NAG Fortran Library Routine Document

E04HDF

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

E04HDF checks that a user-supplied routine for calculating second derivatives of an objective function is consistent with a user-supplied routine for calculating the corresponding first derivatives.

2 Specification

```
 SUBROUTINE E04HDF(N, FUNCT, HESS, X, G, HESL, LH, HESD, IW, LIW, W, LW,
1                  IFAIL)
 INTEGER           N, LH, IW(LIW), LIW, LW, IFAIL
 real              X(N), G(N), HESL(LH), HESD(N), W(LW)
 EXTERNAL          FUNCT, HESS
```

3 Description

Routines for minimizing a function $F(x_1, x_2, \ldots, x_n)$ of the variables $x_1, x_2, \ldots, x_n$ may require the user to provide a subroutine to evaluate the second derivatives of F. E04HDF is designed to check the second derivatives calculated by such user-supplied routines. As well as the routine to be checked (HESS), the user must supply a routine (FUNCT) to evaluate the first derivatives, and a point $x = (x_1, x_2, \ldots, x_n)^T$ at which the checks will be made. Note that E04HDF checks routines of the form required for E04LBF.

E04HDF first calls FUNCT and HESS to evaluate the first and second derivatives of F at x. The user-supplied Hessian matrix (H, say) is projected onto two orthogonal vectors y and z to give the scalars $y^T Hy$ and $z^T Hz$ respectively. The same projections of the Hessian matrix are also estimated by finite differences, giving

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

respectively, where $g(\ )$ denotes the vector of first derivatives at the point in brackets and h is a small positive scalar. If the relative difference between p and $y^T Hy$ or between q and $z^T Hz$ is judged too large, an error indicator is set.

4 References

None.

5 Parameters

1: N – INTEGER *Input*

On entry: the number n of independent variables in the objective function.

Constraint: $N \geq 1$.

2: FUNCT – SUBROUTINE, supplied by the user. *External Procedure*

FUNCT must evaluate the function and its first derivatives at a given point. (E04LBF gives the user the option of resetting a parameter of FUNCT to cause the minimization process to terminate immediately. E04HDF will also terminate immediately, without finishing the checking process, if the parameter in question is reset.)

Its specification is:

```
SUBROUTINE FUNCT(IFLAG, N, XC, FC, GC, IW, LIW, W, LW)
INTEGER          IFLAG, N, IW(LIW), LIW, LW
real             XC(N), FC, GC(N), W(LW)
```

1: IFLAG – INTEGER *Input/Output*

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

On exit: if the user sets IFLAG to some negative number in FUNCT and returns control to E04HDF, E04HDF will terminate immediately with IFAIL set to the user's setting of IFLAG.

2: N – INTEGER *Input*

On entry: the number n of variables.

3: XC(N) – ***real*** array *Input*

On entry: the point x at which the function and first derivatives are required.

4: FC – ***real*** *Output*

On exit: unless FUNCT resets IFLAG, FC must be set to the value of the objective function F at the current point x.

5: GC(N) – ***real*** array *Output*

On exit: unless FUNCT resets IFLAG, GC(j) must be set to the value of the first derivative $\frac{\partial F}{\partial x_j}$ at the point x, for $j = 1, 2, \ldots, n$.

6: IW(LIW) – INTEGER array *Workspace*
7: LIW – INTEGER *Input*
8: W(LW) – ***real*** array *Workspace*
9: LW – INTEGER *Input*

These parameters are present so that FUNCT will be of the form required by E04LBF. FUNCT is called with E04HDF's parameters IW, LIW, W, LW as these parameters. If the advice given in E04LBF is being followed, the user will have no reason to examine or change any elements of IW or W. In any case, FUNCT **must not change** the first 5 × N elements of W.

FUNCT must be declared as EXTERNAL in the (sub)program from which E04HDF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

E04HCF should be used to check the first derivatives calculated by FUNCT before E04HDF is used to check the second derivatives, since E04HDF assumes that the first derivatives are correct.

