NAG Library Routine Document

E04YBF

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

E04YBF checks that a user-supplied subroutine for evaluating the second derivative term of the Hessian matrix of a sum of squares is consistent with a user-supplied subroutine for calculating the corresponding first derivatives.

2 Specification

```
SUBROUTINE E04YBF (M, N, LSQFUN, LSQHES, X, FVEC, FJAC, LDFJAC, B, LB,     &
                   IW, LIW, W, LW, IFAIL)

INTEGER             M, N, LDFJAC, LB, IW(LIW), LIW, LW, IFAIL
REAL (KIND=nag_wp) X(N), FVEC(M), FJAC(LDFJAC,N), B(LB), W(LW)
EXTERNAL            LSQFUN, LSQHES
```

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 $i = 1, 2, \ldots, m$ and $m \geq n$, may require you to supply a subroutine to evaluate the quantities

$$b_{jk} = \sum_{i=1}^{m} f_i \frac{\partial^2 f_i}{\partial x_j \partial x_k}$$

for $j = 1, 2, \ldots, n$ and $k = 1, 2, \ldots, j$. E04YBF is designed to check the b_{jk} calculated by such user-supplied subroutines. As well as the routine to be checked (LSQHES), you must supply a subroutine (LSQFUN) to evaluate the f_i and their first derivatives, and a point $x = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ at which the checks will be made. Note that E04YBF checks routines of the form required by E04HEF. E04YBF is essentially identical to CHKLSH in the NPL Algorithms Library.

E04YBF first calls user-supplied subroutines LSQFUN and LSQHES to evaluate the first derivatives and the b_{jk} at x. Let J denote the m by n matrix of first derivatives of the residuals. The Hessian matrix of the sum of squares,

$$G = J^{\mathrm{T}}J + B,$$

is calculated and projected onto two orthogonal vectors y and z to give the scalars $y^{\mathrm{T}}Gy$ and $z^{\mathrm{T}}Gz$ respectively. The same projections of the Hessian matrix are also estimated by finite differences, giving

$$p = (y^{\mathrm{T}}g(x + hy) - y^{\mathrm{T}}g(x))/h \quad \text{and}$$
$$q = (z^{\mathrm{T}}g(x + hz) - z^{\mathrm{T}}g(x))/h$$

respectively, where $g()$ denotes the gradient vector of the sum of squares at the point in brackets and h is a small positive scalar. If the relative difference between p and $y^{\mathrm{T}}Gy$ or between q and $z^{\mathrm{T}}Gz$ 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: $1 \leq \mathrm{N} \leq \mathrm{M}$.

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

LSQFUN must calculate the vector of values $f_i(x)$ and their first derivatives $\dfrac{\partial f_i}{\partial x_j}$ at any point x. (E04HEF gives you the option of resetting parameters of LSQFUN to cause the minimization process to terminate immediately. E04YBF will also terminate immediately, without finishing the checking process, if the parameter in question is reset.)

The specification of LSQFUN is:

```
SUBROUTINE LSQFUN (IFLAG, M, N, XC, FVEC, FJAC, LDFJAC, IW, LIW,     &
                   W, LW)

INTEGER            IFLAG, M, N, LDFJAC, IW(LIW), LIW, LW
REAL (KIND=nag_wp) XC(N), FVEC(M), FJAC(LDFJAC,N), W(LW)
```

1: IFLAG – INTEGER *Input/Output*

On entry: to LSQFUN, IFLAG will be set to 2.

On exit: if you reset IFLAG to some negative number in LSQFUN and return control to E04YBF, the routine will terminate immediately with IFAIL set to your setting of IFLAG.

2: M – INTEGER *Input*

On entry: the numbers m of residuals.

3: N – INTEGER *Input*

On entry: the numbers n of variables.

4: XC(N) – REAL (KIND=nag_wp) array *Input*

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

5: FVEC(M) – REAL (KIND=nag_wp) array *Output*

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

6: FJAC(LDFJAC, N) – REAL (KIND=nag_wp) array *Output*

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

7: LDFJAC – INTEGER *Input*

On entry: the first dimension of the array FJAC as declared in the (sub)program from which E04YBF is called.

8: IW(LIW) – INTEGER array *Workspace*
9: LIW – INTEGER *Input*
10: W(LW) – REAL (KIND=nag_wp) array *Workspace*
11: LW – INTEGER *Input*

These parameters are present so that LSQFUN will be of the form required by E04HEF. LSQFUN is called with E04YBF's parameters IW, LIW, W, LW as these parameters. If the recommendation in E04HEF is followed, you will have no reason to examine or change the elements of IW or W. In any case, LSQFUN **must not** change the first $5 \times \mathrm{N} + \mathrm{M} + \mathrm{M} \times \mathrm{N} + \mathrm{N} \times (\mathrm{N}-1)/2$ (or $6 + 2 \times \mathrm{M}$ if $\mathrm{N} = 1$) elements of W.

