NAG Toolbox for MATLAB

c05az

1 Purpose

c05az locates a simple zero of a continuous function on a given interval by a combination of the methods of linear interpolation, linear extrapolation and bisection. It uses reverse communication for evaluating the function.

2 Syntax

```
[x, y, fx, c, ind, ifail] = c05az(x, y, fx, tolx, c, ind, 'ir', ir)
```

3 Description

You must supply an initial interval $[\mathbf{x}, \mathbf{y}]$ containing a simple zero of the function $f(x)$ (the choice of **x** and **y** must be such that $f(\mathbf{x}) \times f(\mathbf{y}) \leq 0.0$). The function combines the methods of bisection, linear interpolation and linear extrapolation (see Dahlquist and Björck 1974), to find a sequence of sub-intervals of the initial interval such that the final interval $[\mathbf{x}, \mathbf{y}]$ contains the zero and $|\mathbf{x} - \mathbf{y}|$ is less than some tolerance specified by **tolx** and **ir** (see Section 5). In fact, since the intervals $[\mathbf{x}, \mathbf{y}]$ are determined only so that $f(\mathbf{x}) \times f(\mathbf{y}) \leq 0$, it is possible that the final interval may contain a discontinuity or a pole of f (violating the requirement that f be continuous). c05az checks if the sign change is likely to correspond to a pole of f and gives an error return in this case.

c05az returns to the calling program for each evaluation of $f(x)$. On each return you should set $\mathbf{fx} = f(\mathbf{x})$ and call c05az again.

The function is a modified version of procedure 'zeroin' given by Brent 1973.

4 References

Brent R P 1973 *Algorithms for Minimization Without Derivatives* Prentice–Hall

Bus J C P and Dekker T J 1975 Two efficient algorithms with guaranteed convergence for finding a zero of a function *ACM Trans. Math. Software* **1** 330–345

Dahlquist G and Björck *A* 1974 *Numerical Methods* Prentice–Hall

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 **parameter IND**. Between intermediate exits and re-entries, **all parameters other than fx must remain unchanged**.

5.1 Compulsory Input Parameters

1: **x – double scalar**

2: **y – double scalar**

On initial entry: **x** and **y** must define an initial interval containing the zero, such that $f(\mathbf{x}) \times f(\mathbf{y}) \leq 0$. It is not necessary that $\mathbf{x} < \mathbf{y}$.

3: **fx – double scalar**

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

If $\mathbf{ind} = -1$, **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**.

4: **tolx – double scalar**

On initial entry: the accuracy to which the zero is required. The type of error test is specified by **ir** (below).

Constraint: **tolx** > 0.

5: **c**(**17**) **– double array**

On initial entry: if **ind** $= 1$, no elements of **c** need be set.

If **ind** $= -1$, **c**(1) must contain $f(\mathbf{y})$, other elements of **c** need not be set.

6: **ind – int32 scalar**

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

ind $= 1$

fx and **c**(1) need not be set.

ind $= -1$

fx and **c**(1) must contain $f(\mathbf{x})$ and $f(\mathbf{y})$ respectively.

Constraint: on entry **ind** $= -1$, 1, 2, 3 or 4.

5.2 Optional Input Parameters

1: **ir – int32 scalar**

On initial entry: indicates the type of error test.

ir $= 0$

The test is: $|\mathbf{x} - \mathbf{y}| \leq 2.0 \times \mathbf{tolx} \times \max(1.0, |\mathbf{x}|)$.

ir $= 1$

The test is: $|\mathbf{x} - \mathbf{y}| \leq 2.0 \times \mathbf{tolx}$.

ir $= 2$

The test is: $|\mathbf{x} - \mathbf{y}| \leq 2.0 \times \mathbf{tolx} \times |\mathbf{x}|$.

Suggested value: **ir** $= 0$.

