NAG Library Routine Document

D04BAF

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

D04BAF calculates a set of derivatives (up to order 14) of a function at a point with respect to a single variable. A corresponding set of error estimates is also returned. Derivatives are calculated using an extension of the Neville algorithm. This routine differs from D04AAF, in that the abscissae and corresponding function values must be calculated before this routine is called. The abscissae may be generated using D04BBF.

2 Specification

```
SUBROUTINE D04BAF (XVAL, FVAL, DER, EREST, IFAIL)
INTEGER            IFAIL
REAL (KIND=nag_wp) XVAL(21), FVAL(21), DER(14), EREST(14)
```

3 Description

D04BAF provides a set of approximations:

$$\mathrm{DER}(j), \quad j = 1, 2, \ldots, 14$$

to the derivatives:

$$f^{(j)}(x_0), \quad j = 1, 2, \ldots, 14$$

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

$$\mathrm{EREST}(j), \quad j = 1, 2, \ldots, 14$$

which hopefully satisfy:

$$\left|\mathrm{DER}(j) - f^{(j)}(x_0)\right| < \mathrm{EREST}(j), \quad j = 1, 2, \ldots, 14.$$

The results $\mathrm{DER}(j)$ and $\mathrm{EREST}(j)$ are based on 21 function values:

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

The abscissae x and the corresponding function values $f(x)$ should be passed into D04BAF as the vectors XVAL and FVAL respectively. The step size h is derived from the abscissae in XVAL. See Section 9 for a discussion of how the derived value of h may affect the results of D04BAF. The order in which the abscissae and function values are stored in XVAL and FVAL is irrelevant, provided that the function value at any given index corresponds to the value of the abscissa at the same index. Abscissae may be automatically generated using D04BBF if desired. For each derivative D04BAF employs an extension of the Neville Algorithm (see Lyness and Moler (1969)) to obtain a selection of approximations.

For example, for odd derivatives, this 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 $f(x_0 \pm (2i-1)h)$, for $k = k, \ldots, k+p$. 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:

$$R_p = U_p - L_p, \quad p = s, s+1, \ldots, 6$$

where $U_p = \max_k \left(T_{k,p,s}\right)$ and $L_p = \min_k \left(T_{k,p,s}\right)$, for $k = 0, 1, \ldots, 9-p$, and let $\bar{p}$ be such that $R_{\bar{p}} = \min_p \left(R_p\right)$, for $p = s, \ldots, 6$.

This routine returns:

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

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:

$$\begin{array}{ll} K_j = 1, & j \leq 9 \\ K_j = 1.5, & j = 10, 11 \\ K_j = 2 & j \geq 12, \end{array}$$

on the basis of performance statistics.

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

4 References

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

5 Parameters

1: XVAL(21) – REAL (KIND=nag_wp) array *Input*

On entry: the abscissae at which the function has been evaluated, as described in Section 3. These can be generated by calling D04BBF. The order of the abscissae is irrelevant.

Constraint: the values in XVAL must span the set $\{x_0, x_0 \pm (2j-1)h\}$, for $j = 1, 2, \ldots, 10$.

2: FVAL(21) – REAL (KIND=nag_wp) array *Input*

On entry: FVAL(j) must contain the function value at XVAL(j), for $j = 1, 2, \ldots, 21$.

3: DER(14) – REAL (KIND=nag_wp) array *Output*

On exit: the 14 derivative estimates.

4: EREST(14) – REAL (KIND=nag_wp) array *Output*

On exit: the 14 error estimates for the derivatives.

5: IFAIL – INTEGER *Input/Output*

On 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.

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, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

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, the values of XVAL are not correctly spaced.
Derived $h = \langle value \rangle$.

The derived h is below tolerance.
Derived $h > \langle value \rangle$ is required. Derived $h = \langle value \rangle$.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

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 Sections 3, 5 and 9).

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 D04BAF 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 step length h (see Section 9). The only hard and fast rule is that for a given objective function and h, 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 Parallelism and Performance

Not applicable.

9 Further Comments

The results depend very critically on the choice of the step length h. The overall accuracy is diminished as h becomes small (because of the effect of round-off error) and as h becomes large (because the discretization error also becomes large). If this routine is used four or five times with different values of h one can find a reasonably good value. A process in which the value of h is successively halved (or doubled) is usually quite effective. Experience has shown that in cases in which the Taylor series for the objective function about x_0 has a finite radius of convergence R, the choices of $h > R/19$ are not likely to lead to good results. In this case some function values lie outside the circle of convergence.

As mentioned, the order of the abscissae in XVAL does not matter, provided the corresponding values of FVAL are ordered identically. If the abscissae are generated by D04BBF, then they will be in ascending order. In particular, the target abscissa x_0 will be stored in XVAL(11).

