NAG Toolbox for MATLAB

e04hd

1 Purpose

e04hd checks that a (sub)program for calculating second derivatives of an objective function is consistent with a (sub)program for calculating the corresponding first derivatives.

2 Syntax

```
[g, hesl, hesd, iw, w, ifail] = e04hd(funct, h, x, lh, iw, w, 'n', n,
'liw', liw, 'lw', lw)
```

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 you to provide a (sub)program to evaluate the second derivatives of F. e04hd is designed to check the second derivatives calculated by such user-supplied (sub)programs. As well as the function to be checked (**h**), you must supply a (sub)program (**funct**) to evaluate the first derivatives, and a point $x = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ at which the checks will be made. Note that e04hd checks functions of the form required for e04lb.

e04hd first calls user-supplied (sub)programs **funct** and **h** 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 $y^{\mathrm{T}}Hy$ and $z^{\mathrm{T}}Hz$ respectively. The same projections of the Hessian matrix are also estimated by finite differences, giving

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

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 $y^{\mathrm{T}}Hy$ or between q and $z^{\mathrm{T}}Hz$ is judged too large, an error indicator is set.

4 References

None.

5 Parameters

5.1 Compulsory Input Parameters

1: **funct – string containing name of m-file**

funct must evaluate the function and its first derivatives at a given point. (e04lb gives you the option of resetting parameters of **funct** to cause the minimization process to terminate immediately. e04hd will also terminate immediately, without finishing the checking process, if the parameter in question is reset.)

Its specification is:

```
[iflag, fc, gc, iw, w] = funct(iflag, n, xc, iw, liw, w, lw)
```

Input Parameters

1: **iflag – int32 scalar**

To **funct**, **iflag** will be set to 2.

If you set **iflag** to some negative number in **funct** and return control to e04hd, e04hd will terminate immediately with **ifail** set to your setting of **iflag**.

2: **n – int32 scalar**

The number n of variables.

3: **xc(n) – double array**

The point x at which the function and first derivatives are required.

4: **iw(liw) – int32 array**
5: **liw – int32 scalar**
6: **w(lw) – double array**
7: **lw – int32 scalar**

These parameters are present so that **funct** will be of the form required by e04lb. **funct** is called with e04hd's parameters **iw**, **liw**, **w**, **lw** as these parameters. If the advice given in e04lb is being followed, you will have no reason to examine or change any elements of **iw** or **w**. In any case, **funct must not change** the first $5 \times$ **n** elements of **w**.

Output Parameters

1: **iflag – int32 scalar**

To **funct**, **iflag** will be set to 2.

If you set **iflag** to some negative number in **funct** and return control to e04hd, e04hd will terminate immediately with **ifail** set to your setting of **iflag**.

2: **fc – double scalar**

Unless **funct** resets **iflag**, **fc** must be set to the value of the objective function F at the current point x.

3: **gc(n) – double array**

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 $j = 1, 2, \ldots, n$.

4: **iw(liw) – int32 array**
5: **w(lw) – double array**

These parameters are present so that **funct** will be of the form required by e04lb. **funct** is called with e04hd's parameters **iw**, **liw**, **w**, **lw** as these parameters. If the advice given in e04lb is being followed, you will have no reason to examine or change any elements of **iw** or **w**. In any case, **funct must not change** the first $5 \times$ **n** elements of **w**.

e04hc should be used to check the first derivatives calculated by **funct** before e04hd is used to check the second derivatives, since e04hd assumes that the first derivatives are correct.

2: **h – string containing name of m-file**

h must evaluate the second derivatives of the function at a given point. (As with user-supplied (sub)program **funct**, a parameter can be set to cause immediate termination.)

Its specification is:

```
[iflag, fhesl, fhesd, iw, w] = h(iflag, n, xc, lh, fhesd, iw, liw,
w, lw)
```

Input Parameters

1: **iflag – int32 scalar**

Is set to a nonnegative number.

