NAG Toolbox

nag_opt_uncon_simplex (e04cb)

1 Purpose

nag_opt_uncon_simplex (e04cb) minimizes a general function $F(\mathbf{x})$ of n independent variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ by the Nelder and Mead simplex method (see Nelder and Mead (1965)). Derivatives of the function need not be supplied.

2 Syntax

```
[x, f, user, ifail] = nag_opt_uncon_simplex(x, tolf, tolx, funct, monit, maxcal, 'n', n, 'user', user)
[x, f, user, ifail] = e04cb(x, tolf, tolx, funct, monit, maxcal, 'n', n, 'user', user)
```

3 Description

nag_opt_uncon_simplex (e04cb) finds an approximation to a minimum of a function F of n variables. You must supply a function to calculate the value of F for any set of values of the variables.

The method is iterative. A simplex of $n + 1$ points is set up in the n-dimensional space of the variables (for example, in 2 dimensions the simplex is a triangle) under the assumption that the problem has been scaled so that the values of the independent variables at the minimum are of order unity. The starting point you have provided is the first vertex of the simplex, the remaining n vertices are generated by nag_opt_uncon_simplex (e04cb). The vertex of the simplex with the largest function value is reflected in the centre of gravity of the remaining vertices and the function value at this new point is compared with the remaining function values. Depending on the outcome of this test the new point is accepted or rejected, a further expansion move may be made, or a contraction may be carried out. See Nelder and Mead (1965) and Parkinson and Hutchinson (1972) for more details. When no further progress can be made the sides of the simplex are reduced in length and the method is repeated.

The method can be slow, but computational bottlenecks have been reduced following Singer and Singer (2004). However, nag_opt_uncon_simplex (e04cb) is robust, and therefore very useful for functions that are subject to inaccuracies.

There are the following options for successful termination of the method: based only on the function values at the vertices of the current simplex (see (1)); based only on a volume ratio between the current simplex and the initial one (see (2)); or based on which one of the previous two tests passes first. The volume test may be useful if F is discontinuous, while the function-value test should be sufficient on its own if F is continuous.

4 References

Nelder J A and Mead R (1965) A simplex method for function minimization *Comput. J.* **7** 308–313

Parkinson J M and Hutchinson D (1972) An investigation into the efficiency of variants of the simplex method *Numerical Methods for Nonlinear Optimization* (ed F A Lootsma) Academic Press

Singer S and Singer S (2004) Efficient implementation of the Nelder–Mead search algorithm *Appl. Num. Anal. Comp. Math.* **1(3)** 524–534

5 Parameters

5.1 Compulsory Input Parameters

1: **x**(**n**) – REAL (KIND=nag_wp) array

A guess at the position of the minimum. Note that the problem should be scaled so that the values of the **x**(i) are of order unity.

2: **tolf** – REAL (KIND=nag_wp)

The error tolerable in the function values, in the following sense. If f_i, for $i = 1, 2, \ldots, n+1$, are the individual function values at the vertices of the current simplex, and if f_m is the mean of these values, then you can request that nag_opt_uncon_simplex (e04cb) should terminate if

$$\sqrt{\frac{1}{n+1}\sum_{i=1}^{n+1}(f_i - f_m)^2} < \textbf{tolf}. \tag{1}$$

You may specify **tolf** $= 0$ if you wish to use only the termination criterion (2) on the spatial values: see the description of **tolx**.

Constraint: **tolf** must be greater than or equal to the ***machine precision*** (see Chapter X02), or if **tolf** equals zero then **tolx** must be greater than or equal to the ***machine precision***.

3: **tolx** – REAL (KIND=nag_wp)

The error tolerable in the spatial values, in the following sense. If LV denotes the 'linearized' volume of the current simplex, and if LV_{init} denotes the 'linearized' volume of the initial simplex, then you can request that nag_opt_uncon_simplex (e04cb) should terminate if

$$\frac{LV}{LV_{\text{init}}} < \textbf{tolx}. \tag{2}$$

You may specify **tolx** $= 0$ if you wish to use only the termination criterion (1) on function values: see the description of **tolf**.

Constraint: **tolx** must be greater than or equal to the ***machine precision*** (see Chapter X02), or if **tolx** equals zero then **tolf** must be greater than or equal to the ***machine precision***.

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

funct must evaluate the function F at a specified point. It should be tested separately before being used in conjunction with nag_opt_uncon_simplex (e04cb).

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

Input Parameters

1: **n** – INTEGER

n, the number of variables.

2: **xc**(**n**) – REAL (KIND=nag_wp) array

The point at which the function value is required.

3: **user** – INTEGER array

funct is called from nag_opt_uncon_simplex (e04cb) with the object supplied to nag_opt_uncon_simplex (e04cb).

Output Parameters

1: **fc** – REAL (KIND=nag_wp)

The value of the function F at the current point **x**.

2: **user** – INTEGER array

5: **monit** – SUBROUTINE, supplied by the NAG Library or the user.

monit may be used to monitor the optimization process. It is invoked once every iteration.

If no monitoring is required, **monit** may be string `nag_opt_uncon_simplex_dummy_monit (e04cbk)`

```
[user] = monit(fmin, fmax, sim, n, ncall, serror, vratio, user)
```

Input Parameters

1: **fmin** – REAL (KIND=nag_wp)

The smallest function value in the current simplex.