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

HESS must evaluate the second derivatives of the function at a given point. (As with FUNCT, a parameter can be set to cause immediate termination.)

Its specification is:

```
SUBROUTINE HESS(IFLAG, N, XC, FHESL, LH, FHESD, IW, LIW, W, LW)
INTEGER         IFLAG, N, LH, IW(LIW), LIW, LW
real            XC(N), FHESL(LH), FHESD(N), W(LW)
```

1: IFLAG – INTEGER *Input/Output*

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

On exit: if HESS resets IFLAG to a negative number, E04HDF will terminate immediately with IFAIL set to the user's setting of IFLAG.

2: N – INTEGER *Input*

On entry: the number n of variables.

3: XC(N) – ***real*** array *Input*

On entry: the point x at which the second derivatives of $F(x)$ are required.

4: FHESL(LH) – ***real*** array *Output*

On exit: unless IFLAG is reset, HESS must place the strict lower triangle of the second derivative matrix of F (evaluated at the point x) in FHESL, stored by rows, i.e., FHESL$((i-1)(i-2)/2+j)$ must be set to the value of $\frac{\partial^2 F}{\partial x_i \partial x_j}$ at the point x, for $i = 2, 3, \ldots, n$; $j = 1, 2, \ldots, i-1$. (The upper triangle is not required because the matrix is symmetric.)

5: LH – INTEGER *Input*

On entry: the length of the array FHESL.

6: FHESD(N) – ***real*** array *Input/Output*

On entry: contains the value of $\frac{\partial F}{\partial x_j}$ at the point x, for $j = 1, 2, \ldots, n$. Routines written to take advantage of a similar feature of E04LBF can be tested as they stand by E04HDF.

On exit: unless IFLAG is reset, HESS must place the diagonal elements of the second derivative matrix of F (evaluated at the point x) in FHESD, i.e., FHESD(j) must be set to the value of $\frac{\partial^2 F}{\partial x_j^2}$ at the point x, for $j = 1, 2, \ldots, n$.

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

As in FUNCT, these parameters correspond to the parameters IW, LIW, W and LW of E04HDF. HESS **must not change** the first $5 \times$ N elements of W.

HESS must be declared as EXTERNAL in the (sub)program from which E04HDF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

4: X(N) – ***real*** array *Input*

On entry: X(j), for $j = 1, 2, \ldots, n$ must contain the co-ordinates of a suitable point at which to check the derivatives calculated by FUNCT. 'Obvious' settings, such as 0.0 or 1.0, should not be used since, at such particular points, incorrect terms may take correct values (particularly zero), so that errors could go undetected. Similarly, it is advisable that no two elements of X should be the same.

5: G(N) – ***real*** array *Output*

On exit: unless the user sets IFLAG negative in the first call of FUNCT, G(j) contains the value of the first derivative $\frac{\partial F}{\partial x_j}$ at the point given in X, as calculated by FUNCT, for $j = 1, 2, \ldots, $N.

6: HESL(LH) – ***real*** array *Output*

On exit: unless the user sets IFLAG negative in HESS, HESL contains the strict lower triangle of the second derivative matrix of F, as evaluated by HESS at the point given in X, stored by rows.

7: LH – INTEGER *Input*

On entry: the dimension of the array HESL as declared in the (sub)program from which E04HDF is called.

Constraint: $\mathrm{LH} \geq \max(1, \mathrm{N} \times (\mathrm{N}-1)/2)$.

8: HESD(N) – ***real*** array *Output*

On exit: unless the user set IFLAG negative in HESS, HESD contains the diagonal elements of the second derivative matrix of F, as evaluated by HESS at the point given in X.

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

This array is in the parameter list so that it can be used by other library routines for passing INTEGER quantities to FUNCT or HESS. It is not examined or changed by E04HDF. The general user must provide an array IW, but is advised not to use it.