LSQFUN must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub)program from which E04YBF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

Note: E04YAF should be used to check the first derivatives calculated by LSQFUN before E04YBF is used to check the b_{jk} since E04YBF assumes that the first derivatives are correct.

4: LSQHES – SUBROUTINE, supplied by the user. *External Procedure*

LSQHES must calculate the elements of the symmetric matrix

$$B(x) = \sum_{i=1}^{m} f_i(x) G_i(x),$$

at any point x, where $G_i(x)$ is the Hessian matrix of $f_i(x)$. (As with LSQFUN, a parameter can be set to cause immediate termination.)

The specification of LSQHES is:

```
SUBROUTINE LSQHES (IFLAG, M, N, FVEC, XC, B, LB, IW, LIW, W, LW)
INTEGER            IFLAG, M, N, LB, IW(LIW), LIW, LW
REAL (KIND=nag_wp) FVEC(M), XC(N), B(LB), W(LW)
```

1: IFLAG – INTEGER *Input/Output*

On entry: is set to a non-negative number.

On exit: if LSQHES resets IFLAG to some negative number, E04YBF will terminate immediately, with IFAIL set to your setting of IFLAG.

2: M – INTEGER *Input*

On entry: the numbers m of residuals.

3: N – INTEGER *Input*

On entry: the numbers n of variables.

4: FVEC(M) – REAL (KIND=nag_wp) array *Input*

On entry: the value of the residual f_i at the point x, for $i = 1, 2, \ldots, m$, so that the values of the f_i can be used in the calculation of the elements of B.

5: XC(N) – REAL (KIND=nag_wp) array *Input*

On entry: the point x at which the elements of B are to be evaluated.

6: B(LB) – REAL (KIND=nag_wp) array *Output*

On exit: unless IFLAG is reset to a negative number B must contain the lower triangle of the matrix $B(x)$, evaluated at the point in XC, stored by rows. (The upper triangle is not

needed because the matrix is symmetric.) More precisely, B$(j(j-1)/2+k)$ must contain $\sum_{i=1}^{m} f_i \dfrac{\partial^2 f_i}{\partial x_j \partial x_k}$ evaluated at the point x, for $j = 1, 2, \ldots, n$ and $k = 1, 2, \ldots, j$.

7: LB – INTEGER *Input*

On entry: gives the length of the array B.

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

9: LIW – INTEGER *Input*

10: W(LW) – REAL (KIND=nag_wp) array *Workspace*

11: LW – INTEGER *Input*

As in LSQFUN, these parameters correspond to the parameters IW, LIW, W, LW of E04YBF. LSQHES **must not change** the first $5 \times \mathrm{N} + \mathrm{M} \times \mathrm{N} + \mathrm{N} \times (\mathrm{N}-1)/2$ (or $6 + 2 \times \mathrm{M}$ if $\mathrm{N} = 1$) elements of W.

LSQHES must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub)program from which E04YBF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

5: X(N) – REAL (KIND=nag_wp) array *Input*

On entry: X(j), for $j = 1, 2, \ldots, n$, must be set to the coordinates of a suitable point at which to check the b_{jk} calculated by LSQHES. '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 could go undetected. For a similar reason, it is preferable that no two elements of X should have the same value.

6: FVEC(M) – REAL (KIND=nag_wp) array *Output*

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

7: FJAC(LDFJAC, N) – REAL (KIND=nag_wp) array *Output*

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

8: LDFJAC – INTEGER *Input*

On entry: the first dimension of the array FJAC as declared in the (sub)program from which E04YBF is called.

Constraint: LDFJAC $\geq$ M.

9: B(LB) – REAL (KIND=nag_wp) array *Output*

On exit: unless you set IFLAG negative in LSQHES, B$(j \times (j-1)/2 + k)$ contains the value of b_{jk} at the point given in X as calculated by LSQHES, for $j = 1, 2, \ldots, n$ and $k = 1, 2, \ldots, j$.

10: LB – INTEGER *Input*

On entry: the dimension of the array B as declared in the (sub)program from which E04YBF is called.

Constraint: LB $\geq (\mathrm{N}+1) \times \mathrm{N}/2$.

11: IW(LIW) – INTEGER array *Workspace*

This array appears in the parameter list purely so that, if E04YBF is called by another library routine, the library routine can pass quantities to user-supplied subroutines LSQFUN and LSQHES via IW. IW is not examined or changed by E04YBF. In general you must provide an array IW, but are advised not to use it.

12: LIW – INTEGER *Input*

On entry: the actual length of IW as declared in the subroutine from which E04YBF is called.

