

# NAG Library Routine Document

## E02ADF

**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

E02ADF computes weighted least squares polynomial approximations to an arbitrary set of data points.

### 2 Specification

```
SUBROUTINE E02ADF (M, KPLUS1, LDA, X, Y, W, WORK1, WORK2, A, S, IFAIL)
INTEGER          M, KPLUS1, LDA, IFAIL
REAL (KIND=nag_wp) X(M), Y(M), W(M), WORK1(3*M), WORK2(2*KPLUS1),      &
                  A(LDA,KPLUS1), S(KPLUS1)
```

### 3 Description

E02ADF determines least squares polynomial approximations of degrees  $0, 1, \dots, k$  to the set of data points  $(x_r, y_r)$  with weights  $w_r$ , for  $r = 1, 2, \dots, m$ .

The approximation of degree  $i$  has the property that it minimizes  $\sigma_i$  the sum of squares of the weighted residuals  $\epsilon_r$ , where

$$\epsilon_r = w_r(y_r - f_r)$$

and  $f_r$  is the value of the polynomial of degree  $i$  at the  $r$ th data point.

Each polynomial is represented in Chebyshev series form with normalized argument  $\bar{x}$ . This argument lies in the range  $-1$  to  $+1$  and is related to the original variable  $x$  by the linear transformation

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

Here  $x_{\max}$  and  $x_{\min}$  are respectively the largest and smallest values of  $x_r$ . The polynomial approximation of degree  $i$  is represented as

$$\frac{1}{2}a_{i+1,1}T_0(\bar{x}) + a_{i+1,2}T_1(\bar{x}) + a_{i+1,3}T_2(\bar{x}) + \dots + a_{i+1,i+1}T_i(\bar{x}),$$

where  $T_j(\bar{x})$ , for  $j = 0, 1, \dots, i$ , are the Chebyshev polynomials of the first kind of degree  $j$  with argument  $(\bar{x})$ .

For  $i = 0, 1, \dots, k$ , the routine produces the values of  $a_{i+1,j+1}$ , for  $j = 0, 1, \dots, i$ , together with the value of the root-mean-square residual  $s_i = \sqrt{\sigma_i/(m - i - 1)}$ . In the case  $m = i + 1$  the routine sets the value of  $s_i$  to zero.

The method employed is due to Forsythe (1957) and is based on the generation of a set of polynomials orthogonal with respect to summation over the normalized dataset. The extensions due to Clenshaw (1960) to represent these polynomials as well as the approximating polynomials in their Chebyshev series forms are incorporated. The modifications suggested by Reinsch and Gentleman (see Gentleman (1969)) to the method originally employed by Clenshaw for evaluating the orthogonal polynomials from their Chebyshev series representations are used to give greater numerical stability.

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

Subsequent evaluation of the Chebyshev series representations of the polynomial approximations should be carried out using E02AEF.

## 4 References

Clenshaw C W (1960) Curve fitting with a digital computer *Comput. J.* **2** 170–173

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

Forsythe G E (1957) Generation and use of orthogonal polynomials for data fitting with a digital computer *J. Soc. Indust. Appl. Math.* **5** 74–88

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

Hayes J G (ed.) (1970) *Numerical Approximation to Functions and Data* Athlone Press, London

## 5 Arguments

- 1: M – INTEGER *Input*  
*On entry:* the number  $m$  of data points.  
*Constraint:*  $M \geq mdist \geq 2$ , where  $mdist$  is the number of distinct  $x$  values in the data.
- 2: KPLUS1 – INTEGER *Input*  
*On entry:*  $k + 1$ , where  $k$  is the maximum degree required.  
*Constraint:*  $0 < KPLUS1 \leq mdist$ , where  $mdist$  is the number of distinct  $x$  values in the data.
- 3: LDA – INTEGER *Input*  
*On entry:* the first dimension of the array A as declared in the (sub)program from which E02ADF is called.  
*Constraint:*  $LDA \geq KPLUS1$ .
- 4: X(M) – REAL (KIND=nag\_wp) array *Input*  
*On entry:* the values  $x_r$  of the independent variable, for  $r = 1, 2, \dots, m$ .  
*Constraint:* the values must be supplied in nondecreasing order with  $X(M) > X(1)$ .
- 5: Y(M) – REAL (KIND=nag\_wp) array *Input*  
*On entry:* the values  $y_r$  of the dependent variable, for  $r = 1, 2, \dots, m$ .
- 6: W(M) – REAL (KIND=nag\_wp) array *Input*  
*On entry:* the set of weights,  $w_r$ , for  $r = 1, 2, \dots, m$ . For advice on the choice of weights, see Section 2.1.2 in the E02 Chapter Introduction.  
*Constraint:*  $W(r) > 0.0$ , for  $r = 1, 2, \dots, m$ .
- 7: WORK1( $3 \times M$ ) – REAL (KIND=nag\_wp) array *Workspace*
- 8: WORK2( $2 \times KPLUS1$ ) – REAL (KIND=nag\_wp) array *Workspace*
- 9: A(LDA, KPLUS1) – REAL (KIND=nag\_wp) array *Output*  
*On exit:* the coefficients of  $T_j(\bar{x})$  in the approximating polynomial of degree  $i$ .  $A(i + 1, j + 1)$  contains the coefficient  $a_{i+1, j+1}$ , for  $i = 0, 1, \dots, k$  and  $j = 0, 1, \dots, i$ .