Default: 0

Constraint: **ir** $= 0$, 1 or 2.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **x – double scalar**

2: **y – double scalar**

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

On final exit: **x** and **y** define a smaller interval containing the zero, such that $f(\mathbf{x}) \times f(\mathbf{y}) \leq 0$, and $|\mathbf{x} - \mathbf{y}|$ satisfies the accuracy specified by **tolx** and **ir**, unless an error has occurred. If **ifail** $= 4$, **x** and **y** generally contain very good approximations to a pole; if **ifail** $= 5$, **x** and **y** generally contain very good approximations to the zero (see Section 6). If a point **x** is found such that $f(\mathbf{x}) = 0$, then on final exit **x** $=$ **y** (in this case there is no guarantee that **x** is a simple zero). In all cases, the value returned in **x** is the better approximation to the zero.

3: **fx – double scalar**

Is unchanged, except that after initial entry with **ind** $= -1$, **fx** contains the input value of **c**(1).

4: **c(17) – double array**

On final exit: is undefined.

5: **ind – int32 scalar**

On intermediate exit: contains 2, 3 or 4. The calling program must evaluate f at **x**, storing the result in **fx**, and re-enter c05az with all other parameters unchanged.

On final exit: contains 0.

6: **ifail – int32 scalar**

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

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, $f(\mathbf{x})$ and $f(\mathbf{y})$ have the same sign, with $f(\mathbf{x}) \neq 0.0$.

ifail $= 2$

On entry, **ind** $\neq -1$, 1, 2, 3 or 4.

ifail $= 3$

On entry, **tolx** ≤ 0.0,
or **ir** $\neq 0$, 1 or 2.

ifail $= 4$

An interval $[\mathbf{x}, \mathbf{y}]$ has been determined satisfying the error tolerance specified by **tolx** and **ir** and such that $f(\mathbf{x}) \times f(\mathbf{y}) \leq 0$. However, from observation of the values of f during the calculation of $[\mathbf{x}, \mathbf{y}]$, it seems that the interval $[\mathbf{x}, \mathbf{y}]$ contains a pole rather than a zero. Note that this error exit is not completely reliable: the error exit may be taken in extreme cases when $[\mathbf{x}, \mathbf{y}]$ contains a zero, or the error exit may not be taken when $[\mathbf{x}, \mathbf{y}]$ contains a pole. Both these cases occur most frequently when **tolx** is large.

ifail $= 5$

The tolerance **tolx** is too small for the problem being solved. This indicator is only set when the interval containing the zero has been reduced to one of relative length at most ϵ, the ***machine precision***, but the exit conditions described in Section 3 are not satisfied. It is unsafe to continue reducing the interval beyond this point, but the final values of **x** and **y** returned are accurate approximations to the zero.

7 Accuracy

The accuracy of the final value **x** as an approximation of the zero is determined by **tolx** and **ir** (see Section 5). A relative accuracy criterion (**ir** $= 2$) should not be used when the initial values **x** and **y** are of different orders of magnitude. In this case a change of origin of the independent variable may be appropriate. For example, if the initial interval $[\mathbf{x}, \mathbf{y}]$ is transformed linearly to the interval $[1, 2]$, then the zero can be determined to a precise number of figures using an absolute (**ir** $= 1$) or relative (**ir** $= 2$) error test and the effect of the transformation back to the original interval can also be determined. Except for the accuracy check, such a transformation has no effect on the calculation of the zero.

8 Further Comments

For most problems, the time taken on each call to c05az will be negligible compared with the time spent evaluating $f(x)$ between calls to c05az.

If the calculation terminates because $f(\mathbf{x}) = 0.0$, then on return **y** is set to **x**. (In fact, $\mathbf{y} = \mathbf{x}$ on return only in this case and, possibly, when $\mathbf{ifail} = 5$.) There is no guarantee that the value returned in **x** corresponds to a **simple** root and you should check whether it does.

One way to check this is to compute the derivative of f at the point **x**, preferably analytically, or, if this is not possible, numerically, perhaps by using a central difference estimate.

If $f'(\mathbf{x}) = 0.0$, then **x** must correspond to a multiple zero of f rather than a simple zero.

9 Example

```
x = 0;
y = 1;
fx = 0;
tolx = 1e-05;
c = zeros(17,1);
ind = int32(1);
while (ind ~= 0)
  [x, y, fx, c, ind, ifail] = c05az(x, y, fx, tolx, c, ind);
  fx = exp(-x) - x;
end
  [x,y]
```

```
ans =
    0.5671    0.5671
```