Constraint: LIW ≥ 1.

13: W(LW) – REAL (KIND=nag_wp) array *Workspace*

14: LW – INTEGER *Input*

On entry: the actual length of W as declared in the subroutine from which E04YBF is called.

Constraints:

if $\mathrm{N} > 1$, $\mathrm{LW} \geq 5 \times \mathrm{N} + \mathrm{M} + \mathrm{M} \times \mathrm{N} + \mathrm{N} \times (\mathrm{N} - 1)/2$;
if $\mathrm{N} = 1$, $\mathrm{LW} \geq 6 + 2 \times \mathrm{M}$.

15: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, because for this routine the values of the output parameters may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

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

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).

Note: E04YBF may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the routine:

IFAIL < 0

A negative value of IFAIL indicates an exit from E04YBF because you have set IFLAG negative in user-supplied subroutines LSQFUN or LSQHES. The setting of IFAIL will be the same as your setting of IFLAG. The check on LSQHES will not have been completed.

IFAIL $= 1$

On entry, $\mathrm{M} < \mathrm{N}$,
or $\mathrm{N} < 1$,
or $\mathrm{LDFJAC} < \mathrm{M}$,
or $\mathrm{LB} < (\mathrm{N} + 1) \times \mathrm{N}/2$,
or $\mathrm{LIW} < 1$,
or $\mathrm{LW} < 5 \times \mathrm{N} + \mathrm{M} + \mathrm{M} \times \mathrm{N} + \mathrm{N} \times (\mathrm{N} - 1)/2$, if $\mathrm{N} > 1$,
or $\mathrm{LW} < 6 + 2 \times \mathrm{M}$, if $\mathrm{N} = 1$.

IFAIL $= 2$

You should check carefully the derivation and programming of expressions for the b_{jk}, because it is very unlikely that LSQHES is calculating them correctly.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

IFAIL is set to 2 if

$$|y^{\mathrm{T}}Gy - p| \geq \sqrt{h}(|y^{\mathrm{T}}Gy| + 1.0) \quad \text{or}$$
$$|z^{\mathrm{T}}Gz - q| \geq \sqrt{h}(|z^{\mathrm{T}}Gz| + 1.0)$$

where h is set equal to $\sqrt{\epsilon}$ (ϵ being the ***machine precision*** as given by X02AJF) and other quantities are defined as in Section 3.

8 Parallelism and Performance

Not applicable.

9 Further Comments

E04YBF calls LSQHES once and LSQFUN three times.

10 Example

Suppose that it is intended to use E04HEF 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

This example program could be used to check the b_{jk} calculated by LSQHES required. (The call of E04YBF is preceded by a call of E04YAF to check LSQFUN which calculates the first derivatives.)

