NAG Toolbox

nag_opt_one_var_deriv (e04bb)

1 Purpose

nag_opt_one_var_deriv (e04bb) searches for a minimum, in a given finite interval, of a continuous function of a single variable, using function and first derivative values. The method (based on cubic interpolation) is intended for functions which have a continuous first derivative (although it will usually work if the derivative has occasional discontinuities).

2 Syntax

```
[e1, e2, a, b, maxcal, x, f, g, user, ifail] = nag_opt_one_var_deriv(funct, e1,
e2, a, b, maxcal, 'user', user)
```

```
[e1, e2, a, b, maxcal, x, f, g, user, ifail] = e04bb(funct, e1, e2, a, b, maxcal,
'user', user)
```

3 Description

nag_opt_one_var_deriv (e04bb) is applicable to problems of the form:

$$\text{Minimize}\, F(x) \quad \text{subject to} \quad a \leq x \leq b$$

when the first derivative $\frac{dF}{dx}$ can be calculated. The function normally computes a sequence of x values which tend in the limit to a minimum of $F(x)$ subject to the given bounds. It also progressively reduces the interval $[a, b]$ in which the minimum is known to lie. It uses the safeguarded cubic-interpolation method described in Gill and Murray (1973).

You must supply a **funct** to evaluate $F(x)$ and $\frac{dF}{dx}$. The arguments **e1** and **e2** together specify the accuracy

$$Tol(x) = \mathbf{e1} \times |x| + \mathbf{e2}$$

to which the position of the minimum is required. Note that **funct** is never called at a point which is closer than $Tol(x)$ to a previous point.

If the original interval $[a, b]$ contains more than one minimum, nag_opt_one_var_deriv (e04bb) will normally find one of the minima.

4 References

Gill P E and Murray W (1973) Safeguarded steplength algorithms for optimization using descent methods *NPL Report NAC 37* National Physical Laboratory

5 Parameters

5.1 Compulsory Input Parameters

1: **funct** – SUBROUTINE, supplied by the user.

You must supply this function to calculate the values of $F(x)$ and $\frac{dF}{dx}$ at any point x in $[a, b]$.

It should be tested separately before being used in conjunction with nag_opt_one_var_deriv (e04bb).

```
[fc, gc, user] = funct(xc, user)
```

Input Parameters

1: **xc** – REAL (KIND=nag_wp)

The point x at which the values of F and $\frac{dF}{dx}$ are required.

2: **user** – INTEGER array

funct is called from nag_opt_one_var_deriv (e04bb) with the object supplied to nag_opt_one_var_deriv (e04bb).

Output Parameters

1: **fc** – REAL (KIND=nag_wp)

Must be set to the value of the function F at the current point x.

2: **gc** – REAL (KIND=nag_wp)

Must be set to the value of the first derivative $\frac{dF}{dx}$ at the current point x.

3: **user** – INTEGER array

2: **e1** – REAL (KIND=nag_wp)

The relative accuracy to which the position of a minimum is required. (Note that, since **e1** is a relative tolerance, the scaling of x is automatically taken into account.)

e1 should be no smaller than 2ϵ, and preferably not much less than $\sqrt{\epsilon}$, where ϵ is the ***machine precision***.

3: **e2** – REAL (KIND=nag_wp)

The absolute accuracy to which the position of a minimum is required. **e2** should be no smaller than 2ϵ.

4: **a** – REAL (KIND=nag_wp)

The lower bound a of the interval containing a minimum.

5: **b** – REAL (KIND=nag_wp)

The upper bound b of the interval containing a minimum.

6: **maxcal** – INTEGER

The maximum number of calls of **funct** to be allowed.

Constraint: **maxcal** ≥ 2. (Few problems will require more than 20.)

There will be an error exit (see Section 6) after **maxcal** calls of **funct**

5.2 Optional Input Parameters

1: **user** – INTEGER array

user is not used by nag_opt_one_var_deriv (e04bb), but is passed to **funct**. Note that for large objects it may be more efficient to use a global variable which is accessible from the m-files than to use **user**.

5.3 Output Parameters

1: **e1** – REAL (KIND=nag_wp)

If you set **e1** to 0.0 (or to any value less than ϵ), **e1** will be reset to the default value $\sqrt{\epsilon}$ before starting the minimization process.