10 Example

This example evaluates the derivatives of the polygamma function, calculated using S14AEF, and compares the first 3 derivatives calculated to those found using S14AEF.

10.1 Program Text

```
    Program d04bafe

!     D04BAF Example Program Text
!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: d04baf, d04bbf, nag_wp, s14aef, x04cbf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter    :: x_0 = 0.05_nag_wp
      Integer, Parameter               :: indent = 0, ncols = 80, nout = 6,    &
                                          n_der_comp = 3, n_display = 3,       &
                                          n_hbase = 4, zeroth = 0
      Character (1), Parameter         :: chlabel = 'C', diag = 'N', form =    &
                                          ' ', matrix = 'G', nolabel = 'N'
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: hbase
      Integer                          :: ifail, j, k
      Character (50)                   :: title
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: actder(n_display), der(14),          &
                                          der_comp(n_hbase,n_der_comp,14),     &
                                          erest(14), fval(21), xval(21)
      Character (10)                   :: clabs(n_der_comp), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'D04BAF Example Program Results'
      Write (nout,*)
      Write (nout,*) ' Find the derivatives of the polygamma (psi) function'
      Write (nout,*) ' using function values generated by S14AEF.'
      Write (nout,*)
      Write (nout,*) ' Demonstrate the effect of successively reducing HBASE.'
      Write (nout,*)
!     Select an initial separation distance HBASE.
      hbase = 0.0025_nag_wp

!     Compute the actual derivatives at target location x_0 using s14aef for
!     comparison.
      Do j = 1, n_display
        ifail = 0
        actder(j) = s14aef(x_0,j,ifail)
      End Do

!     Attempt N_HBASE approximations, reducing HBASE by factor 0.1 each time.
      Do j = 1, n_hbase
```

```
!       Generate the abscissa XVAL using D04BBF
        Call d04bbf(x_0,hbase,xval)

!       Calculate the corresponding objective function values.
        Do k = 1, 21
          ifail = 0
          fval(k) = s14aef(xval(k),zeroth,ifail)
        End Do

!       Call D04BAF to calculate the derivative estimates
        ifail = 0
        Call d04baf(xval,fval,der,erest,ifail)

!       Store results in DER_COMP
        der_comp(j,1,1:14) = hbase
        der_comp(j,2,1:14) = der(1:14)
        der_comp(j,3,1:14) = erest(1:14)

!       Decrease hbase for next loop
        hbase = hbase*0.1_nag_wp
      End Do

!     Display Results for first N_DISPLAY derivatives

      Do j = 1, n_display
        Write (nout,99996) j, actder(j)
        Write (clabs(1),99997) 'hbase     '
        Write (clabs(2),99998) 'DER', j, '   '
        Write (clabs(3),99999) 'EREST', j
        Write (title,99999) ' Derivative and error estimates for derivative ', &
          j
        Flush (nout)

!       Use X04CBF to display the matrix
        ifail = 0
        Call x04cbf(matrix,diag,n_hbase,n_der_comp,der_comp(1,1,j),n_hbase, &
          form,title,nolabel,rlabs,chlabel,clabs,ncols,indent,ifail)
        Write (nout,*)
      End Do

99999 Format (A,'(',I1,')')
99998 Format (A,'(',I1,')',A)
99997 Format (A)
99996 Format (1X,' Derivative (',I1,') calculated using S14AEF :',1X,Es11.4)
    End Program d04bafe
```

10.2 Program Data

None.

10.3 Program Results

```
 D04BAF Example Program Results

  Find the derivatives of the polygamma (psi) function
  using function values generated by S14AEF.

  Demonstrate the effect of successively reducing HBASE.

  Derivative (1) calculated using S14AEF :  4.0153E+02
  Derivative and error estimates for derivative (1)
          hbase       DER(1)     EREST(1)
     2.5000E-03   4.0204E+02   1.3940E+02
     2.5000E-04   4.0153E+02   4.9170E-11
     2.5000E-05   4.0153E+02   2.1799E-10
     2.5000E-06   4.0153E+02   1.1826E-09

  Derivative (2) calculated using S14AEF : -1.6002E+04
  Derivative and error estimates for derivative (2)
          hbase       DER(2)     EREST(2)
```

```
    2.5000E-03  -1.6022E+04   5.5760E+03
    2.5000E-04  -1.6002E+04   1.2831E-07
    2.5000E-05  -1.6002E+04   6.0543E-06
    2.5000E-06  -1.6002E+04   9.5762E-04

 Derivative (3) calculated using S14AEF :  9.6001E+05
 Derivative and error estimates for derivative (3)
        hbase       DER(3)     EREST(3)
    2.5000E-03   9.1465E+05  -7.3750E+06
    2.5000E-04   9.6001E+05   2.3718E-04
    2.5000E-05   9.6001E+05   4.2253E-02
    2.5000E-06   9.6001E+05   5.9679E+01
```