# NAG Library Routine Document

# E02AEF

**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

E02AEF evaluates a polynomial from its Chebyshev series representation.

## 2 Specification

```
SUBROUTINE E02AEF (NPLUS1, A, XCAP, P, IFAIL)

INTEGER            NPLUS1, IFAIL
REAL (KIND=nag_wp) A(NPLUS1), XCAP, P
```

## 3 Description

E02AEF evaluates the polynomial

$$\frac{1}{2}a_1T_0(\bar{x}) + a_2T_1(\bar{x}) + a_3T_2(\bar{x}) + \cdots + a_{n+1}T_n(\bar{x})$$

for any value of $\bar{x}$ satisfying $-1 \leq \bar{x} \leq 1$. Here $T_j(\bar{x})$ denotes the Chebyshev polynomial of the first kind of degree $j$ with argument $\bar{x}$. The value of $n$ is prescribed by you.

In practice, the variable $\bar{x}$ will usually have been obtained from an original variable $x$, where $x_{\text{min}} \leq x \leq x_{\text{max}}$ and

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

Note that this form of the transformation should be used computationally rather than the mathematical equivalent

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

since the former guarantees that the computed value of $\bar{x}$ differs from its true value by at most $4\epsilon$, where $\epsilon$ is the ***machine precision***, whereas the latter has no such guarantee.

The method employed is based on the three-term recurrence relation due to Clenshaw (1955), with modifications to give greater numerical stability due to Reinsch and Gentleman (see Gentleman (1969)).

For further details of the algorithm and its use see Cox (1974) and Cox and Hayes (1973).

## 4 References

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

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

Cox M G and Hayes J G (1973) Curve fitting: a guide and suite of algorithms for the non-specialist user *NPL Report NAC26* National Physical Laboratory

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 terms in the series (i.e., one greater than the degree of the polynomial).

*Constraint*: NPLUS1 $\geq 1$.

2: A(NPLUS1) – REAL (KIND=nag_wp) array *Input*

*On entry*: A($i$) must be set to the value of the $i$th coefficient in the series, for $i = 1, 2, \ldots, n+1$.

3: XCAP – REAL (KIND=nag_wp) *Input*

*On entry*: $\bar{x}$, the argument at which the polynomial is to be evaluated. It should lie in the range $-1$ to $+1$, but a value just outside this range is permitted (see Section 6) to allow for possible rounding errors committed in the transformation from $x$ to $\bar{x}$ discussed in Section 3. Provided the recommended form of the transformation is used, a successful exit is thus assured whenever the value of $x$ lies in the range $x_{\text{min}}$ to $x_{\text{max}}$.

4: P – REAL (KIND=nag_wp) *Output*

*On exit*: the value of the polynomial.

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 $= 1$

ABS(XCAP) $> 1.0 + 4\epsilon$, where $\epsilon$ is the ***machine precision***. In this case the value of P is set arbitrarily to zero.

IFAIL $= 2$

On entry, NPLUS1 $< 1$.

## 7 Accuracy

The rounding errors committed are such that the computed value of the polynomial is exact for a slightly perturbed set of coefficients $a_i + \delta a_i$. The ratio of the sum of the absolute values of the $\delta a_i$ to the sum of the absolute values of the $a_i$ is less than a small multiple of $(n+1) \times$ ***machine precision***.

## 8 Further Comments

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

It is expected that a common use of E02AEF will be the evaluation of the polynomial approximations produced by E02ADF and E02AFF.

## 9 Example

Evaluate at 11 equally-spaced points in the interval $-1 \leq \bar{x} \leq 1$ the polynomial of degree 4 with Chebyshev coefficients, 2.0, 0.5, 0.25, 0.125, 0.0625.

The example program is written in a general form that will enable a polynomial of degree $n$ in its Chebyshev series form to be evaluated at $m$ equally-spaced points in the interval $-1 \leq \bar{x} \leq 1$. The program is self-starting in that any number of datasets can be supplied.

### 9.1 Program Text

```
    Program e02aefe

!     E02AEF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: e02aef, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: p, xcap
      Integer                          :: i, ifail, m, n, nplus1, r
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: real
!     .. Executable Statements ..
      Write (nout,*) 'E02AEF Example Program Results'

!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) m
      Read (nin,*) n
      nplus1 = n + 1
      Allocate (a(nplus1))

      Read (nin,*)(a(i),i=1,nplus1)

      Do r = 1, m
        xcap = real(2*r-m-1,kind=nag_wp)/real(m-1,kind=nag_wp)

        ifail = 0
        Call e02aef(nplus1,a,xcap,p,ifail)

        If (r==1) Then
          Write (nout,*)
          Write (nout,*) '  R       Argument       Value of polynomial'
        End If

        Write (nout,99999) r, xcap, p
      End Do

99999 Format (1X,I3,F14.4,4X,F14.4)
    End Program e02aefe
```

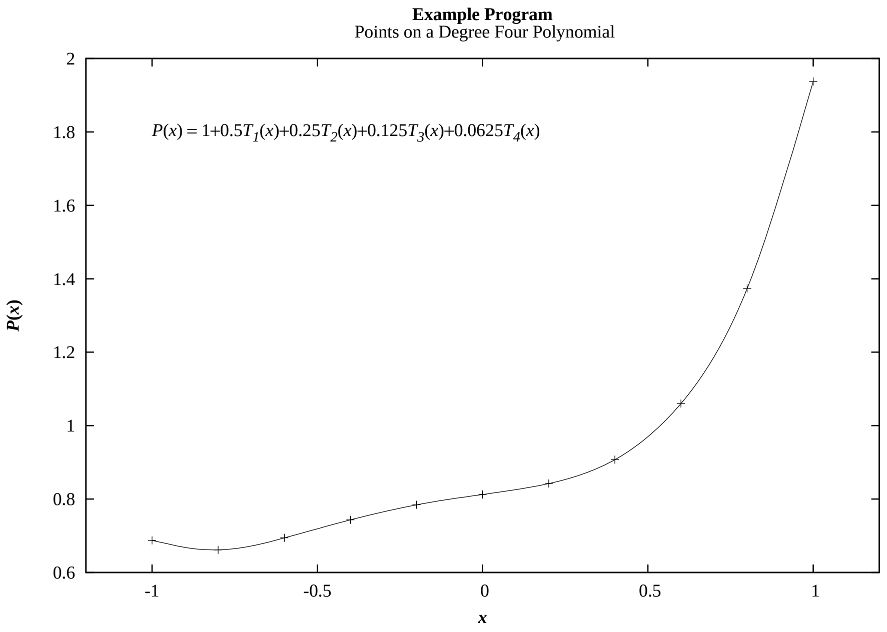
## 9.2 Program Data

```
E02AEF Example Program Data
  11
   4
     2.0000
     0.5000
     0.2500
     0.1250
     0.0625
```

## 9.3 Program Results

```
 E02AEF Example Program Results

   R       Argument       Value of polynomial
   1       -1.0000            0.6875
   2       -0.8000            0.6613
   3       -0.6000            0.6943
   4       -0.4000            0.7433
   5       -0.2000            0.7843
   6        0.0000            0.8125
   7        0.2000            0.8423
   8        0.4000            0.9073
   9        0.6000            1.0603
  10        0.8000            1.3733
  11        1.0000            1.9375
```


Example Program
Points on a Degree Four Polynomial
$P(x) = 1+0.5T_1(x)+0.25T_2(x)+0.125T_3(x)+0.0625T_4(x)$
P(x)
x