2: **e2** – REAL (KIND=nag_wp)

If you set **e2** to 0.0 (or to any value less than ϵ), **e2** will be reset to the default value $\sqrt{\epsilon}$.

3: **a** – REAL (KIND=nag_wp)

An improved lower bound on the position of the minimum.

4: **b** – REAL (KIND=nag_wp)

An improved upper bound on the position of the minimum.

5: **maxcal** – INTEGER

The total number of times that **funct** was actually called.

6: **x** – REAL (KIND=nag_wp)

The estimated position of the minimum.

7: **f** – REAL (KIND=nag_wp)

The function value at the final point given in **x**.

8: **g** – REAL (KIND=nag_wp)

The value of the first derivative at the final point in **x**.

9: **user** – INTEGER array

10: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Note: nag_opt_one_var_deriv (e04bb) may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the function:

ifail $= 1$ (*warning*)

On entry, $(\mathbf{a} + \mathbf{e2}) \geq \mathbf{b}$,
or $\mathbf{maxcal} < 2$.

ifail $= 2$ (*warning*)

The number of calls of **funct** has exceeded **maxcal**. This may have happened simply because **maxcal** was set too small for a particular problem, or may be due to a mistake in **funct**. If no mistake can be found in **funct**, restart nag_opt_one_var_deriv (e04bb) (preferably with the values of **a** and **b** given on exit from the previous call of nag_opt_one_var_deriv (e04bb)).

ifail $= -99$

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

ifail $= -399$

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

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

If $F(x)$ is δ-unimodal for some $\delta < Tol(x)$, where $Tol(x) = \mathbf{e1} \times |x| + \mathbf{e2}$, then, on exit, x approximates the minimum of $F(x)$ in the original interval $[a, b]$ with an error less than $3 \times Tol(x)$.

8 Further Comments

Timing depends on the behaviour of $F(x)$, the accuracy demanded and the length of the interval $[a, b]$. Unless $F(x)$ and $\frac{dF}{dx}$ can be evaluated very quickly, the run time will usually be dominated by the time spent in **funct**.

If $F(x)$ has more than one minimum in the original interval $[a, b]$, nag_opt_one_var_deriv (e04bb) will determine an approximation x (and improved bounds a and b) for one of the minima.

If nag_opt_one_var_deriv (e04bb) finds an x such that $F(x - \delta_1) > F(x) < F(x + \delta_2)$ for some $\delta_1, \delta_2 \geq Tol(x)$, the interval $[x - \delta_1, x + \delta_2]$ will be regarded as containing a minimum, even if $F(x)$ is less than $F(x - \delta_1)$ and $F(x + \delta_2)$ only due to rounding errors in the function. Therefore **funct** should be programmed to calculate $F(x)$ as accurately as possible, so that nag_opt_one_var_deriv (e04bb) will not be liable to find a spurious minimum. (For similar reasons, $\frac{dF}{dx}$ should be evaluated as accurately as possible.)

9 Example

A sketch of the function

$$F(x) = \frac{\sin x}{x}$$

shows that it has a minimum somewhere in the range $[3.5, 5.0]$. The following program shows how nag_opt_one_var_deriv (e04bb) can be used to obtain a good approximation to the position of a minimum.

9.1 Program Text

```
     function e04bb_example

fprintf('e04bb example results\n\n');

e1 = 0;
e2 = 0;
a = 3.5;
b = 5;
maxcal = nag_int(30);

[e1, e2, a, b, maxcal, x, f, g, user, ifail] = ...
e04bb( ...
       @funct, e1, e2, a, b, maxcal);

fprintf('The minimum lies in the interval  [%11.8f,%11.8f]\n', a, b);
fprintf('Estimated position of minimum, x = %11.8f\n', x);
fprintf('Function value at minimum,  f(x) = %7.4f\n', f);
fprintf('Gradient value at minimum, f''(x) = %8.1e\n', g);
```

```
fprintf('Number of function evaluations   = %2d\n', maxcal);

function [fc,gc,user] = funct(xc,user)
  fc=sin(xc)/xc;
  gc=(cos(xc)-fc)/xc;
```

9.2 Program Results

```
     e04bb example results

The minimum lies in the interval  [ 4.49340946, 4.49340952]
Estimated position of minimum, x =  4.49340946
Function value at minimum,  f(x) = -0.2172
Gradient value at minimum, f'(x) = -3.8e-16
Number of function evaluations   =  6
```
