NAG Library Routine Document

E02AGF

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

E02AGF computes constrained weighted least squares polynomial approximations in Chebyshev series form to an arbitrary set of data points. The values of the approximations and any number of their derivatives can be specified at selected points.

2 Specification

```
SUBROUTINE E02AGF (M, KPLUS1, LDA, XMIN, XMAX, X, Y, W, MF, XF, YF, LYF,    &
                   IP, A, S, NP1, WRK, LWRK, IWRK, LIWRK, IFAIL)

INTEGER            M, KPLUS1, LDA, MF, LYF, IP(MF), NP1, LWRK,             &
                   IWRK(LIWRK), LIWRK, IFAIL
REAL (KIND=nag_wp) XMIN, XMAX, X(M), Y(M), W(M), XF(MF), YF(LYF),          &
                   A(LDA,KPLUS1), S(KPLUS1), WRK(LWRK)
```

3 Description

E02AGF determines least squares polynomial approximations of degrees up to k to the set of data points (x_r, y_r) with weights w_r, for $r = 1, 2, \ldots, m$. The value of k, the maximum degree required, is to be prescribed by you. At each of the values xf_r, for $r = 1, 2, \ldots, mf$, of the independent variable x, the approximations and their derivatives up to order p_r are constrained to have one of the values yf_s, for $s = 1, 2, \ldots, n$, specified by you, where $n = mf + \sum_{r=0}^{mf} p_r$.

The approximation of degree i has the property that, subject to the imposed constraints, it minimizes σ_i, the sum of the squares of the weighted residuals ϵ_r, for $r = 1, 2, \ldots, m$, where

$$\epsilon_r = w_r(y_r - f_i(x_r))$$

and $f_i(x_r)$ is the value of the polynomial approximation of degree i at the rth 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

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

where $x_{\min}$ and $x_{\max}$, specified by you, are respectively the lower and upper end points of the interval of x over which the polynomials are to be defined.

The polynomial approximation of degree i can be written as

$$\tfrac{1}{2}a_{i,0} + a_{i,1}T_1(\bar{x}) + \cdots + a_{ij}T_j(\bar{x}) + \cdots + a_{ii}T_i(\bar{x})$$

where $T_j(\bar{x})$ is the Chebyshev polynomial of the first kind of degree j with argument $\bar{x}$. For $i = n, n+1, \ldots, k$, the routine produces the values of the coefficients a_{ij}, for $j = 0, 1, \ldots, i$, together with the value of the root mean square residual,

$$S_i = \sqrt{\frac{\sigma_i}{(m' + n - i - 1)}},$$

where m' is the number of data points with nonzero weight.

Values of the approximations may subsequently be computed using E02AEF or E02AKF.

First E02AGF determines a polynomial $\mu(\bar{x})$, of degree $n-1$, which satisfies the given constraints, and a polynomial $\nu(\bar{x})$, of degree n, which has value (or derivative) zero wherever a constrained value (or derivative) is specified. It then fits $y_r - \mu(x_r)$, for $r = 1, 2, \ldots, m$, with polynomials of the required degree in $\bar{x}$ each with factor $\nu(\bar{x})$. Finally the coefficients of $\mu(\bar{x})$ are added to the coefficients of these fits to give the coefficients of the constrained polynomial approximations to the data points (x_r, y_r), for $r = 1, 2, \ldots, m$. The method employed is given in Hayes (1970): it is an extension of Forsythe's orthogonal polynomials method (see Forsythe (1957)) as modified by Clenshaw (see Clenshaw (1960)).

4 References

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

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

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

5 Arguments

1: M – INTEGER *Input*

On entry: m, the number of data points to be fitted.

Constraint: M ≥ 1.

2: KPLUS1 – INTEGER *Input*

On entry: $k+1$, where k is the maximum degree required.

Constraint: $n+1 \leq$ KPLUS1 $\leq m'' + n$ is the total number of constraints and m'' is the number of data points with nonzero weights and distinct abscissae which do not coincide with any of the XF(r).

3: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which E02AGF is called.

Constraint: LDA $\geq$ KPLUS1.

