I,2) + XC(3)*T(I,3)
         IF (IFLAG.NE.1) FVECC(I) = XC(1) + T(I,1)/DENOM - Y(I)
         IF (IFLAG.NE.0) THEN
            FJACC(I,1) = 1.0e0
            DUMMY = -1.0e0/(DENOM*DENOM)
            FJACC(I,2) = T(I,1)*T(I,2)*DUMMY
            FJACC(I,3) = T(I,1)*T(I,3)*DUMMY
         END IF
   20 CONTINUE
      RETURN
      END
```

9.2 Program Data

```
E04YAF 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

```
E04YAF Example Program Results

The test point is
   0.19000  -1.34000   0.88000

1st derivatives are consistent with residual values

At the test point, LSQFUN gives

      Residuals                   1st derivatives
     -2.029E-03      1.000E+00     -4.061E-02     -2.707E-03
     -1.076E-01      1.000E+00     -9.689E-02     -1.384E-02
     -2.330E-01      1.000E+00     -1.785E-01     -4.120E-02
     -3.785E-01      1.000E+00     -3.043E-01     -1.014E-01
     -5.836E-01      1.000E+00     -5.144E-01     -2.338E-01
     -8.689E-01      1.000E+00     -9.100E-01     -5.460E-01
     -1.346E+00      1.000E+00     -1.810E+00     -1.408E+00
     -2.374E+00      1.000E+00     -4.726E+00     -4.726E+00
     -2.975E+00      1.000E+00     -6.076E+00     -6.076E+00
     -4.013E+00      1.000E+00     -7.876E+00     -7.876E+00
     -5.323E+00      1.000E+00     -1.040E+01     -1.040E+01
     -7.292E+00      1.000E+00     -1.418E+01     -1.418E+01
     -1.057E+01      1.000E+00     -2.048E+01     -2.048E+01
     -1.713E+01      1.000E+00     -3.308E+01     -3.308E+01
     -3.681E+01      1.000E+00     -7.089E+01     -7.089E+01
```