10.1 Program Text

```
!   E04YBF Example Program Text
!   Mark 25 Release. NAG Copyright 2014.

    Module e04ybfe_mod

!     E04YBF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Accessibility Statements ..
      Private
      Public                                :: lsqfun, lsqhes
!     .. Parameters ..
      Integer, Parameter, Public            :: liw = 1, mdec = 15, ndec = 3,    &
                                               nin = 5, nout = 6
      Integer, Parameter, Public            :: lb = ndec*(ndec+1)/2
      Integer, Parameter, Public            :: ldfjac = mdec
      Integer, Parameter, Public            ::                                  &
                              lw = 5*ndec + mdec + mdec*ndec + ndec*(ndec-1)/2
!     .. Local Arrays ..
      Real (Kind=nag_wp), Public, Save      :: t(mdec,ndec), y(mdec)
    Contains
      Subroutine lsqfun(iflag,m,n,xc,fvec,fjac,ldfjac,iw,liw,w,lw)

!       Routine to evaluate the residuals and their 1st derivatives

!       .. Scalar Arguments ..
        Integer, Intent (Inout)             :: iflag
        Integer, Intent (In)                :: ldfjac, liw, lw, m, n
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Inout)  :: fjac(ldfjac,n), w(lw)
        Real (Kind=nag_wp), Intent (Out)    :: fvec(m)
        Real (Kind=nag_wp), Intent (In)     :: xc(n)
        Integer, Intent (Inout)             :: iw(liw)
!       .. Local Scalars ..
        Real (Kind=nag_wp)                  :: denom, dummy
        Integer                             :: i
!       .. Executable Statements ..
        Do i = 1, m
          denom = xc(2)*t(i,2) + xc(3)*t(i,3)
          fvec(i) = xc(1) + t(i,1)/denom - y(i)
```

```
          fjac(i,1) = 1.0E0_nag_wp
          dummy = -1.0E0_nag_wp/(denom*denom)
          fjac(i,2) = t(i,1)*t(i,2)*dummy
          fjac(i,3) = t(i,1)*t(i,3)*dummy
        End Do

        Return

      End Subroutine lsqfun
      Subroutine lsqhes(iflag,m,n,fvec,xc,b,lb,iw,liw,w,lw)

!       Routine to compute the lower triangle of the matrix B
!       (stored by rows in the array B)

!       .. Scalar Arguments ..
        Integer, Intent (Inout)             :: iflag
        Integer, Intent (In)                :: lb, liw, lw, m, n
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Out)    :: b(lb)
        Real (Kind=nag_wp), Intent (In)     :: fvec(m), xc(n)
        Real (Kind=nag_wp), Intent (Inout)  :: w(lw)
        Integer, Intent (Inout)             :: iw(liw)
!       .. Local Scalars ..
        Real (Kind=nag_wp)                  :: dummy, sum22, sum32, sum33
        Integer                             :: i
!       .. Executable Statements ..
        b(1) = 0.0E0_nag_wp
        b(2) = 0.0E0_nag_wp
        sum22 = 0.0E0_nag_wp
        sum32 = 0.0E0_nag_wp
        sum33 = 0.0E0_nag_wp

        Do i = 1, m
          dummy = 2.0E0_nag_wp*t(i,1)/(xc(2)*t(i,2)+xc(3)*t(i,3))**3
          sum22 = sum22 + fvec(i)*dummy*t(i,2)**2
          sum32 = sum32 + fvec(i)*dummy*t(i,2)*t(i,3)
          sum33 = sum33 + fvec(i)*dummy*t(i,3)**2
        End Do

        b(3) = sum22
        b(4) = 0.0E0_nag_wp
        b(5) = sum32
        b(6) = sum33

        Return

      End Subroutine lsqhes
    End Module e04ybfe_mod
    Program e04ybfe

!     E04YBF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: e04yaf, e04ybf, nag_wp
      Use e04ybfe_mod, Only: lb, ldfjac, liw, lsqfun, lsqhes, lw, mdec, ndec,  &
                             nin, nout, t, y
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Integer                             :: i, ifail, k, m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp)                  :: b(lb), fjac(ldfjac,ndec),        &
                                             fvec(mdec), w(lw), x(ndec)
      Integer                             :: iw(liw)
!     .. Executable Statements ..
      Write (nout,*) 'E04YBF Example Program Results'

!     Skip heading in data file
      Read (nin,*)

      m = mdec
```

```
      n = ndec

!     Observations of TJ (J = 1, 2, ..., n) are held in T(I, J)
!     (I = 1, 2, ..., m)

      Do i = 1, m
        Read (nin,*) y(i), t(i,1:n)
      End Do

!     Set up an arbitrary point at which to check the derivatives

      x(1:n) = (/0.19E0_nag_wp,-1.34E0_nag_wp,0.88E0_nag_wp/)

!     Check the 1st derivatives

      ifail = 0
      Call e04yaf(m,n,lsqfun,x,fvec,fjac,ldfjac,iw,liw,w,lw,ifail)

      Write (nout,*)
      Write (nout,*) 'The test point is'
      Write (nout,99999) x(1:n)

!     Check the evaluation of B

      ifail = -1
      Call e04ybf(m,n,lsqfun,lsqhes,x,fvec,fjac,ldfjac,b,lb,iw,liw,w,lw,ifail)

      If (ifail>=0 .And. ifail/=1) Then

        Select Case (ifail)
        Case (0)
          Write (nout,*)
          Write (nout,*) 'The matrix B is consistent with 1st derivatives'
        Case (2)
          Write (nout,*)
          Write (nout,*) 'Probable error in calculation of the matrix B'
        End Select

        Write (nout,*)
        Write (nout,*) 'At the test point, LSQFUN gives'
        Write (nout,*)
        Write (nout,*) '      Residuals                    1st derivatives'
        Write (nout,99998)(fvec(i),fjac(i,1:n),i=1,m)
        Write (nout,*)
        Write (nout,*) 'and LSQHES gives the lower triangle of the matrix B'
        Write (nout,*)

        k = 1

        Do i = 1, n
          Write (nout,99998) b(k:(k+i-1))
          k = k + i
        End Do

      End If

99999 Format (1X,4F10.5)
99998 Format (1X,1P,4E15.3)
    End Program e04ybfe
```

10.2 Program Data

```
E04YBF 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
```

10.3 Program Results

```
E04YBF Example Program Results

The test point is
   0.19000  -1.34000   0.88000

The matrix B is consistent with 1st derivatives

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

and LSQHES gives the lower triangle of the matrix B

      0.000E+00
      0.000E+00      1.571E+04
      0.000E+00      1.571E+04      1.571E+04
```