NAG Library Routine Document

E02AHF

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

E02AHF determines the coefficients in the Chebyshev series representation of the derivative of a polynomial given in Chebyshev series form.

2 Specification

```
SUBROUTINE E02AHF (NP1, XMIN, XMAX, A, IA1, LA, PATM1, ADIF, IADIF1,      &
                   LADIF, IFAIL)

INTEGER            NP1, IA1, LA, IADIF1, LADIF, IFAIL
REAL (KIND=nag_wp) XMIN, XMAX, A(LA), PATM1, ADIF(LADIF)
```

3 Description

E02AHF forms the polynomial which is the derivative of a given polynomial. Both the original polynomial and its derivative are represented in Chebyshev series form. Given the coefficients a_i, for $i = 0, 1, \ldots, n$, of a polynomial $p(x)$ of degree n, where

$$p(x) = \tfrac{1}{2}a_0 + a_1T_1(\bar{x}) + \cdots + a_nT_n(\bar{x})$$

the routine returns the coefficients $\bar{a}_i$, for $i = 0, 1, \ldots, n-1$, of the polynomial $q(x)$ of degree $n-1$, where

$$q(x) = \frac{dp(x)}{dx} = \tfrac{1}{2}\bar{a}_0 + \bar{a}_1T_1(\bar{x}) + \cdots + \bar{a}_{n-1}T_{n-1}(\bar{x}).$$

Here $T_j(\bar{x})$ denotes the Chebyshev polynomial of the first kind of degree j with argument $\bar{x}$. It is assumed that the normalized variable $\bar{x}$ in the interval $[-1, +1]$ was obtained from your original variable x in the interval $[x_{\mathrm{min}}, x_{\mathrm{max}}]$ by the linear transformation

$$\bar{x} = \frac{2x - (x_{\mathrm{max}} + x_{\mathrm{min}})}{x_{\mathrm{max}} - x_{\mathrm{min}}}$$

and that you require the derivative to be with respect to the variable x. If the derivative with respect to $\bar{x}$ is required, set $x_{\mathrm{max}} = 1$ and $x_{\mathrm{min}} = -1$.

Values of the derivative can subsequently be computed, from the coefficients obtained, by using E02AKF.

The method employed is that of Chebyshev series (see Chapter 8 of Modern Computing Methods (1961)), modified to obtain the derivative with respect to x. Initially setting $\bar{a}_{n+1} = \bar{a}_n = 0$, the routine forms successively

$$\bar{a}_{i-1} = \bar{a}_{i+1} + \frac{2}{x_{\mathrm{max}} - x_{\mathrm{min}}} 2ia_i, \quad i = n, n-1, \ldots, 1.$$

4 References

Modern Computing Methods (1961) Chebyshev-series *NPL Notes on Applied Science* **16** (2nd Edition) HMSO

5 Parameters

1: NP1 – INTEGER *Input*

On entry: $n + 1$, where n is the degree of the given polynomial $p(x)$. Thus NP1 is the number of coefficients in this polynomial.

Constraint: $\text{NP1} \geq 1$.

2: XMIN – REAL (KIND=nag_wp) *Input*

3: XMAX – REAL (KIND=nag_wp) *Input*

On entry: the lower and upper end points respectively of the interval $[x_{\text{min}}, x_{\text{max}}]$. The Chebyshev series representation is in terms of the normalized variable $\bar{x}$, where

$$\bar{x} = \frac{2x - (x_{\text{max}} + x_{\text{min}})}{x_{\text{max}} - x_{\text{min}}}.$$

Constraint: $\text{XMAX} > \text{XMIN}$.

4: A(LA) – REAL (KIND=nag_wp) array *Input*

On entry: the Chebyshev coefficients of the polynomial $p(x)$. Specifically, element $i \times \text{IA1}$ of A must contain the coefficient a_i, for $i = 0, 1, \ldots, n$. Only these $n + 1$ elements will be accessed.

Unchanged on exit, but see ADIF, below.

5: IA1 – INTEGER *Input*

On entry: the index increment of A. Most frequently the Chebyshev coefficients are stored in adjacent elements of A, and IA1 must be set to 1. However, if for example, they are stored in $\text{A}(1), \text{A}(4), \text{A}(7), \ldots$, then the value of IA1 must be 3. See also Section 9.

Constraint: $\text{IA1} \geq 1$.

6: LA – INTEGER *Input*

On entry: the dimension of the array A as declared in the (sub)program from which E02AHF is called.

Constraint: $\text{LA} \geq 1 + (\text{NP1} - 1) \times \text{IA1}$.

7: PATM1 – REAL (KIND=nag_wp) *Output*

On exit: the value of $p(x_{\text{min}})$. If this value is passed to the integration routine E02AJF with the coefficients of $q(x)$, then the original polynomial $p(x)$ is recovered, including its constant coefficient.

