D04AAF – NAG Fortran Library Routine Document

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

D04AAF calculates a set of derivatives (up to order 14) of a function of one real variable at a point, together with a corresponding set of error estimates, using an extension of the Neville algorithm.

2 Specification

```
SUBROUTINE D04AAF(XVAL, NDER, HBASE, DER, EREST, FUN, IFAIL)
INTEGER           NDER, IFAIL
real              XVAL, HBASE, DER(14), EREST(14), FUN
EXTERNAL          FUN
```

3 Description

This routine provides a set of approximations:

$$\mathrm{DER}(j), \quad j = 1, 2, \ldots, n$$

to the derivatives:

$$f^{(j)}(x_0), \quad j = 1, 2, \ldots, n$$

of a real valued function $f(x)$ at a real abscissa x_0, together with a set of error estimates:

$$\mathrm{EREST}(j), \quad j = 1, 2, \ldots, n$$

which hopefully satisfy:

$$|\mathrm{DER}(j) - f^{(j)}(x_0)| < \mathrm{EREST}(j), \quad j = 1, 2, \ldots, n.$$

The user provides the value of x_0, a value of n (which is reduced to 14 should it exceed 14) a function (sub)program which evaluates $f(x)$ for all real x, and a step length h. The results DER(j) and EREST(j) are based on 21 function values:

$$f(x_0), f(x_0 \pm (2i-1)h), \quad i = 1, 2, \ldots, 10.$$

Internally the routine calculates the odd order derivatives and the even order derivatives separately. There is a user option for restricting the calculation to only odd (or even) order derivatives. For each derivative the routine employs an extension of the Neville Algorithm (see Lyness and Moler [2]) to obtain a selection of approximations.

For example, for odd derivatives, based on 20 function values, the routine calculates a set of numbers:

$$T_{k,p,s}, \quad p = s, s+1, \ldots, 6, \quad k = 0, 1, \ldots, 9-p$$

each of which is an approximation to $f^{(2s+1)}(x_0)/(2s+1)!$. A specific approximation $T_{k,p,s}$ is of polynomial degree $2p+2$ and is based on polynomial interpolation using function values $f(x_0 \pm (2i-1)h)$, $i = k, k+1, \ldots, k+p$. In the absence of round-off error, the better approximations would be associated with the larger values of p and of k. However, round-off error in function values has an increasingly contaminating effect for successively larger values of p. This routine proceeds to make a judicious choice between all the approximations in the following way.

For a specified value of s, let:

$$\begin{aligned} R_p &= U_p - L_p, && p = s, s+1, \ldots, 6 \\ \text{where} \quad U_p &= \max_k (T_{k,p,s}), && k = 0, 1, \ldots, 9-p \\ L_p &= \min_k (T_{k,p,s}), && k = 0, 1, \ldots 9-p \end{aligned}$$

and let $\bar{p}$ be such that $R_{\bar{p}} = \min_p(R_p)$ for $p = s, s+1, \ldots, 6$.

The routine returns:

$$\mathrm{DER}(2s+1) = \frac{1}{8-\bar{p}} \times \left\{ \sum_{k=0}^{9-\bar{p}} T_{k,\bar{p},s} - U_{\bar{p}} - L_{\bar{p}} \right\} \times (2s+1)!$$

and

$$\mathrm{EREST}(2s+1) = R_{\bar{p}} \times (2s+1)! \times K_{2s} + 1$$

where K_j is a safety factor which has been assigned the values:

$K_j = 1$	$j \leq 9$
$K_j = 1.5$	$j = 10, 11$
$K_j = 2$	$j \geq 12$

on the basis of performance statistics.

The even order derivatives are calculated in a precisely analogous manner.

4 References

[1] Lyness J N and Moler C B (1966) van der Monde systems and numerical differentiation *Numer. Math.* **8** 458–464

[2] Lyness J N and Moler C B (1969) Generalised Romberg methods for integrals of derivatives *Numer. Math.* **14** 1–14

5 Parameters

1: XVAL — ***real*** *Input*

On entry: the point at which the derivatives are required, x_0.

