NAG Library Routine Document

C05AVF

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

C05AVF attempts to locate an interval containing a simple zero of a continuous function using a binary search. It uses reverse communication for evaluating the function.

2 Specification

```
SUBROUTINE C05AVF(X, FX, H, BOUNDL, BOUNDU, Y, C, IND, IFAIL)
INTEGER            IND, IFAIL
double precision   X, FX, H, BOUNDL, BOUNDU, Y, C(11)
```

3 Description

You must supply an initial point X and a step H. C05AVF attempts to locate a short interval $[\mathrm{X},\mathrm{Y}] \subset [\mathrm{BOUNDL},\mathrm{BOUNDU}]$ containing a simple zero of $f(x)$.

(On exit we may have $\mathrm{X} > \mathrm{Y}$; X is determined as the first point encountered in a binary search where the sign of $f(x)$ differs from the sign of $f(x)$ at the initial input point X.) The routine attempts to locate a zero of $f(x)$ using H, $0.1 \times \mathrm{H}$, $0.01 \times \mathrm{H}$ and $0.001 \times \mathrm{H}$ in turn as its basic step before quitting with an error exit if unsuccessful.

C05AVF returns to the calling program for each evaluation of $f(x)$. On each return you should set $\mathrm{FX} = f(\mathrm{X})$ and call C05AVF again.

4 References

None.

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*** *Input/Output*

On initial entry: the best available approximation to the zero.

Constraint: X must lie in the closed interval $[\mathrm{BOUNDL},\mathrm{BOUNDU}]$ (see below)

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

On final exit: contains one end of an interval containing the zero, the other end being in Y (below), unless an error has occurred. If $\mathrm{IFAIL} = 4$, X and Y are the end points of the largest interval searched. If a zero is located exactly, its value is returned in X (and in Y).

2: FX – ***double precision*** *Input*

On initial entry: if $\mathrm{IND} = 1$, FX need not be set.

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

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

3: H – ***double precision*** *Input/Output*

On initial entry: a basic step size which is used in the binary search for an interval containing a zero. The basic step sizes $\mathrm{H}, 0.1 \times \mathrm{H}$, $0.01 \times \mathrm{H}$ and $0.001 \times \mathrm{H}$ are used in turn when searching for the zero.

Constraint: either $\mathrm{X} + \mathrm{H}$ or $\mathrm{X} - \mathrm{H}$ must lie inside the closed interval $[\mathrm{BOUNDL}, \mathrm{BOUNDU}]$.

H must be sufficiently large that $\mathrm{X} + \mathrm{H} \neq \mathrm{X}$ on the computer.

On final exit: is undefined.

4: BOUNDL – ***double precision*** *Input*
5: BOUNDU – ***double precision*** *Input*

On initial entry: BOUNDL and BOUNDU must contain respectively lower and upper bounds for the interval of search for the zero.

Constraint: $\mathrm{BOUNDL} < \mathrm{BOUNDU}$.

6: Y – ***double precision*** *Input/Output*

On initial entry: need not be set.

On final exit: contains the closest point found to the final value of X, such that $f(\mathrm{X}) \times f(\mathrm{Y}) \leq 0.0$. If a value X is found such that $f(\mathrm{X}) = 0$, then $\mathrm{Y} = \mathrm{X}$. On final exit with $\mathrm{IFAIL} = 4$, X and Y are the end points of the largest interval searched.

7: $\mathrm{C}(11)$ – ***double precision*** array *Communication Array*

On initial entry: need not be set.

On final exit: if $\mathrm{IFAIL} = 0$ or 4, $\mathrm{C}(1)$ contains $f(\mathrm{Y})$.

8: IND – INTEGER *Input/Output*

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

$\mathrm{IND} = 1$
FX need not be set.

$\mathrm{IND} = -1$
FX must contain $f(\mathrm{X})$.

On intermediate exit: contains 2 or 3. The calling program must evaluate f at X, storing the result in FX, and re-enter C05AVF with all other parameters unchanged.

On final exit: contains 0.

Constraint: on entry $\mathrm{IND} = -1$, 1, 2 or 3.

9: 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: $\mathrm{IFAIL} = 0$ 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 $\mathrm{IFAIL} \neq 0$ 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, $\mathrm{BOUNDU} \leq \mathrm{BOUNDL}$,
or $\mathrm{X} \notin [\mathrm{BOUNDL}, \mathrm{BOUNDU}]$,
or both $\mathrm{X} + \mathrm{H}$ and $\mathrm{X} - \mathrm{H} \notin [\mathrm{BOUNDL}, \mathrm{BOUNDU}]$.

IFAIL $= 2$

On initial entry, H is too small to be used to perturb the initial value of X in the search.

IFAIL $= 3$

The parameter IND is incorrectly set on initial or intermediate entry.

IFAIL $= 4$

The routine has been unable to determine an interval containing a simple zero starting from the initial value of X and using the step H. If you have prior knowledge that a simple zero lies in the interval $[\mathrm{BOUNDL}, \mathrm{BOUNDU}]$, you should vary X and H in an attempt to find it. (See also Section 8.)