4: XMIN – REAL (KIND=nag_wp) *Input*

5: XMAX – REAL (KIND=nag_wp) *Input*

On entry: the lower and upper end points, respectively, of the interval $[x_{\min}, x_{\max}]$. Unless there are specific reasons to the contrary, it is recommended that XMIN and XMAX be set respectively to the lowest and highest value among the x_r and xf_r. This avoids the danger of extrapolation provided there is a constraint point or data point with nonzero weight at each end point.

Constraint: XMAX $>$ XMIN.

6: X(M) – REAL (KIND=nag_wp) array *Input*

On entry: X(r) must contain the value x_r of the independent variable at the rth data point, for $r = 1, 2, \ldots, m$.

Constraint: the X(r) must be in nondecreasing order and satisfy XMIN $\leq$ X(r) $\leq$ XMAX.

7: Y(M) – REAL (KIND=nag_wp) array *Input*

On entry: Y(r) must contain y_r, the value of the dependent variable at the rth data point, for $r = 1, 2, \ldots, m$.

8: W(M) – REAL (KIND=nag_wp) array *Input*

On entry: W(r) must contain the weight w_r to be applied to the data point x_r, for $r = 1, 2, \ldots, m$. For advice on the choice of weights see the E02 Chapter Introduction. Negative weights are treated as positive. A zero weight causes the corresponding data point to be ignored. Zero weight should be given to any data point whose x and y values both coincide with those of a constraint (otherwise the denominators involved in the root mean square residuals S_i will be slightly in error).

9: MF – INTEGER *Input*

On entry: mf, the number of values of the independent variable at which a constraint is specified.

Constraint: MF ≥ 1.

10: XF(MF) – REAL (KIND=nag_wp) array *Input*

On entry: XF(r) must contain xf_r, the value of the independent variable at which a constraint is specified, for $r = 1, 2, \ldots,$ MF.

Constraint: these values need not be ordered but must be distinct and satisfy XMIN $\leq$ XF(r) $\leq$ XMAX.

11: YF(LYF) – REAL (KIND=nag_wp) array *Input*

On entry: the values which the approximating polynomials and their derivatives are required to take at the points specified in XF. For each value of XF(r), YF contains in successive elements the required value of the approximation, its first derivative, second derivative, $\ldots, p_r$th derivative, for $r = 1, 2, \ldots, mf$. Thus the value, yf_s, which the kth derivative of each approximation ($k = 0$ referring to the approximation itself) is required to take at the point XF(r) must be contained in YF(s), where

$$s = r + k + p_1 + p_2 + \cdots + p_{r-1},$$

where $k = 0, 1, \ldots, p_r$ and $r = 1, 2, \ldots, mf$. The derivatives are with respect to the independent variable x.

12: LYF – INTEGER *Input*

On entry: the dimension of the array YF as declared in the (sub)program from which E02AGF is called.

Constraint: LYF $\geq$ MF $+ \sum_{i=1}^{\text{MF}}$ IP(i).

13: IP(MF) – INTEGER array *Input*

On entry: IP(r) must contain p_r, the order of the highest-order derivative specified at XF(r), for $r = 1, 2, \ldots, mf$. $p_r = 0$ implies that the value of the approximation at XF(r) is specified, but not that of any derivative.

Constraint: IP(r) ≥ 0, for $r = 1, 2, \ldots,$ MF.

14: A(LDA, KPLUS1) – REAL (KIND=nag_wp) array *Output*

On exit: A($i + 1, j + 1$) contains the coefficient a_{ij} in the approximating polynomial of degree i, for $i = n, \ldots, k$ and $j = 0, 1, \ldots, i$.

15: S(KPLUS1) – REAL (KIND=nag_wp) array *Output*

On exit: S($i + 1$) contains S_i, for $i = n, \ldots, k$, the root mean square residual corresponding to the approximating polynomial of degree i. In the case where the number of data points with nonzero weight is equal to $k + 1 - n$, S_i is indeterminate: the routine sets it to zero. For the interpretation

of the values of S_i and their use in selecting an appropriate degree, see Section 3.1 in the E02 Chapter Introduction.