2: NDER — INTEGER *Input*

On entry: NDER must be set so that its absolute value is the highest order derivative required. If NDER > 0, all derivatives up to order min(NDER,14) are calculated. If NDER < 0 and NDER is even, only even order derivatives up to order min($-$NDER,14) are calculated. If NDER < 0 and NDER is odd, only odd order derivatives up to order min($-$NDER,13) are calculated.

3: HBASE — ***real*** *Input*

On entry: the initial step length which may be positive or negative.

(If set to zero the routine does not proceed with any calculation, but sets the error flag IFAIL and returns to the (sub)program from which D04AAF is called.) For advice on the choice of HBASE see Section 8.

4: DER(14) — ***real*** array *Output*

On exit: an approximation to the jth derivative of $f(x)$ at $x =$ XVAL, so long as the jth derivative is one of those requested by the user when specifying NDER. For other values of j, DER(j) is unused.

5: EREST(14) — ***real*** array *Output*

On exit: an estimate of the absolute error in the corresponding result DER(j) so long as the jth derivative is one of those requested by the user when specifying NDER. The sign of EREST(j) is positive unless the result DER(j) is questionable. It is set negative when $|\mathrm{DER}(j)| < |\mathrm{EREST}(j)|$ or when for some other reason there is doubt about the validity of the result DER(j) (see Section 6). For other values of j, EREST(j) is unused.

6: FUN — ***real*** FUNCTION, supplied by the user. *External Procedure*

FUN must evaluate the function $f(x)$ at a specified point.

Its specification is:

```
real FUNCTION FUN(X)
real               X
```

1: X — ***real*** *Input*

On entry: the value of the argument x.

For users with equally spaced tabular data, the following information may be useful:

(i) in any call of D04AAF the only values of X that will be required are X = XVAL and X = XVAL $\pm$ $(2j-1)$HBASE, for j = 1,2,. . .,10; and

(ii) FUN(XVAL) is always called, but it is disregarded when only odd order derivatives are required.

FUN must be declared as EXTERNAL in the (sub)program from which D04AAF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

7: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

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

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NDER = 0,
or HBASE = 0.

If IFAIL has a value zero on exit then D04AAF has terminated successfully, but before any use is made of a derivative DER(j) the value of EREST(j) must be checked.

7 Accuracy

The accuracy of the results is problem dependent. An estimate of the accuracy of each result DER(j) is returned in EREST(j) (see Section 3, Section 5 and Section 8).

A basic feature of any floating-point routine for numerical differentiation based on real function values on the real axis is that successively higher order derivative approximations are successively less accurate. It is expected that in most cases DER(14) will be unusable. As an aid to this process, the sign of EREST(j) is set negative when the estimated absolute error is greater than the approximate derivative itself, i.e., when the approximate derivative may be so inaccurate that it may even have the wrong sign. It is also set negative in some other cases when information available to the routine indicates that the corresponding value of DER(j) is questionable.

The actual values in EREST depend on the accuracy of the function values, the properties of the machine arithmetic, the analytic properties of the function being differentiated and the user-provided step length HBASE (see Section 8). The only hard and fast rule is that for a given FUN(X) and HBASE, the values of EREST(j) increase with increasing j. The limit of 14 is dictated by experience. Only very rarely can one obtain meaningful approximations for higher order derivatives on conventional machines.

8 Further Comments

The time taken by the routine depends on the time spent for function evaluations. Otherwise the time is roughly equivalent to that required to evaluate the function 21 times and calculate a finite difference table having about 200 entries in total.

The results depend very critically on the choice of the user-provided step length HBASE. The overall accuracy is diminished as HBASE becomes small (because of the effect of round-off error) and as HBASE becomes large (because the discretisation error also becomes large). If the routine is used four or five times with different values of HBASE one can find a reasonably good value. A process in which the value of HBASE is successively halved (or doubled) is usually quite effective. Experience has shown that in cases in which the Taylor series for FUN(X) about XVAL has a finite radius of convergence R, the choices of HBASE > R/21 are not likely to lead to good results. In this case some function values lie outside the circle of convergence.