8: ADIF(LADIF) – REAL (KIND=nag_wp) array *Output*

On exit: the Chebyshev coefficients of the derived polynomial $q(x)$. (The differentiation is with respect to the variable x.) Specifically, element $i \times \text{IADIF1} + 1$ of ADIF contains the coefficient $\bar{a}_i$, for $i = 0, 1, \ldots, n - 1$. Additionally, element $n \times \text{IADIF1} + 1$ is set to zero. A call of the routine may have the array name ADIF the same as A, provided that note is taken of the order in which elements are overwritten, when choosing the starting elements and increments IA1 and IADIF1, i.e., the coefficients $a_0, a_1, \ldots, a_{i-1}$ must be intact after coefficient $\bar{a}_i$ is stored. In particular, it is possible to overwrite the a_i completely by having $\text{IA1} = \text{IADIF1}$, and the actual arrays for A and ADIF identical.

9: IADIF1 – INTEGER *Input*

On entry: the index increment of ADIF. Most frequently the Chebyshev coefficients are required in adjacent elements of ADIF, and IADIF1 must be set to 1. However, if, for example, they are to be stored in $\mathrm{ADIF}(1), \mathrm{ADIF}(4), \mathrm{ADIF}(7), \ldots$, then the value of IADIF1 must be 3. See Section 9.

Constraint: $\mathrm{IADIF1} \geq 1$.

10: LADIF – INTEGER *Input*

On entry: the dimension of the array ADIF as declared in the (sub)program from which E02AHF is called.

Constraint: $\mathrm{LADIF} \geq 1 + (\mathrm{NP1} - 1) \times \mathrm{IADIF1}$.

11: 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: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

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

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = 1$

On entry, $\mathrm{NP1} < 1$,
or $\mathrm{XMAX} \leq \mathrm{XMIN}$,
or $\mathrm{IA1} < 1$,
or $\mathrm{LA} \leq (\mathrm{NP1} - 1) \times \mathrm{IA1}$,
or $\mathrm{IADIF1} < 1$,
or $\mathrm{LADIF} \leq (\mathrm{NP1} - 1) \times \mathrm{IADIF1}$.

$\mathrm{IFAIL} = -99$

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

See Section 3.8 in the Essential Introduction for further information.

$\mathrm{IFAIL} = -399$

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

See Section 3.7 in the Essential Introduction for further information.

$\mathrm{IFAIL} = -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

There is always a loss of precision in numerical differentiation, in this case associated with the multiplication by $2i$ in the formula quoted in Section 3.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken is approximately proportional to $n+1$.

The increments IA1, IADIF1 are included as parameters to give a degree of flexibility which, for example, allows a polynomial in two variables to be differentiated with respect to either variable without rearranging the coefficients.

10 Example

Suppose a polynomial has been computed in Chebyshev series form to fit data over the interval $[-0.5, 2.5]$. The following program evaluates the first and second derivatives of this polynomial at 4 equally spaced points over the interval. (For the purposes of this example, XMIN, XMAX and the Chebyshev coefficients are simply supplied in DATA statements. Normally a program would first read in or generate data and compute the fitted polynomial.)

10.1 Program Text

```
    Program e02ahfe

!     E02AHF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: e02ahf, e02akf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter    :: xmax = 2.5E0_nag_wp
      Real (Kind=nag_wp), Parameter    :: xmin = -0.5E0_nag_wp
      Integer, Parameter               :: nout = 6, np1 = 7
      Integer, Parameter               :: la = np1
      Integer, Parameter               :: ladif = np1
      Real (Kind=nag_wp), Parameter    :: a(la) = (/2.53213E0_nag_wp,          &
          1.13032E0_nag_wp,0.27150E0_nag_wp,0.04434E0_nag_wp,0.00547E0_nag_wp, &
                                          0.00054E0_nag_wp,0.00004E0_nag_wp/)
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: deriv, deriv2, patm1, x
      Integer                          :: i, ifail, m
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: adif(ladif), adif2(ladif)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: real
!     .. Executable Statements ..
      Write (nout,*) 'E02AHF Example Program Results'

      ifail = 0
      Call e02ahf(np1,xmin,xmax,a,1,la,patm1,adif,1,ladif,ifail)

      ifail = 0
      Call e02ahf(np1-1,xmin,xmax,adif,1,ladif,patm1,adif2,1,ladif,ifail)

      m = 4

      Write (nout,*)
```

```
       Write (nout,*) '   I  Argument    1st deriv    2nd deriv'

       Do i = 1, m
         x = (xmin*real(m-i,kind=nag_wp)+xmax*real(i-1,kind=nag_wp))/ &
           real(m-1,kind=nag_wp)

         ifail = 0
         Call e02akf(np1-1,xmin,xmax,adif,1,ladif,x,deriv,ifail)

         ifail = 0
         Call e02akf(np1-2,xmin,xmax,adif2,1,ladif,x,deriv2,ifail)

         Write (nout,99999) i, x, deriv, deriv2
       End Do

99999 Format (1X,I4,F9.4,2(4X,F9.4))
    End Program e02ahfe
```

10.2 Program Data

None.

10.3 Program Results

```
 E02AHF Example Program Results

    I  Argument    1st deriv    2nd deriv
    1  -0.5000       0.2453       0.1637
    2   0.5000       0.4777       0.3185
    3   1.5000       0.9304       0.6203
    4   2.5000       1.8119       1.2056
```