16: NP1 – INTEGER *Output*

On exit: $n+1$, where n is the total number of constraint conditions imposed: $n = mf + p_1 + p_2 + \cdots + p_{mf}$.

17: WRK(LWRK) – REAL (KIND=nag_wp) array *Output*

On exit: contains weighted residuals of the highest degree of fit determined (k). The residual at x_r is in element $2(n+1)+3(m+k+1)+r$, for $r = 1, 2, \ldots, m$. The rest of the array is used as workspace.

18: LWRK – INTEGER *Input*

On entry: the dimension of the array WRK as declared in the (sub)program from which E02AGF is called.

Constraint: LWRK $\geq \max(4 \times \text{M} + 3 \times \text{KPLUS1}, 8 \times n + 5 \times ipmax + \text{MF} + 10) + 2 \times n + 2$, where $ipmax = \max(\text{IP}(r))$, for $r = 1, 2, \ldots, mf$.

19: IWRK(LIWRK) – INTEGER array *Workspace*

20: LIWRK – INTEGER *Input*

On entry: the dimension of the array IWRK as declared in the (sub)program from which E02AGF is called.

Constraint: LIWRK $\geq 2 \times \text{MF} + 2$.

21: 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 $= 1$

On entry, M < 1,
or KPLUS1 $< n + 1$,
or LDA $<$ KPLUS1,
or MF < 1,
or LYF $< n$,
or LWRK is too small (see Section 5),
or LIWRK $< 2 \times$ MF $+ 2$.

(Here n is the total number of constraint conditions.)

IFAIL $= 2$

IP$(r) < 0$ for some $r = 1, 2, \ldots,$ MF.

IFAIL $= 3$

XMIN $\geq$ XMAX, or XF(r) is not in the interval XMIN to XMAX for some $r = 1, 2, \ldots,$ MF, or the XF(r) are not distinct.

IFAIL $= 4$

X(r) is not in the interval XMIN to XMAX for some $r = 1, 2, \ldots,$ M.

IFAIL $= 5$

X$(r) <$ X$(r-1)$ for some $r = 2, 3, \ldots,$ M.

IFAIL $= 6$

KPLUS1 $> m'' + n$, where m'' is the number of data points with nonzero weight and distinct abscissae which do not coincide with any XF(r). Thus there is no unique solution.

IFAIL $= 7$

The polynomials $\mu(x)$ and/or $\nu(x)$ cannot be determined. The problem supplied is too ill-conditioned. This may occur when the constraint points are very close together, or large in number, or when an attempt is made to constrain high-order derivatives.

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 complete error analysis exists for either the interpolating algorithm or the approximating algorithm. However, considerable experience with the approximating algorithm shows that it is generally extremely satisfactory. Also the moderate number of constraints, of low-order, which are typical of data fitting applications, are unlikely to cause difficulty with the interpolating routine.

8 Parallelism and Performance

E02AGF is not threaded in any implementation.

9 Further Comments

The time taken to form the interpolating polynomial is approximately proportional to n^3, and that to form the approximating polynomials is very approximately proportional to $m(k+1)(k+1-n)$.

To carry out a least squares polynomial fit without constraints, use E02ADF. To carry out polynomial interpolation only, use E01AEF.

10 Example

This example reads data in the following order, using the notation of the argument list above:

MF

IP(i), XF(i), Y-value and derivative values (if any) at XF(i), for $i = 1, 2, \ldots, \mathrm{MF}$

M

X(i), Y(i), W(i), for $i = 1, 2, \ldots, \mathrm{M}$

k, XMIN, XMAX

The output is:

the root mean square residual for each degree from n to k;

the Chebyshev coefficients for the fit of degree k;

the data points, and the fitted values and residuals for the fit of degree k.

The program is written in a generalized form which will read any number of datasets.