2: **fmax** – REAL (KIND=nag_wp)

The largest function value in the current simplex.

3: **sim**$(\mathbf{n}+\mathbf{1},\mathbf{n})$ – REAL (KIND=nag_wp) array

The $n+1$ position vectors of the current simplex.

4: **n** – INTEGER

n, the number of variables.

5: **ncall** – INTEGER

The number of times that **funct** has been called so far.

6: **serror** – REAL (KIND=nag_wp)

The current value of the standard deviation in function values used in termination test (1).

7: **vratio** – REAL (KIND=nag_wp)

The current value of the linearized volume ratio used in termination test (2).

8: **user** – INTEGER array

monit is called from nag_opt_uncon_simplex (e04cb) with the object supplied to nag_opt_uncon_simplex (e04cb).

Output Parameters

1: **user** – INTEGER array

6: **maxcal** – INTEGER

The maximum number of function evaluations to be allowed.

Constraint: **maxcal** ≥ 1.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the dimension of the array **x**.

n, the number of variables.

Constraint: $\mathbf{n} \geq 1$.

2: **user** – INTEGER array

user is not used by nag_opt_uncon_simplex (e04cb), but is passed to **funct** and **monit**. 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: $\mathbf{x}(\mathbf{n})$ – REAL (KIND=nag_wp) array

The value of **x** corresponding to the function value in **f**.

2: **f** – REAL (KIND=nag_wp)

The lowest function value found.

3: **user** – INTEGER array

4: **ifail** – INTEGER

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

6 Error Indicators and Warnings

Errors or warnings detected by the function:

$\mathbf{ifail} = 1$

Constraint: if $\mathbf{tolf} \neq 0.0$ and $\mathbf{tolx} \neq 0.0$ then both should be greater than or equal to the ***machine precision***.

Constraint: if $\mathbf{tolf} = 0.0$ then **tolx** is greater than or equal to the ***machine precision***.

Constraint: if $\mathbf{tolx} = 0.0$ then **tolf** is greater than or equal to the ***machine precision***.

Constraint: $\mathbf{maxcal} \geq 1$.

Constraint: $\mathbf{n} \geq 1$.

$\mathbf{ifail} = 2$

maxcal function evaluations have been completed without any other termination test passing. Check the coding of **funct** before increasing the value of **maxcal**.

$\mathbf{ifail} = -99$

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

$\mathbf{ifail} = -399$

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

$\mathbf{ifail} = -999$

Dynamic memory allocation failed.

7 Accuracy

On a successful exit the accuracy will be as defined by **tolf** or **tolx**, depending on which criterion was satisfied first.

8 Further Comments

Local workspace arrays of fixed lengths (depending on **n**) are allocated internally by nag_opt_uncon_simplex (e04cb). The total size of these arrays amounts to $\mathbf{n}^2 + 6\mathbf{n} + 2$ double elements.

The time taken by nag_opt_uncon_simplex (e04cb) depends on the number of variables, the behaviour of the function and the distance of the starting point from the minimum. Each iteration consists of 1 or 2 function evaluations unless the size of the simplex is reduced, in which case $n + 1$ function evaluations are required.

9 Example

This example finds a minimum of the function

$$F(x_1, x_2) = e^{x_1}\left(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1\right).$$

This example uses the initial point $(-1, 1)$ and we expect to reach the minimum at $(0.5, -1)$.

9.1 Program Text

```
function e04cb_example

fprintf('e04cb example results\n\n');

global print_info
print_info = false;

x0     = [-1; 1];
tolf   = sqrt(x02aj);
tolx   = sqrt(tolf);
maxcal = nag_int(100);
[x, f, user, ifail] = e04cb( ...
                            x0, tolf, tolx, @funct, @monit, maxcal);

fprintf('Minimum function value = %12.3e\n',f);
fprintf('Minumum location,    x = (%7.3f,%7.3f)\n',x);

fig1 = figure;
[xx,yy] = meshgrid([-1.5:0.1:1.5],[-2:0.1:2]);
z = exp(xx).*(4*xx.*(xx+yy)+2*yy.*(yy+1)+1);
[~,h] = contour(xx,yy,z);
h.LevelList = [0:0.1:1,1.3,1.6,2,2.6,3.2,4,5,6.5,8,10, ...
 13,16,20,25,32,40,50,65,80,100];
colormap(lines);
text(x(1),x(2),'*');
text(x0(1),x0(2),'X');
title('Contours of f: X - Initial Point, * - Local Minimum');

function [fc, user] = funct(n, xc, user)
  fc = exp(xc(1))*(4*xc(1)*(xc(1)+xc(2))+2*xc(2)*(xc(2) +1)+1);

function [user] = monit(fmin, fmax, sim, n, ncall, serror, vratio, user)
global print_info
  if (print_info)
    fprintf('\nNumber of function calls            = %10d\n', ncall);
    fprintf('Smallest function value             = %10.6f\n', fmin);
    fprintf('The simplex is\n');
    disp(sim);
    fprintf('Standard deviation in f(vertices)    = %10.6f\n', serror);
    fprintf('Current/initial simplex volume ratio = %10.6f\n', vratio);
  end
```

9.2 Program Results

```
     e04cb example results

Minimum function value =    1.789e-08
Minumum location,    x = (  0.500, -1.000)
```