- 10: S(KPLUS1) – REAL (KIND=nag\_wp) array *Output*  
*On exit:* S( $i + 1$ ) contains the root-mean-square residual  $s_i$ , for  $i = 0, 1, \dots, k$ , as described in Section 3. For the interpretation of the values of the  $s_i$  and their use in selecting an appropriate degree, see Section 3.1 in the E02 Chapter Introduction.
- 11: IFAIL – INTEGER *Input/Output*  
*On entry:* IFAIL must be set to 0,  $-1$  or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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 argument, 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

The weights are not all strictly positive.

IFAIL = 2

The values of  $X(r)$ , for  $r = 1, 2, \dots, M$ , are not in nondecreasing order.

IFAIL = 3

All  $X(r)$  have the same value: thus the normalization of X is not possible.

IFAIL = 4

On entry,  $KPLUS1 < 1$  (so the maximum degree required is negative)  
 or  $KPLUS1 > mdist$ , where  $mdist$  is the number of distinct  $x$  values in the data (so there cannot be a unique solution for degree  $k = KPLUS1 - 1$ ).

IFAIL = 5

LDA < KPLUS1.

IFAIL =  $-99$

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

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL =  $-399$

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

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL =  $-999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

## 7 Accuracy

No error analysis for the method has been published. Practical experience with the method, however, is generally extremely satisfactory.

## 8 Parallelism and Performance

E02ADF is not threaded in any implementation.

## 9 Further Comments

The time taken is approximately proportional to  $m(k+1)(k+11)$ .

The approximating polynomials may exhibit undesirable oscillations (particularly near the ends of the range) if the maximum degree  $k$  exceeds a critical value which depends on the number of data points  $m$  and their relative positions. As a rough guide, for equally-spaced data, this critical value is about  $2 \times \sqrt{m}$ . For further details see page 60 of Hayes (1970).

## 10 Example

Determine weighted least squares polynomial approximations of degrees 0, 1, 2 and 3 to a set of 11 prescribed data points. For the approximation of degree 3, tabulate the data and the corresponding values of the approximating polynomial, together with the residual errors, and also the values of the approximating polynomial at points half-way between each pair of adjacent data points.

The example program supplied is written in a general form that will enable polynomial approximations of degrees  $0, 1, \dots, k$  to be obtained to  $m$  data points, with arbitrary positive weights, and the approximation of degree  $k$  to be tabulated. E02AEF is used to evaluate the approximating polynomial. The program is self-starting in that any number of datasets can be supplied.

### 10.1 Program Text

```

Program e02adfe

!      E02ADF Example Program Text

!      Mark 26 Release. NAG Copyright 2016.

!      .. Use Statements ..
Use nag_library, Only: e02adf, e02aef, nag_wp
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
Real (Kind=nag_wp)         :: fit, x1, xarg, xcapr, xm
Integer                    :: i, ifail, iwght, j, k, kplus1, lda, &
                           m, r
!      .. Local Arrays ..
Real (Kind=nag_wp), Allocatable :: a(:,,:), ak(:), s(:), w(:), work1(:), &
                           work2(:,), x(:), y(:)
!      .. Executable Statements ..
Write (nout,*) 'E02ADF Example Program Results'

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

Read (nin,*) m
Read (nin,*) k, iwght
kplus1 = k + 1
lda = kplus1
Allocate (a(lda,kplus1),s(kplus1),w(m),work1(3*m),work2(2*kplus1),x(m), &
        y(m))

Do r = 1, m