The dataset supplied specifies 5 data points in the interval $[0.0, 4.0]$ with unit weights, to which are to be fitted polynomials, p, of degrees up to 4, subject to the 3 constraints:

$$p(0.0) = 1.0, \quad p'(0.0) = -2.0, \quad p(4.0) = 9.0.$$

10.1 Program Text

```
    Program e02agfe

!     E02AGF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: e02agf, e02akf, f16dnf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: fiti, xmax, xmin
      Integer                          :: i, ifail, ipmax, k, kplus1, la, lda, &
                                          liwrk, lwrk, lyf, m, mf, n, np1
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), s(:), w(:), wrk(:), x(:),    &
                                          xf(:), y(:), yf(:)
      Integer, Allocatable             :: ip(:), iwrk(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, sum
!     .. Executable Statements ..
      Write (nout,*) 'E02AGF Example Program Results'

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

      Read (nin,*) mf
      liwrk = 2*mf + 2
      Allocate (ip(mf),xf(mf),iwrk(liwrk))

      Read (nin,*) ip(1:mf)

!     Get max(IP) for later use

      Call f16dnf(mf,ip,1,i,ipmax)

      Read (nin,*) xf(1:mf)

      lyf = mf + sum(ip(1:mf))
```

```
      Allocate (yf(lyf))

      Read (nin,*) yf(1:lyf)

      Read (nin,*) m
      Allocate (x(m),y(m),w(m))

      Read (nin,*)(x(i),y(i),w(i),i=1,m)

      Read (nin,*) k, xmin, xmax
      kplus1 = k + 1
      n = lyf
      np1 = n + 1
      lwrk = max(4*m+3*kplus1,8*n+5*ipmax+mf+10) + 2*n + 2
      lda = kplus1
      Allocate (wrk(lwrk),a(lda,kplus1),s(kplus1))

      ifail = 0
      Call e02agf(m,kplus1,lda,xmin,xmax,x,y,w,mf,xf,yf,lyf,ip,a,s,np1,wrk,    &
        lwrk,iwrk,liwrk,ifail)

      Write (nout,*)
      Write (nout,*) 'Degree  RMS residual'
      Write (nout,99999)(i,s(i+1),i=np1-1,k)
      Write (nout,*)
      Write (nout,99996) 'Details of the fit of degree ', k
      Write (nout,*)
      Write (nout,*) '  Index   Coefficient'

      Do i = 1, kplus1
        Write (nout,99997) i - 1, a(kplus1,i)
      End Do

      Write (nout,*)
      Write (nout,*) '     I      X(I)       Y(I)       Fit     Residual'

      Do i = 1, m
        la = lda*kplus1 - k

        ifail = 0
        Call e02akf(kplus1,xmin,xmax,a(kplus1,1),lda,la,x(i),fiti,ifail)

        Write (nout,99998) i, x(i), y(i), fiti, fiti - y(i)
      End Do

99999 Format (1X,I4,1P,E15.2)
99998 Format (1X,I6,3F11.4,E11.2)
99997 Format (1X,I6,F11.4)
99996 Format (1X,A,I2)
    End Program e02agfe
```

10.2 Program Data

```
E02AGF Example Program Data
    2                                                   : MF
    1       0                                           : IP
    0.0     4.0                                         : XF
    1.0      -2.0
    9.0                                                 : YF
    5                                                   : M
       0.5      0.03       1.0
       1.0     -0.75       1.0
       2.0     -1.0        1.0
       2.5     -0.1        1.0
       3.0      1.75       1.0                         : X, Y, W
    4       0.0       4.0                               : K, XMIN, XMAX
```

10.3 Program Results

```
 E02AGF Example Program Results

 Degree  RMS residual
    3       2.55E-03
    4       2.94E-03

 Details of the fit of degree  4

   Index   Coefficient
      0     3.9980
      1     3.4995
      2     3.0010
      3     0.5005
      4    -0.0000

      I      X(I)       Y(I)       Fit     Residual
      1     0.5000     0.0300     0.0310   0.10E-02
      2     1.0000    -0.7500    -0.7508  -0.78E-03
      3     2.0000    -1.0000    -1.0020  -0.20E-02
      4     2.5000    -0.1000    -0.0961   0.39E-02
      5     3.0000     1.7500     1.7478  -0.22E-02
```