10: LIW – INTEGER *Input*

On entry: the dimension of the array IW as declared in the (sub)program from which E04HDF is called.

Constraint: $\mathrm{LIW} \geq 1$.

11: W(LW) – ***real*** array *Workspace*
12: LW – INTEGER *Input*

On entry: the dimension of the array W as declared in the (sub)program from which E04HDF is called.

Constraint: $\mathrm{LW} \geq 5 \times \mathrm{N}$.

13: 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: $\mathrm{IFAIL} = 0$ unless the routine detects an error (see Section 6).

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 $\mathrm{IFAIL} \neq 0$ 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.**

6 Error Indicators and Warnings

If on entry $\mathrm{IFAIL} = 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL < 0

A negative value of IFAIL indicates an exit from E04HDF because the user has set IFLAG negative in FUNCT or HESS. The setting of IFLAG will be the same as the user's setting of IFLAG. The check on HESS will not have been completed.

IFAIL = 1

On entry, N < 1,
or LH < max(1, N × (N − 1)/2),
or LIW < 1,
or LW < 5 × N.

IFAIL = 2

The user should check carefully the derivation and programming of expressions for the second derivatives of $F(x)$, because it is very unlikely that HESS is calculating them correctly.

7 Accuracy

IFAIL is set to 2 if

$$|y^T Hy - p| \geq \sqrt{h} \times (|y^T Hy| + 1.0) \quad \text{or}$$
$$|z^T Hz - q| \geq \sqrt{h} \times (|z^T Hz| + 1.0)$$

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

8 Further Comments

E04HDF calls HESS once and FUNCT three times.

9 Example

Suppose that it is intended to use E04LBF to minimize

$$F = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$$

The following program could be used to check the second derivatives calculated by the routine HESS required. (The call of E04HDF is preceded by a call of E04HCF to check the routine FUNCT which calculates the first derivatives.)