```

```

      If (iwght/=1) Then
        Read (nin,*) x(r), y(r), w(r)
      Else
        Read (nin,*) x(r), y(r)
        w(r) = 1.0E0_nag_wp
      End If

End Do

ifail = 0
Call e02adf(m,kplus1,lda,x,y,w,work1,work2,a,s,ifail)

Do i = 0, k
  Write (nout,*)
  Write (nout,99998) 'Degree', i, '   R.M.S. residual =', s(i+1)
  Write (nout,*)
  Write (nout,*) ' J Chebyshev coeff A(J)'
  Write (nout,99997)(j,a(i+1,j),j=1,i+1)
End Do

Allocate (ak(kplus1))

ak(1:kplus1) = a(kplus1,1:kplus1)
x1 = x(1)
xm = x(m)

Write (nout,*)
Write (nout,99996) 'Polynomial approximation and residuals for degree', &
  k
Write (nout,*)
Write (nout,*)
Write (nout,*) ' R   Abscissa   Weight   Ordinate   Polynomial   Residual' &

Do r = 1, m
  xcapr = ((x(r)-x1)-(xm-x(r)))/(xm-x1)

  ifail = 0
  Call e02aef(kplus1,ak,xcapr,fit,ifail)

  Write (nout,99999) r, x(r), w(r), y(r), fit, fit - y(r)

  If (r<m) Then
    xarg = 0.5E0_nag_wp*(x(r)+x(r+1))
    xcapr = ((xarg-x1)-(xm-xarg))/(xm-x1)

    ifail = 0
    Call e02aef(kplus1,ak,xcapr,fit,ifail)

    Write (nout,99995) xarg, fit
  End If

End Do

99999 Format (1X,I3,4F11.4,E11.2)
99998 Format (1X,A,I4,A,E12.2)
99997 Format (1X,I3,F15.4)
99996 Format (1X,A,I4)
99995 Format (4X,F11.4,22X,F11.4)
End Program e02adfe

```

## 10.2 Program Data

E02ADF Example Program Data

```

11
 3  2
   1.00   10.40   1.00
   2.10    7.90   1.00
   3.10    4.70   1.00
   3.90    2.50   1.00

```

4.90	1.20	1.00
5.80	2.20	0.80
6.50	5.10	0.80
7.10	9.20	0.70
7.80	16.10	0.50
8.40	24.50	0.30
9.00	35.30	0.20

### 10.3 Program Results

E02ADF Example Program Results

Degree 0 R.M.S. residual = 0.41E+01

J Chebyshev coeff A(J)  
1 12.1740

Degree 1 R.M.S. residual = 0.43E+01

J Chebyshev coeff A(J)  
1 12.2954  
2 0.2740

Degree 2 R.M.S. residual = 0.17E+01

J Chebyshev coeff A(J)  
1 20.7345  
2 6.2016  
3 8.1876

Degree 3 R.M.S. residual = 0.68E-01

J Chebyshev coeff A(J)  
1 24.1429  
2 9.4065  
3 10.8400  
4 3.0589

Polynomial approximation and residuals for degree 3

R	Abcissa	Weight	Ordinate	Polynomial	Residual
1	1.0000	1.0000	10.4000	10.4461	0.46E-01
	1.5500			9.3106	
2	2.1000	1.0000	7.9000	7.7977	-0.10E+00
	2.6000			6.2555	
3	3.1000	1.0000	4.7000	4.7025	0.25E-02
	3.5000			3.5488	
4	3.9000	1.0000	2.5000	2.5533	0.53E-01
	4.4000			1.6435	
5	4.9000	1.0000	1.2000	1.2390	0.39E-01
	5.3500			1.4257	
6	5.8000	0.8000	2.2000	2.2425	0.42E-01
	6.1500			3.3803	
7	6.5000	0.8000	5.1000	5.0116	-0.88E-01
	6.8000			6.8400	
8	7.1000	0.7000	9.2000	9.0982	-0.10E+00
	7.4500			12.3171	
9	7.8000	0.5000	16.1000	16.2123	0.11E+00
	8.1000			20.1266	
10	8.4000	0.3000	24.5000	24.6048	0.10E+00
	8.7000			29.6779	
11	9.0000	0.2000	35.3000	35.3769	0.77E-01

**Example Program**  
 Least-squares Cubic Polynomial Approximation to a set of 11 Data Points