9 Example

This example evaluates the odd-order derivatives of the function:

$$f(x) = \frac{1}{2}e^{2x} - 1$$

up to order 7 at the point $x = \frac{1}{2}$. Several different values of HBASE are used, to illustrate that:

(i) extreme choices of HBASE, either too large or too small, yield poor results;
(ii) the quality of these results is adequately indicated by the values of EREST.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     D04AAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Local Scalars ..
      real             HBASE, XVAL
      INTEGER          I, IFAIL, J, K, L, NDER
*     .. Local Arrays ..
      real             DER(14), EREST(14)
*     .. External Functions ..
      real             FUN
      EXTERNAL         FUN
*     .. External Subroutines ..
      EXTERNAL         D04AAF
*     .. Intrinsic Functions ..
      INTRINSIC        ABS
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D04AAF Example Program Results'
      WRITE (NOUT,*)
      WRITE (NOUT,*)
     +'Four separate runs to calculate the first four odd order derivati
     +ves of'
      WRITE (NOUT,*) '   FUN(X) = 0.5*exp(2.0*X-1.0) at X = 0.5.'
      WRITE (NOUT,*) 'The exact results are 1, 4, 16 and 64'
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Input parameters common to all four runs'
      WRITE (NOUT,*) '   XVAL = 0.5    NDER = -7    IFAIL = 0'
```

```
      WRITE (NOUT,*)
      HBASE = 0.5e0
      NDER = -7
      L = ABS(NDER)
      IF (NDER.GE.0) THEN
         J = 1
      ELSE
         J = 2
      END IF
      XVAL = 0.5e0
      DO 40 K = 1, 4
         IFAIL = 0
*
         CALL D04AAF(XVAL,NDER,HBASE,DER,EREST,FUN,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'with step length', HBASE,
     +     '  the results are'
         WRITE (NOUT,*) 'Order        Derivative       Error estimate'
         DO 20 I = 1, L, J
            WRITE (NOUT,99998) I, DER(I), EREST(I)
   20    CONTINUE
         HBASE = HBASE*0.1e0
   40 CONTINUE
      STOP
*
99999 FORMAT (1X,A,F9.4,A)
99998 FORMAT (1X,I2,2e21.4)
      END
*
      real FUNCTION FUN(X)
*     .. Scalar Arguments ..
      real              X
*     .. Intrinsic Functions ..
      INTRINSIC         EXP
*     .. Executable Statements ..
      FUN = 0.5e0*EXP(2.0e0*X-1.0e0)
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
D04AAF Example Program Results

Four separate runs to calculate the first four odd order derivatives of
   FUN(X) = 0.5*exp(2.0*X-1.0) at X = 0.5.
The exact results are 1, 4, 16 and 64

Input parameters common to all four runs
   XVAL = 0.5    NDER = -7    IFAIL = 0
```

```
with step length   0.5000  the results are
Order        Derivative       Error estimate
 1           0.1392E+04          -0.1073E+06
 3          -0.3139E+04          -0.1438E+06
 5           0.8762E+04          -0.2479E+06
 7          -0.2475E+05          -0.4484E+06

with step length   0.0500  the results are
Order        Derivative       Error estimate
 1           0.1000E+01           0.1529E-10
 3           0.4000E+01           0.2112E-08
 5           0.1600E+02           0.3815E-06
 7           0.6400E+02           0.7384E-04

with step length   0.0050  the results are
Order        Derivative       Error estimate
 1           0.1000E+01           0.1277E-13
 3           0.4000E+01           0.4190E-09
 5           0.1600E+02           0.1463E-04
 7           0.6404E+02           0.2973E+00

with step length   0.0005  the results are
Order        Derivative       Error estimate
 1           0.1000E+01           0.1427E-12
 3           0.4000E+01           0.3087E-06
 5           0.1599E+02           0.6331E+00
 7           0.3825E+05          -0.1964E+07
```