E02AFF – 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

E02AFF computes the coefficients of a polynomial, in its Chebyshev-series form, which interpolates (passes exactly through) data at a special set of points. Least-squares polynomial approximations can also be obtained.

2 Specification

```
SUBROUTINE E02AFF(NPLUS1, F, A, IFAIL)
INTEGER         NPLUS1, IFAIL
real            F(NPLUS1), A(NPLUS1)
```

3 Description

This routine computes the coefficients a_j, for $j = 1, 2, \ldots, n+1$, in the Chebyshev-series

$$\tfrac{1}{2}a_1 T_0(\bar{x}) + a_2 T_1(\bar{x}) + a_3 T_2(\bar{x}) + \ldots + a_{n+1} T_n(\bar{x}),$$

which interpolates the data f_r at the points

$$\bar{x}_r = \cos\left((r-1)\pi/n\right), \quad r = 1, 2, \ldots, n+1.$$

Here $T_j(\bar{x})$ denotes the Chebyshev polynomial of the first kind of degree j with argument $\bar{x}$. The use of these points minimizes the risk of unwanted fluctuations in the polynomial and is recommended when the data abscissae can be chosen by the user, e.g., when the data is given as a graph. For further advantages of this choice of points, see Clenshaw [2].

In terms of the user's original variables, x say, the values of x at which the data f_r are to be provided are

$$x_r = \tfrac{1}{2}(x_{\max} - x_{\min})\cos\left(\pi(r-1)/n\right) + \tfrac{1}{2}(x_{\max} + x_{\min}), \quad r = 1, 2, \ldots, n+1$$

where $x_{\max}$ and $x_{\min}$ are respectively the upper and lower ends of the range of x over which the user wishes to interpolate.

Truncation of the resulting series after the term involving a_{i+1}, say, yields a least-squares approximation to the data. This approximation, $p(\bar{x})$, say, is the polynomial of degree i which minimizes

$$\tfrac{1}{2}\epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2 + \ldots + \epsilon_n^2 + \tfrac{1}{2}\epsilon_{n+1}^2,$$

where the residual $\epsilon_r = p(\bar{x}_r) - f_r$, for $r = 1, 2, \ldots, n+1$.

The method employed is based on the application of the three-term recurrence relation due to Clenshaw [1] for the evaluation of the defining expression for the Chebyshev coefficients (see, for example, Clenshaw [2]). The modifications to this recurrence relation suggested by Reinsch and Gentleman (see [5]) are used to give greater numerical stability.

For further details of the algorithm and its use see Cox [3] and [4].

Subsequent evaluation of the computed polynomial, perhaps truncated after an appropriate number of terms, should be carried out using E02AEF.

4 References

[1] Clenshaw C W (1955) A note on the summation of Chebyshev series *Math. Tables Aids Comput.* **9** 118–120

[2] Clenshaw C W (1962) Mathematical tables *Chebyshev Series for Mathematical Functions* HMSO

[3] Cox M G (1974) A data-fitting package for the non-specialist user *Software for Numerical Mathematics* (ed D J Evans) Academic Press

[4] Cox M G and Hayes J G (1973) Curve fitting: A guide and suite of algorithms for the non-specialist user *NPL Report NAC 26* National Physical Laboratory

[5] Gentleman W M (1969) An error analysis of Goertzel's (Watt's) method for computing Fourier coefficients *Comput. J.* **12** 160–165

5   Parameters

1:   NPLUS1 — INTEGER   *Input*

On entry: the number $n + 1$ of data points (one greater than the degree n of the interpolating polynomial).

Constraint: NPLUS1 ≥ 2.

2:   F(NPLUS1) — ***real*** array   *Input*

On entry: for $r = 1, 2, \ldots, n+1$, F(r) must contain f_r the value of the dependent variable (ordinate) corresponding to the value
$$\bar{x}_r = \cos\left(\pi(r-1)/n\right)$$
of the independent variable (abscissa) $\bar{x}$, or equivalently to the value
$$x(r) = \tfrac{1}{2}(x_{\max} - x_{\min})\cos\left(\pi(r-1)/n\right) + \tfrac{1}{2}(x_{\max} + x_{\min})$$
of the user's original variable x. Here $x_{\max}$ and $x_{\min}$ are respectively the upper and lower ends of the range over which the user wishes to interpolate.

3:   A(NPLUS1) — ***real*** array   *Output*

On exit: A(j) is the coefficient a_j in the interpolating polynomial, for $j = 1, 2, \ldots, n+1$.

4:   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,   NPLUS1 < 2.

7   Accuracy

The rounding errors committed are such that the computed coefficients are exact for a slightly perturbed set of ordinates $f_r + \delta f_r$. The ratio of the sum of the absolute values of the δf_r to the sum of the absolute values of the f_r is less than a small multiple of $(n+1)\epsilon$, where ϵ is the ***machine precision***.

8   Further Comments

The time taken by the routine is approximately proportional to $(n+1)^2 + 30$.

For choice of degree when using the routine for least-squares approximation, see Section 3.2 of the Chapter Introduction.

