NAG Toolbox

nag_roots_contfn_cntin_rcomm (c05ax)

1 Purpose

nag_roots_contfn_cntin_rcomm (c05ax) attempts to locate a zero of a continuous function using a continuation method based on a secant iteration. It uses reverse communication for evaluating the function.

2 Syntax

```
[x, c, ind, ifail] = nag_roots_contfn_cntin_rcomm(x, fx, tol, ir, c, ind, 'scal',
scal)
```

```
[x, c, ind, ifail] = c05ax(x, fx, tol, ir, c, ind, 'scal', scal)
```

3 Description

nag_roots_contfn_cntin_rcomm (c05ax) uses a modified version of an algorithm given in Swift and Lindfield (1978) to compute a zero α of a continuous function $f(x)$. The algorithm used is based on a continuation method in which a sequence of problems

$$f(x) - \theta_r f(x_0), \quad r = 0, 1, \ldots, m$$

are solved, where $1 = \theta_0 > \theta_1 > \cdots > \theta_m = 0$ (the value of m is determined as the algorithm proceeds) and where x_0 is your initial estimate for the zero of $f(x)$. For each θ_r the current problem is solved by a robust secant iteration using the solution from earlier problems to compute an initial estimate.

You must supply an error tolerance **tol**. **tol** is used directly to control the accuracy of solution of the final problem ($\theta_m = 0$) in the continuation method, and $\sqrt{\textbf{tol}}$ is used to control the accuracy in the intermediate problems ($\theta_1, \theta_2, \ldots, \theta_{m-1}$).

4 References

Swift A and Lindfield G R (1978) Comparison of a continuation method for the numerical solution of a single nonlinear equation *Comput. J.* **21** 359–362

5 Parameters

Note: this function uses **reverse communication.** Its use involves an initial entry, intermediate exits and re-entries, and a final exit, as indicated by the argument **ind**. Between intermediate exits and re-entries, **all arguments other than fx must remain unchanged**.

5.1 Compulsory Input Parameters

1: **x** – REAL (KIND=nag_wp)

On initial entry: an initial approximation to the zero.

2: **fx** – REAL (KIND=nag_wp)

On initial entry: if **ind** $= 1$, **fx** need not be set.

If **ind** $= -1$, **fx** must contain $f(\mathbf{x})$ for the initial value of **x**.

On intermediate re-entry: must contain $f(\mathbf{x})$ for the current value of **x**.

3: **tol** – REAL (KIND=nag_wp)

On initial entry: a value that controls the accuracy to which the zero is determined. **tol** is used in determining the convergence of the secant iteration used at each stage of the continuation process. It is used directly when solving the last problem ($\theta_m = 0$ in Section 3), and $\sqrt{\textbf{tol}}$ is used for the problem defined by θ_r, $r < m$. Convergence to the accuracy specified by **tol** is not guaranteed, and so you are recommended to find the zero using at least two values for **tol** to check the accuracy obtained.

Constraint: $\textbf{tol} > 0.0$.

4: **ir** – INTEGER

On initial entry: indicates the type of error test required, as follows. Solving the problem defined by θ_r, $1 \leq r \leq m$, involves computing a sequence of secant iterates $x_r^0, x_r^1, \dots$. This sequence will be considered to have converged only if:

for $\textbf{ir} = 0$,

$$\left|x_r^{(i+1)} - x_r^{(i)}\right| \leq eps \times \max\left(1.0, \left|x_r^{(i)}\right|\right),$$

for $\textbf{ir} = 1$,

$$\left|x_r^{(i+1)} - x_r^{(i)}\right| \leq eps,$$

for $\textbf{ir} = 2$,

$$\left|x_r^{(i+1)} - x_r^{(i)}\right| \leq eps \times \left|x_r^{(i)}\right|,$$

for some $i > 1$; here eps is either **tol** or $\sqrt{\textbf{tol}}$ as discussed above. Note that there are other subsidiary conditions (not given here) which must also be satisfied before the secant iteration is considered to have converged.

Constraint: $\textbf{ir} = 0$, 1 or 2.

5: **c(26)** – REAL (KIND=nag_wp) array

($\mathbf{c}(5)$ contains the current θ_r, this value may be useful in the event of an error exit.)

6: **ind** – INTEGER

On initial entry: must be set to 1 or -1.

$\textbf{ind} = 1$
: **fx** need not be set.

$\textbf{ind} = -1$
: **fx** must contain $f(\mathbf{x})$.

Constraint: on entry $\textbf{ind} = -1$, 1, 2, 3 or 4.

5.2 Optional Input Parameters

1: **scal** – REAL (KIND=nag_wp)

Suggested value: $\sqrt{\epsilon}$, where ϵ is the ***machine precision*** returned by nag_machine_precision (x02aj).

