

NAG Library Routine Document

C05AZF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

C05AZF locates a simple zero of a continuous function on a given interval by a combination of the methods of nonlinear interpolation, linear extrapolation and bisection. It uses reverse communication for evaluating the function.

2 Specification

```
SUBROUTINE C05AZF(X, Y, FX, TOLX, IR, C, IND, IFAIL)
  INTEGER          IR, IND, IFAIL
  double precision X, Y, FX, TOLX, C(17)
```

3 Description

You must supply an initial interval $[X, Y]$ containing a simple zero of the function $f(x)$ (the choice of X and Y must be such that $f(X) \times f(Y) \leq 0.0$). The routine combines the methods of bisection, nonlinear 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 $[X, Y]$ contains the zero and $|X - Y|$ is less than some tolerance specified by $TOLX$ and IR (see Section 5). In fact, since the intervals $[X, Y]$ are determined only so that $f(X) \times f(Y) \leq 0.0$, it is possible that the final interval may contain a discontinuity or a pole of f (violating the requirement that f be continuous). C05AZF checks if the sign change is likely to correspond to a pole of f and gives an error return in this case.

A feature of the algorithm used by this routine is that unlike some other methods it guarantees convergence within about $\log_2[(b - a)/\delta]^2$ function evaluations, where δ is related to the parameter $TOLX$. See Brent (1973) for more details.

C05AZF returns to the calling program for each evaluation of $f(x)$. On each return you should set $FX = f(X)$ and call C05AZF again.

The routine 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 routine 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**.

- | | | |
|----|-------------------------------|---------------------|
| 1: | X – double precision | <i>Input/Output</i> |
| 2: | Y – double precision | <i>Input/Output</i> |

On initial entry: X and Y must define an initial interval containing the zero, such that $f(X) \times f(Y) \leq 0.0$. It is not necessary that $X < Y$.

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

On final exit: X and Y define a smaller interval containing the zero, such that $f(X) \times f(Y) \leq 0.0$, and $|X - Y|$ 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 $f(X) = 0.0$, 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 precision**

Input

On initial entry: if IND = 1, FX need not be set.

If IND = -1, FX must contain $f(X)$ for the initial value of X.

On intermediate re-entry: must contain $f(X)$ for the current value of X.

4: TOLX – **double precision**

Input

On initial entry: the accuracy to which the zero is required. The type of error test is specified by IR.

Constraint: TOLX > 0.

5: IR – INTEGER

Input

On initial entry: indicates the type of error test.

IR = 0

The test is: $|X - Y| \leq 2.0 \times \text{TOLX} \times \max(1.0, |X|)$.

IR = 1

The test is: $|X - Y| \leq 2.0 \times \text{TOLX}$.

IR = 2

The test is: $|X - Y| \leq 2.0 \times \text{TOLX} \times |X|$.

Suggested value: IR = 0.

Constraint: IR = 0, 1 or 2.

6: C(17) – **double precision** array

Input/Output

On initial entry: if IND = 1, no elements of C need be set.

If IND = -1, C(1) must contain $f(Y)$, other elements of C need not be set.

On final exit: is undefined.

7: IND – INTEGER

Input/Output

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 $f(X)$ and $f(Y)$ respectively.

On intermediate exit: contains 2, 3 or 4. The calling program must evaluate f at X, storing the result in FX, and re-enter C05AZF with all other parameters unchanged.

On final exit: contains 0.

Constraint: on entry IND = -1, 1, 2, 3 or 4.

8: IFAIL – INTEGER

Input/Output

On initial entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

On final exit: IFAIL = 0 unless the routine detects an error (see Section 6).

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, because for this routine the values of the output parameters may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1 . **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1 , explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry, $f(X)$ and $f(Y)$ have the same sign, with $f(X) \neq 0.0$.

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 $[X, Y]$ has been determined satisfying the error tolerance specified by TOLX and IR and such that $f(X) \times f(Y) \leq 0$. However, from observation of the values of f during the calculation of $[X, Y]$, it seems that the interval $[X, Y]$ 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 $[X, Y]$ contains a zero, or the error exit may not be taken when $[X, Y]$ 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 2ϵ , where ϵ is the *machine precision*, but the exit condition specified by IR is 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 $[X, Y]$ 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 C05AZF will be negligible compared with the time spent evaluating $f(x)$ between calls to C05AZF.

If the calculation terminates because $f(X) = 0.0$, then on return Y is set to X. (In fact, $Y = X$ on return only in this case and, possibly, when 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 $f'(X) = 0.0$, then X must correspond to a multiple zero of f rather than a simple zero.

9 Example

This example calculates a zero of $e^{-x} - x$ with an initial interval $[0, 1]$, $TOLX = 1.0D-5$ and a mixed error test.

9.1 Program Text

```
*      C05AZF Example Program Text
*      Mark 14 Revised. NAG Copyright 1989.
*      .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*      .. Local Scalars ..
      DOUBLE PRECISION FX, TOLX, X, Y
      INTEGER          IFAIL, IND, IR
*      .. Local Arrays ..
      DOUBLE PRECISION C(17)
*      .. External Functions ..
      DOUBLE PRECISION F
      EXTERNAL          F
*      .. External Subroutines ..
      EXTERNAL          C05AZF
*      .. Executable Statements ..
      WRITE (NOUT,*) 'C05AZF Example Program Results'
      WRITE (NOUT,*)
      WRITE (NOUT,*) ' Iterations'
      WRITE (NOUT,*)
      TOLX = 1.0D-5
      X = 0.0D0
      Y = 1.0D0
      IR = 0
      IFAIL = 1
      IND = 1
*
      20 CALL C05AZF(X,Y,FX,TOLX,IR,C,IND,IFAIL)
*
      IF (IND.NE.0) THEN
        FX = F(X)
        WRITE (NOUT,99999) ' X =', X, '    FX =', FX, '    IND =', IND
        GO TO 20
      ELSE IF (IFAIL.EQ.0) THEN
        WRITE (NOUT,*)
        WRITE (NOUT,*) ' Solution'
        WRITE (NOUT,*)
        WRITE (NOUT,99998) ' X =', X, '    Y =', Y
      ELSE
        IF (IFAIL.LT.0) WRITE (NOUT,*)
        WRITE (NOUT,99997) ' ** C05AZF returned with IFAIL = ', IFAIL
        IF (IFAIL.EQ.4 .OR. IFAIL.EQ.5) WRITE (NOUT,99998) ' X =', X,
+          ' Y =', Y
      END IF
*
      99999 FORMAT (1X,A,F8.5,A,E12.4,A,I2)
      99998 FORMAT (1X,A,F8.5,A,F8.5)
      99997 FORMAT (1X,A,I5)
      END
*
      DOUBLE PRECISION FUNCTION F(X)
*      .. Scalar Arguments ..
      DOUBLE PRECISION X
*      .. Intrinsic Functions ..
```

```
      INTRINSIC      EXP
*      .. Executable Statements ..
      F = EXP(-X) - X
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

C05AZF Example Program Results

Iterations

X = 0.00000	FX = 0.1000E+01	IND = 2
X = 1.00000	FX = -0.6321E+00	IND = 3
X = 0.61270	FX = -0.7081E-01	IND = 4
X = 0.56707	FX = 0.1154E-03	IND = 4
X = 0.56714	FX = -0.9448E-06	IND = 4
X = 0.56713	FX = 0.1473E-04	IND = 4
X = 0.56714	FX = -0.9448E-06	IND = 4

Solution

X = 0.56714 Y = 0.56713