7 Accuracy

C05AVF is not intended to be used to obtain accurate approximations to the zero of $f(x)$ but rather to locate an interval containing a zero. This interval can then be used as input to an accurate rootfinder such as C05ADF or C05AZF. The size of the interval determined depends somewhat unpredictably on the choice of X and H. The closer X is to the root and the **smaller** the initial value of H, then, in general, the smaller (more accurate) the interval determined; however, the accuracy of this statement depends to some extent on the behaviour of $f(x)$ near $x = \mathrm{X}$ and on the size of H.

8 Further Comments

For most problems, the time taken on each call to C05AVF will be negligible compared with the time spent evaluating $f(x)$ between calls to C05AVF. However, the initial value of X and H will clearly affect the timing. The closer that X is to the root, and the **larger** the initial value of H then the less time taken. (However taking a large H can affect the accuracy and reliability of the routine, see below.)

You are expected to choose BOUNDL and BOUNDU as physically (or mathematically) realistic limits on the interval of search. For example, it may be known, from physical arguments, that no zero of $f(x)$ of interest will lie outside $[\mathrm{BOUNDL}, \mathrm{BOUNDU}]$. Alternatively, $f(x)$ may be more expensive to evaluate for some values of X than for others and such expensive evaluations can sometimes be avoided by careful choice of BOUNDL and BOUNDU.

The choice of BOUNDL and BOUNDU affects the search only in that these values provide physical limitations on the search values and that the search is terminated if it seems, from the available information about $f(x)$, that the zero lies outside $[\mathrm{BOUNDL}, \mathrm{BOUNDU}]$. In this case (IFAIL $= 4$ on exit), only one of f(BOUNDL) and f(BOUNDU) may have been evaluated and a zero close to the other end of the interval could be missed. The actual interval searched is returned in the parameters X and Y and you can call C05AVF again to search the remainder of the original interval.

Though C05AVF is intended primarily for determining an interval containing a zero of $f(x)$, it may be used to shorten a known interval. This could be useful if, for example, a large interval containing the zero is known and it is also known that the root lies close to one end of the interval; by setting X to this end of the interval and H small, a short interval will usually be determined. However, it is worth noting that once any interval containing a zero has been determined, a call to C05AZF will usually be the most efficient way to calculate an interval of specified length containing the zero. To assist in this determination, the information in X, Y, FX and C(1) on successful exit from C05AVF is in the correct form for a call to routine C05AZF with IND $= -1$.

If the calculation terminates because $f(\mathrm{X}) = 0.0$, then on return Y is set to X. (In fact, $\mathrm{Y} = \mathrm{X}$ on return only in this case.) In this case, there is no guarantee that the value in X corresponds to a **simple** zero 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 $f'(\mathrm{X}) = 0.0$, then X must correspond to a multiple zero of f rather than a simple zero.

9 Example

This example finds a sub-interval of $[0.0, 4.0]$ containing a simple zero of $x^2 - 3x + 2$. The zero nearest to 3.0 is required and so we set $\mathrm{X} = 3.0$ initially.

9.1 Program Text

```
*     C05AVF Example Program Text
*     Mark 14 Revised. NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Local Scalars ..
      DOUBLE PRECISION BOUNDL, BOUNDU, FX, H, X, Y
      INTEGER          IFAIL, IND
*     .. Local Arrays ..
      DOUBLE PRECISION C(11)
*     .. External Subroutines ..
      EXTERNAL         C05AVF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'C05AVF Example Program Results'
      WRITE (NOUT,*)
      X = 3.0D0
      H = 0.1D0
      BOUNDL = 0.0D0
      BOUNDU = 4.0D0
      IFAIL = 1
      IND = 1
*
   20 CALL C05AVF(X,FX,H,BOUNDL,BOUNDU,Y,C,IND,IFAIL)
*
      IF (IND.NE.0) THEN
         FX = X*X - 3.0D0*X + 2.0D0
         GO TO 20
      ELSE IF (IFAIL.EQ.0) THEN
         WRITE (NOUT,*) 'Interval containing root is [X,Y], where'
         WRITE (NOUT,99999) 'X =', X, '   Y =', Y
         WRITE (NOUT,*) 'Values of f at X and Y are'
         WRITE (NOUT,99998) 'f(X) =', FX, '   f(Y) =', C(1)
      ELSE
         WRITE (NOUT,*)
         WRITE (NOUT,99997) ' ** C05AVF returned with IFAIL = ', IFAIL
      END IF
*
99999 FORMAT (1X,A,F12.4,A,F12.4)
99998 FORMAT (1X,2(A,F12.2))
99997 FORMAT (1X,A,I5)
      END
```

9.2 Program Data

None.

9.3 Program Results

```
C05AVF Example Program Results

Interval containing root is [X,Y], where
X =      1.7000   Y =      2.5000
Values of f at X and Y are
f(X) =       -0.21   f(Y) =        0.75
```