If **h** resets **iflag** to a negative number, e04hd will terminate immediately with **ifail** set to your setting of **iflag**.

2: **n – int32 scalar**

The number n of variables.

3: **xc(n) – double array**

The point x at which the second derivatives of $F(x)$ are required.

4: **lh – int32 scalar**

The length of the array **fhesl**.

5: **fhesd(n) – double array**

Contains the value of $\frac{\partial F}{\partial x_j}$ at the point x, for $j = 1, 2, \ldots, n$. Functions written to take advantage of a similar feature of e04lb can be tested as they stand by e04hd.

Unless **iflag** is reset, **h** 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 $j = 1, 2, \ldots, n$.

6: **iw(liw) – int32 array**
7: **liw – int32 scalar**
8: **w(lw) – double array**
9: **lw – int32 scalar**

As in user-supplied (sub)program **funct**, these parameters correspond to the parameters **iw**, **liw**, **w** and **lw** of e04hd. **h must not change** the first $5 \times$ **n** elements of **w**.

Output Parameters

1: **iflag – int32 scalar**

Is set to a nonnegative number.

If **h** resets **iflag** to a negative number, e04hd will terminate immediately with **ifail** set to your setting of **iflag**.

2: **fhesl(lh) – double array**

Unless **iflag** is reset, **h** 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 $i = 2, 3, \ldots, n$ and $j = 1, 2, \ldots, i-1$. (The upper triangle is not required because the matrix is symmetric.)

3: **fhesd(n) – double array**

Contains the value of $\frac{\partial F}{\partial x_j}$ at the point x, for $j = 1, 2, \ldots, n$. Functions written to take advantage of a similar feature of e04lb can be tested as they stand by e04hd.

Unless **iflag** is reset, **h** 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 $j = 1, 2, \ldots, n$.

4: **iw(liw) – int32 array**
5: **w(lw) – double array**

As in user-supplied (sub)program **funct**, these parameters correspond to the parameters **iw**, **liw**, **w** and **lw** of e04hd. **h must not change** the first $5 \times$ **n** elements of **w**.

3: **x(n) – double array**

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 user-supplied (sub)program **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.

4: **lh – int32 scalar**

Constraint: **lh** $\geq \max(1, \mathbf{n} \times (\mathbf{n} - 1)/2)$.

5: **iw(liw) – int32 array**

This array is in the parameter list so that it can be used by other library functions for passing integer quantities to user-supplied (sub)program **funct** or user-supplied (sub)program **h**. It is not examined or changed by e04hd. In general you must provide an array **iw**, but are advised not to use it.

6: **w(lw) – double array**

Constraint: **lw** $\geq 5 \times$ **n**.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the arrays **x**, **g**, **hesd**. (An error is raised if these dimensions are not equal.)

the number n of independent variables in the objective function.

Constraint: **n** ≥ 1.

2: **liw – int32 scalar**

Default: The dimension of the array **iw**.

Constraint: **liw** ≥ 1.

3: **lw – int32 scalar**

Default: The dimension of the array **w**.

Constraint: **lw** $\geq 5 \times$ **n**.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **g(n) – double array**

Unless you set **iflag** negative in the first call of user-supplied (sub)program **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 $j = 1, 2, \ldots, \mathbf{n}$.

2: **hesl(lh) – double array**

Unless you set **iflag** negative in the user-supplied (sub)program **h**, **hesl** contains the strict lower triangle of the second derivative matrix of F, as evaluated by **h** at the point given in **x**, stored by rows.

3: **hesd(n) – double array**

Unless you set **iflag** negative in the user-supplied (sub)program **h**, **hesd** contains the diagonal elements of the second derivative matrix of F, as evaluated by **h** at the point given in **x**.

4: **iw(liw) – int32 array**

This array is in the parameter list so that it can be used by other library functions for passing integer quantities to user-supplied (sub)program **funct** or user-supplied (sub)program **h**. It is not examined or changed by e04hd. In general you must provide an array **iw**, but are advised not to use it.

