nag_opt_one_var_deriv (e04bbc)

1. Purpose

nag_opt_one_var_deriv (e04bbc) 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. Specification

```c
#include <nag.h>
#include <nage04.h>

void nag_opt_one_var_deriv(void (*funct)(double xc, double *fc,
                                          double *gc, Nag_Comm *comm),
                           double e1, double e2, double *a, double *b,
                           Integer max_fun, double *x, double *f,
                           double *g, Nag_Comm *comm, NagError *fail)
```

3. Description

nag_opt_one_var_deriv is applicable to problems of the form:

\[ \text{Minimize } F(x) \text{ subject to } a \leq x \leq b \]

when the first derivative \(dF/dx\) can be calculated. nag_opt_one_var_deriv 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 quadratic-interpolation method described in Gill and Murray (1973).

The user must supply a function funct to evaluate \(F(x)\) and its first derivative. The parameters e1
and e2 together specify the accuracy

\[ \text{Tol}(x) = e1 \times |x| + e2 \]

to which the position of the minimum is required. Note that funct is never called at any 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 will normally
find one of the minima.

4. Parameters

funct

The function funct, supplied by the user, must calculate the values of \(F(x)\) and \(dF/dx\) at any
point \(x\) in \([a, b]\).

The specification of funct is:
void funct(double xc, double *fc, double *gc, Nag_Comm *comm)

xc
Input: \( x \), the point at which the values of \( F \) and \( dF/dx \) are required.

fc
Output: the value of the function \( F \) at the current point \( x \).

gc
Output: the value of the first derivative \( dF/dx \) at the current point \( x \).

comm
Pointer to structure of type Nag_Comm; the following members are relevant to funct.

first – Boolean
Input: will be set to TRUE on the first call to funct and FALSE for all subsequent calls.

nf – Integer
Input: the number of calls made to funct so far.

user – double *

iuser – Integer *

p – Pointer
The type Pointer will be void * with a C compiler that defines void * and char * otherwise.

Before calling nag_opt_one_var_deriv these pointers may be allocated memory by the user and initialized with various quantities for use by funct when called from nag_opt_one_var_deriv.

Note: funct should be tested separately before being used in conjunction with nag_opt_one_var_deriv.

e1
Input: 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.)

It is recommended that \( e1 \) should be no smaller than \( 2\epsilon \), and preferably not much less than \( \sqrt{\epsilon} \), where \( \epsilon \) is the machine precision.

If \( e1 \) is set to a value less than \( \epsilon \), its value is ignored and the default value of \( \sqrt{\epsilon} \) is used instead. In particular, the user may set \( e1 = 0.0 \) to ensure that the default value is used.

e2
Input: the absolute accuracy to which the position of a minimum is required. It is recommended that \( e2 \) should be no smaller than \( 2\epsilon \).

If \( e2 \) is set to a value less than \( \epsilon \), its value is ignored and the default value of \( \sqrt{\epsilon} \) is used instead. In particular, the user may set \( e2 = 0.0 \) to ensure that the default value is used.

a
Input: the lower bound \( a \) of the interval containing a minimum.
Output: an improved lower bound on the position of the minimum.

b
Input: the upper bound \( b \) of the interval containing a minimum.
Output: an improved upper bound on the position of the minimum.

Constraint: \( b > a + e2 \). Note that the value \( e2 = \sqrt{\epsilon} \) applies here if \( e2 < \epsilon \) on entry to nag_opt_one_var_deriv.

max_fun
Input: the maximum number of calls to funct which the user is prepared to allow.

The number of calls to funct actually made by nag_opt_one_var_deriv may be determined by supplying a non-NULL parameter comm (see below) and examining the structure member nf on exit.

Constraint: \( \text{max_fun} \geq 2 \). (Few problems will require more than 20 function calls.)
x  Output: the estimated position of the minimum.

f  Output: the value of $F$ at the final point $x$.

g  Output: the value of the first derivative $dF/dx$ at the final point $x$.

comm
Input/Output: structure containing pointers for communication to user-supplied functions; see the above description of funct for details. The number of times the function funct was called is returned in the member nf.

If the user does not need to make use of this communication feature, the null pointer NAGCOMM_NULL may be used in the call to nag_opt_one_var_deriv; comm will then be declared internally for use in calls to user-supplied functions.

fail
The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_\_REAL_\_ARG_\_GE
On entry, $a+e2 = \langle \text{value} \rangle$ while $b = \langle \text{value} \rangle$.
These parameters must satisfy $a+e2 < b$.

NE_INT_\_ARG_\_LT
On entry, max_fun must not be less than 2: max_fun = \langle \text{value} \rangle.

NW_MAX_FUN
The maximum number of function calls, \langle value \rangle, have been performed.
This may have happened simply because max_fun was set too small for a particular problem, or may be due to a mistake in the user-supplied function, funct. If no mistake can be found in funct, restart nag_opt_one_var_derivative (preferably with the values of a and b given on exit from the previous call to nag_opt_one_var_derivative).

6. 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 $dF/dx$ 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_derivative will determine an approximation $x$ (and improved bounds a and b) for one of the minima.

If nag_opt_one_var_derivative 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 user-supplied function. Therefore funct should be programmed to calculate $F(x)$ as accurately as possible, so that nag_opt_one_var_derivative will not be liable to find a spurious minimum. (For similar reasons, $dF/dx$ should be evaluated as accurately as possible.)

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

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

7. See Also
nag_opt_one_var_no_deriv (e04abc)
8. 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 example program below shows how `nag_opt_one_var_deriv` can be used to obtain a good approximation to the position of a minimum.

8.1. Program Text

```c
/* nag_opt_one_var_deriv(e04bbc) Example Program.
 */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <math.h>
#include <nage04.h>

static void funct(double xc, double *fc, double *gc, Nag_Comm *comm);
static void funct(double xc, *fc, *gc, Nag_Comm *comm);

main()
{
    double a, b;
    double e1, e2;
    double x, f, g;
    Integer max_fun;
    Nag_Comm comm;
    static NagError fail;
    Vprintf("e04bbc Example Program Results.\n\n");
    e1 = 0.0;
    e2 = 0.0;
    a = 3.5;
    b = 5.0;
    max_fun = 30;
    fail.print = TRUE;
e04bbc(funct, e1, e2, &a, &b, max_fun, &x, &f, &g, &comm, &fail);

    Vprintf("The minimum lies in the interval %7.5f to %7.5f.\n", a, b);
```

Vprintf("Its estimated position is %7.5f,\n", x);
Vprintf("where the function value is %9.4e\n", f);
Vprintf("and the gradient is %9.4e.\n", g);
Vprintf("%1ld function evaluations were required.\n", comm.nf);
exit(EXIT_SUCCESS);
}

8.2. Program Data

None.

8.3. Program Results

Example Program Results.

The minimum lies in the interval 4.49341 to 4.49341.
Its estimated position is 4.49341,
where the function value is -2.1723e-01
and the gradient is 4.3239e-16.
6 function evaluations were required.