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.

```
*     E04HDF Example Program Text.
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          N, LH, LIW, LW
      PARAMETER        (N=4,LH=N*(N-1)/2,LIW=1,LW=5*N)
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Local Scalars ..
      real             F
      INTEGER          I, IFAIL, J, K
*     .. Local Arrays ..
      real             G(N), HESD(N), HESL(LH), W(LW), X(N)
      INTEGER          IW(LIW)
*     .. External Subroutines ..
      EXTERNAL         E04HCF, E04HDF, FUNCT, HESS
*     .. Executable Statements ..
      WRITE (NOUT,*) 'E04HDF Example Program Results'
*     Set up an arbitrary point at which to check the derivatives
      X(1) = 1.46e0
      X(2) = -0.82e0
      X(3) = 0.57e0
      X(4) = 1.21e0
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'The test point is'
```

```
      WRITE (NOUT,99999) (X(J),J=1,N)
*     Check the 1st derivatives
      IFAIL = 0
*
      CALL E04HCF(N,FUNCT,X,F,G,IW,LIW,W,LW,IFAIL)
*
*     Check the 2nd derivatives
      IFAIL = 1
*
      CALL E04HDF(N,FUNCT,HESS,X,G,HESL,LH,HESD,IW,LIW,W,LW,IFAIL)
*
      WRITE (NOUT,*)
      IF (IFAIL.LT.0) THEN
         WRITE (NOUT,99998) 'IFLAG was set to ', IFAIL,
     +     'in FUNCT or HESS'
      ELSE IF (IFAIL.EQ.1) THEN
         WRITE (NOUT,*) 'A parameter is outside its expected range'
      ELSE
         IF (IFAIL.EQ.0) THEN
            WRITE (NOUT,*)
     +        '2nd derivatives are consistent with 1st derivatives'
         ELSE IF (IFAIL.EQ.2) THEN
            WRITE (NOUT,*)
     +        'Probable error in calculation of 2nd derivatives'
         END IF
         WRITE (NOUT,*)
         WRITE (NOUT,99997)
     +     'At the test point, FUNCT gives the function value', F
         WRITE (NOUT,*) 'and the 1st derivatives'
         WRITE (NOUT,99996) (G(J),J=1,N)
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +     'HESS gives the lower triangle of the Hessian matrix'
         WRITE (NOUT,99995) HESD(1)
         K = 1
         DO 20 I = 2, N
            WRITE (NOUT,99995) (HESL(J),J=K,K+I-2), HESD(I)
            K = K + I - 1
   20    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,4F9.4)
99998 FORMAT (1X,A,I3,A)
99997 FORMAT (1X,A,1P,e12.4)
99996 FORMAT (1X,1P,4e12.3)
99995 FORMAT (1X,1P,4e12.3)
      END
*
      SUBROUTINE FUNCT(IFLAG,N,XC,FC,GC,IW,LIW,W,LW)
*     Routine to evaluate objective function and its 1st derivatives.
*     .. Scalar Arguments ..
      real              FC
      INTEGER           IFLAG, LIW, LW, N
*     .. Array Arguments ..
      real              GC(N), W(LW), XC(N)
      INTEGER           IW(LIW)
*     .. Executable Statements ..
      FC = (XC(1)+10.0e0*XC(2))**2 + 5.0e0*(XC(3)-XC(4))**2 + (XC(2)
     +     -2.0e0*XC(3))**4 + 10.0e0*(XC(1)-XC(4))**4
      GC(1) = 2.0e0*(XC(1)+10.0e0*XC(2)) + 40.0e0*(XC(1)-XC(4))**3
      GC(2) = 20.0e0*(XC(1)+10.0e0*XC(2)) + 4.0e0*(XC(2)-2.0e0*XC(3))**3
      GC(3) = 10.0e0*(XC(3)-XC(4)) - 8.0e0*(XC(2)-2.0e0*XC(3))**3
      GC(4) = 10.0e0*(XC(4)-XC(3)) - 40.0e0*(XC(1)-XC(4))**3
      RETURN
      END
*
      SUBROUTINE HESS(IFLAG,N,XC,FHESL,LH,FHESD,IW,LIW,W,LW)
*     Routine to evaluate 2nd derivatives
*     .. Scalar Arguments ..
      INTEGER          IFLAG, LH, LIW, LW, N
```

```
*     .. Array Arguments ..
      real              FHESD(N), FHESL(LH), W(LW), XC(N)
      INTEGER           IW(LIW)
*     .. Executable Statements ..
      FHESD(1) = 2.0e0 + 120.0e0*(XC(1)-XC(4))**2
      FHESD(2) = 200.0e0 + 12.0e0*(XC(2)-2.0e0*XC(3))**2
      FHESD(3) = 10.0e0 + 48.0e0*(XC(2)-2.0e0*XC(3))**2
      FHESD(4) = 10.0e0 + 120.0e0*(XC(1)-XC(4))**2
      FHESL(1) = 20.0e0
      FHESL(2) = 0.0e0
      FHESL(3) = -24.0e0*(XC(2)-2.0e0*XC(3))**2
      FHESL(4) = -120.0e0*(XC(1)-XC(4))**2
      FHESL(5) = 0.0e0
      FHESL(6) = -10.0e0
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 E04HDF Example Program Results

 The test point is
    1.4600  -0.8200   0.5700   1.2100

 2nd derivatives are consistent with 1st derivatives

 At the test point, FUNCT gives the function value  6.2273E+01
 and the 1st derivatives
   -1.285E+01  -1.649E+02   5.384E+01   5.775E+00

 HESS gives the lower triangle of the Hessian matrix
    9.500E+00
    2.000E+01   2.461E+02
    0.000E+00  -9.220E+01   1.944E+02
   -7.500E+00   0.000E+00  -1.000E+01   1.750E+01
```