9 Example

Determine the Chebyshev coefficients of the polynomial which interpolates the data $\bar{x}_r, f_r$, for $r = 1, 2, \ldots, 11$, where $\bar{x}_r = \cos(\pi \times (r-1)/10)$ and $f_r = e^{\bar{x}_r}$. Evaluate, for comparison with the values of f_r, the resulting Chebyshev series at $\bar{x}_r$, for $r = 1, 2, \ldots, 11$.

The example program supplied is written in a general form that will enable polynomial interpolations of arbitrary data at the cosine points $\cos(\pi \times (r-1)/n)$, for $r = 1, 2, \ldots, n+1$ to be obtained for any $n \, (= \text{NPLUS1} - 1)$. Note that E02AEF is used to evaluate the interpolating polynomial. The program is self-starting in that any number of data sets can be supplied.

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.

```
*       E02AFF Example Program Text
*       Mark 14 Revised.  NAG Copyright 1989.
*       .. Parameters ..
        INTEGER          NMAX, NP1MAX
        PARAMETER        (NMAX=199,NP1MAX=NMAX+1)
        INTEGER          NIN, NOUT
        PARAMETER        (NIN=5,NOUT=6)
*       .. Local Scalars ..
        real             FIT, PI, PIBY2N
        INTEGER          I, IFAIL, J, N, R
*       .. Local Arrays ..
        real             AN(NP1MAX), F(NP1MAX), XCAP(NP1MAX)
*       .. External Functions ..
        real             X01AAF
        EXTERNAL         X01AAF
*       .. External Subroutines ..
        EXTERNAL         E02AEF, E02AFF
*       .. Intrinsic Functions ..
        INTRINSIC        real, SIN
*       .. Executable Statements ..
        WRITE (NOUT,*) 'E02AFF Example Program Results'
*       Skip heading in data file
        READ (NIN,*)
        PI = X01AAF(PI)
   20   READ (NIN,*,END=80) N
        IF (N.GT.0 .AND. N.LE.NMAX) THEN
           PIBY2N = 0.5e0*PI/real(N)
           READ (NIN,*) (F(R),R=1,N+1)
           DO 40 R = 1, N + 1
              I = R - 1
*
*             The following method of evaluating  XCAP = cos(PI*I/N)
*             ensures that the computed value has a small relative error
*             and, moreover, is bounded in modulus by unity for all
*             I = 0, 1, ..., N.  (It is assumed that the sine routine
*             produces a result with a small relative error for values
*             of the argument between  -PI/4  and  PI/4).
*
              IF (4*I.LE.N) THEN
                 XCAP(I+1) = 1.0e0 - 2.0e0*SIN(PIBY2N*I)**2
              ELSE IF (4*I.GT.3*N) THEN
                 XCAP(I+1) = 2.0e0*SIN(PIBY2N*(N-I))**2 - 1.0e0
              ELSE
```

```
                  XCAP(I+1) = SIN(PIBY2N*(N-2*I))
               END IF
   40     CONTINUE
          IFAIL = 0
*
          CALL E02AFF(N+1,F,AN,IFAIL)
*
          WRITE (NOUT,*)
          WRITE (NOUT,*) '          Chebyshev'
          WRITE (NOUT,*) '  J   coefficient A(J)'
          WRITE (NOUT,99998) (J,AN(J),J=1,N+1)
          WRITE (NOUT,*)
          WRITE (NOUT,*) ' R    Abscissa   Ordinate     Fit'
          DO 60 R = 1, N + 1
             IFAIL = 0
*
             CALL E02AEF(N+1,AN,XCAP(R),FIT,IFAIL)
*
             WRITE (NOUT,99999) R, XCAP(R), F(R), FIT
   60     CONTINUE
          GO TO 20
       END IF
   80 STOP
*
99999 FORMAT (1X,I3,3F11.4)
99998 FORMAT (1X,I3,F14.7)
      END
```

9.2 Program Data

```
E02AFF Example Program Data
  10
     2.7182
     2.5884
     2.2456
     1.7999
     1.3620
     1.0000
     0.7341
     0.5555
     0.4452
     0.3863
     0.3678
```

9.3 Program Results

```
E02AFF Example Program Results

        Chebyshev
  J   coefficient A(J)
  1     2.5320000
  2     1.1303095
  3     0.2714893
  4     0.0443462
  5     0.0055004
  6     0.0005400
  7     0.0000307
```

```
 8    -0.0000006
 9    -0.0000004
10     0.0000049
11    -0.0000200

 R     Abscissa   Ordinate     Fit
 1      1.0000     2.7182     2.7182
 2      0.9511     2.5884     2.5884
 3      0.8090     2.2456     2.2456
 4      0.5878     1.7999     1.7999
 5      0.3090     1.3620     1.3620
 6      0.0000     1.0000     1.0000
 7     -0.3090     0.7341     0.7341
 8     -0.5878     0.5555     0.5555
 9     -0.8090     0.4452     0.4452
10     -0.9511     0.3863     0.3863
11     -1.0000     0.3678     0.3678
```