5: **w(lw) – double array**

6: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Note: e04hd may return useful information for one or more of the following detected errors or warnings.

ifail < 0

A negative value of **ifail** indicates an exit from e04hd because you have set **iflag** negative in the user-supplied (sub)programs **funct** or **h**. The setting of **iflag** will be the same as your setting of **iflag**. The check on **h** will not have been completed.

ifail $= 1$

On entry, $\mathbf{n} < 1$,
or $\mathbf{lh} < \max(1, \mathbf{n} \times (\mathbf{n} - 1)/2)$,
or $\mathbf{liw} < 1$,
or $\mathbf{lw} < 5 \times \mathbf{n}$.

ifail $= 2$

You should check carefully the derivation and programming of expressions for the second derivatives of $F(x)$, because it is very unlikely that user-supplied (sub)program **h** is calculating them correctly.

7 Accuracy

ifail is set to 2 if

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

where h is set equal to $\sqrt{\epsilon}$ (ϵ being the ***machine precision*** as given by x02aj) and other quantities are as defined in Section 3.

8 Further Comments

e04hd calls user-supplied (sub)program **h** once and user-supplied (sub)program **funct** three times.

9 Example

```
e04hd_funct.m
```

```
function [iflag, fc, gc] = funct(iflag, n, xc)
  gc = zeros(n, 1);
  fc = (xc(1)+10*xc(2))^2 + 5*(xc(3)-xc(4))^2 + (xc(2)-2*xc(3))^4 + ...
       10*(xc(1)-xc(4))^4;
  gc(1) = 2*(xc(1)+10*xc(2)) + 40*(xc(1)-xc(4))^3;
  gc(2) = 20*(xc(1)+10*xc(2)) + 4*(xc(2)-2*xc(3))^3;
  gc(3) = 10*(xc(3)-xc(4)) - 8*(xc(2)-2*xc(3))^3;
  gc(4) = 10*(xc(4)-xc(3)) - 40*(xc(1)-xc(4))^3;
```

```
e04hd_hess.m
```

```
function [iflag, fhesl, fhesd] = hess(iflag, n, xc, lh, fhesd)
  fhesl = zeros(lh, 1);
  fhesd = zeros(n, 1);

  fhesd(1) = 2 + 120*(xc(1)-xc(4))^2;
  fhesd(2) = 200 + 12*(xc(2)-2*xc(3))^2;
  fhesd(3) = 10 + 48*(xc(2)-2*xc(3))^2;
  fhesd(4) = 10 + 120*(xc(1)-xc(4))^2;

  fhesl(1) = 20;
  fhesl(2) = 0;
  fhesl(3) = -24*(xc(2)-2*xc(3))^2;
  fhesl(4) = -120*(xc(1)-xc(4))^2;
  fhesl(5) = 0;
  fhesl(6) = -10;
```

```
x = [1.46;
     -0.82;
     0.57;
     1.21];
lh = int32(6);
iw = [int32(0)];
w =  zeros(20,1);
[g, hesl, hesd, iwOut, wOut, ifail] = e04hd('e04hd_funct', 'e04hd_hess',
x, lh, iw, w)
```

```
g =
  -12.8550
 -164.9181
   53.8363
    5.7750
hesl =
   20.0000
```

```
          0
   -92.1984
    -7.5000
          0
   -10.0000
hesd =
     9.5000
   246.0992
   194.3968
    17.5000
iwOut =
           0
wOut =
    -0.0170
    -0.2808
    -0.5983
     0.1863
     0.5726
     0.3279
     0.1075
     0.8916
     1.4600
    -0.8200
     0.5700
     1.2100
     5.4397
    92.1407
    -9.3317
    10.2327
   -12.8550
  -164.9181
    53.8363
     5.7750
ifail =
           0
```