Default: $\sqrt{\textit{\textbf{machine precision}}}$

On initial entry: a factor for use in determining a significant approximation to the derivative of $f(x)$ at $x = x_0$, the initial value. A number of difference approximations to $f'(x_0)$ are calculated using

$$f'(x_0) \sim (f(x_0 + h) - f(x_0))/h$$

where $|h| < |\mathbf{scal}|$ and h has the same sign as **scal**. A significance (cancellation) check is made on each difference approximation and the approximation is rejected if insignificant.

Constraint: **scal** must be sufficiently large that $\mathbf{x} + \mathbf{scal} \neq \mathbf{x}$ on the computer.

5.3 Output Parameters

1: **x** – REAL (KIND=nag_wp)

On intermediate exit: the point at which f must be evaluated before re-entry to the function.

On final exit: the final approximation to the zero.

2: $\mathbf{c}(\mathbf{26})$ – REAL (KIND=nag_wp) array

3: **ind** – INTEGER

On intermediate exit: contains 2, 3 or 4. The calling program must evaluate f at **x**, storing the result in **fx**, and re-enter nag_roots_contfn_cntin_rcomm (c05ax) with all other arguments unchanged.

On final exit: contains 0.

4: **ifail** – INTEGER

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

6 Error Indicators and Warnings

Errors or warnings detected by the function:

$\mathbf{ifail} = 1$

On entry, $\mathbf{tol} \leq 0.0$,
or $\mathbf{ir} \neq 0$, 1 or 2.

$\mathbf{ifail} = 2$

The argument **ind** is incorrectly set on initial or intermediate entry.

$\mathbf{ifail} = 3$

scal is too small, or significant derivatives of f cannot be computed (this can happen when f is almost constant and nonzero, for any value of **scal**).

$\mathbf{ifail} = 4$

The current problem in the continuation sequence cannot be solved, see $\mathbf{c}(5)$ for the value of θ_r. The most likely explanation is that the current problem has no solution, either because the original problem had no solution or because the continuation path passes through a set of insoluble problems. This latter reason for failure should occur rarely, and not at all if the initial approximation to the zero is sufficiently close. Other possible explanations are that **tol** is too small and hence the accuracy requirement is too stringent, or that **tol** is too large and the initial approximation too poor, leading to successively worse intermediate solutions.

$\mathbf{ifail} = 5$

Continuation away from the initial point is not possible. This error exit will usually occur if the problem has not been properly posed or the error requirement is extremely stringent.

$\mathbf{ifail} = 6$

The final problem (with $\theta_m = 0$) cannot be solved. It is likely that too much accuracy has been requested, or that the zero is at $\alpha = 0$ and $\mathbf{ir} = 2$.

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

The accuracy of the approximation to the zero depends on **tol** and **ir**. In general decreasing **tol** will give more accurate results. Care must be exercised when using the relative error criterion (**ir** $= 2$).

If the zero is at $\mathbf{x} = 0$, or if the initial value of **x** and the zero bracket the point $\mathbf{x} = 0$, it is likely that an error exit with **ifail** $= 4$, 5 or 6 will occur.

It is possible to request too much or too little accuracy. Since it is not possible to achieve more than machine accuracy, a value of **tol** $\ll$ ***machine precision*** should not be input and may lead to an error exit with **ifail** $= 4$, 5 or 6. For the reasons discussed under **ifail** $= 4$ in Section 6, **tol** should not be taken too large, say no larger than **tol** $= 1.0\text{e}{-3}$.

8 Further Comments

For most problems, the time taken on each call to nag_roots_contfn_cntin_rcomm (c05ax) will be negligible compared with the time spent evaluating $f(x)$ between calls to nag_roots_contfn_cntin_rcomm (c05ax). However, the initial value of **x** and the choice of **tol** will clearly affect the timing. The closer that **x** is to the root, the less evaluations of f required. The effect of the choice of **tol** will not be large, in general, unless **tol** is very small, in which case the timing will increase.

9 Example

This example calculates a zero of $x - e^{-x}$ with initial approximation $x_0 = 1.0$, and **tol** $= 1.0\text{e}{-3}$ and $1.0\text{e}{-4}$.

9.1 Program Text

```
     function c05ax_example

fprintf('c05ax example results\n\n');

fx = 0;
c = zeros(26, 1);
for k=3:4
  x = 1;
  tol = 10^-k;
  ir = nag_int(0);
  ind = nag_int(1);

  while (ind ~= 0)
    [x, c, ind, ifail] = c05ax(x, fx, tol, ir, c, ind);
    fx = x - exp(-x);
  end

  if ifail == 4 || ifail ==6
    fprintf('FTol = %11.4e, final value = %11.4e, theta = %10.2e\n', tol, ...
             x, c(5));
  elseif ifail == 0
    fprintf('Tol is %11.4e, Root is %11.4e\n', tol, x);
  end
end
```

9.2 Program Results

```
     c05ax example results

Tol is  1.0000e-03, Root is  5.6715e-01
Tol is  1.0000e-04, Root is  5.6715e-01
```