NAG Library Routine Document

E01AEF

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

E01AEF constructs the Chebyshev series representation of a polynomial interpolant to a set of data which may contain derivative values.

2 Specification

```
 SUBROUTINE E01AEF(M, XMIN, XMAX, X, Y, IP, N, ITMIN, ITMAX, A, WRK,
1                  LWRK, IWRK, LIWRK, IFAIL)
 INTEGER           M, IP(M), N, ITMIN, ITMAX, LWRK, IWRK(LIWRK), LIWRK,
1                  IFAIL
 double precision  XMIN, XMAX, X(M), Y(N), A(N), WRK(LWRK)
```

3 Description

Let m distinct values x_i of an independent variable x be given, with $x_{\min} \leq x_i \leq x_{\max}$, for $i = 1, 2, \ldots, m$. For each value x_i, suppose that the value y_i of the dependent variable y together with the first p_i derivatives of y with respect to x are given. Each p_i must therefore be a non-negative integer, with the total number of interpolating conditions, n, equal to $m + \sum_{i=1}^{m} p_i$.

E01AEF calculates the unique polynomial $q(x)$ of degree $n-1$ (or less) which is such that $q^{(k)}(x_i) = y_i^{(k)}$, for $i = 1, 2, \ldots, m$ and $k = 0, 1, \ldots, p_i$. Here $q^{(0)}(x_i)$ means $q(x_i)$. This polynomial is represented in Chebyshev series form in the normalized variable $\bar{x}$, as follows:

$$q(x) = \tfrac{1}{2}a_0 T_0(\bar{x}) + a_1 T_1(\bar{x}) + \cdots + a_{n-1} T_{n-1}(\bar{x}),$$

where

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

so that $-1 \leq \bar{x} \leq 1$ for x in the interval $x_{\min}$ to $x_{\max}$, and where $T_i(\bar{x})$ is the Chebyshev polynomial of the first kind of degree i with argument $\bar{x}$.

(The polynomial interpolant can subsequently be evaluated for any value of x in the given range by using E02AKF. Chebyshev series representations of the derivative(s) and integral(s) of $q(x)$ may be obtained by (repeated) use of E02AHF and E02AJF.)

The method used consists first of constructing a divided-difference table from the normalized $\bar{x}$ values and the given values of y and its derivatives with respect to $\bar{x}$. The Newton form of $q(x)$ is then obtained from this table, as described in Huddleston (1974) and Krogh (1970), with the modification described in Section 8.2. The Newton form of the polynomial is then converted to Chebyshev series form as described in Section 8.3.

Since the errors incurred by these stages can be considerable, a form of iterative refinement is used to improve the solution. This refinement is particularly useful when derivatives of rather high order are given in the data. In reasonable examples, the refinement will usually terminate with a certain accuracy criterion satisfied by the polynomial (see Section 7). In more difficult examples, the criterion may not be satisfied and refinement will continue until the maximum number of iterations (as specified by the input parameter ITMAX) is reached.

In extreme examples, the iterative process may diverge (even though the accuracy criterion is satisfied): if a certain divergence criterion is satisfied, the process terminates at once. In all cases the routine returns the 'best' polynomial achieved before termination. For the definition of 'best' and details of iterative refinement and termination criteria, see Section 8.4.

4 References

Huddleston R E (1974) CDC 6600 routines for the interpolation of data and of data with derivatives *SLL-74-0214* Sandia Laboratories (Reprint)

Krogh F T (1970) Efficient algorithms for polynomial interpolation and numerical differentiation *Math. Comput.* **24** 185–190

5 Parameters

1: M – INTEGER *Input*

On entry: m, the number of given values of the independent variable x.

Constraint: M ≥ 1.

2: XMIN – ***double precision*** *Input*

3: XMAX – ***double precision*** *Input*

On entry: the lower and upper end points, respectively, of the interval $[x_{\min}, x_{\max}]$. If they are not determined by your problem, it is recommended that they be set respectively to the smallest and largest values among the x_i.

Constraint: XMIN $<$ XMAX.

4: X(M) – ***double precision*** array *Input*

On entry: the value of x_i, for $i = 1, 2, \ldots, m$. The X(i) need not be ordered.

Constraint: XMIN $\leq$ X(i) $\leq$ XMAX, and the X(i) must be distinct.

5: Y(N) – ***double precision*** array *Input*

On entry: the given values of the dependent variable, and derivatives, as follows:

The first $p_1 + 1$ elements contain $y_1, y_1^{(1)}, \ldots, y_1^{(p_1)}$ in that order.

The next $p_2 + 1$ elements contain $y_2, y_2^{(1)}, \ldots, y_2^{(p_2)}$ in that order.

$\vdots$

The last $p_m + 1$ elements contain $y_m, y_m^{(1)}, \ldots, y_m^{(p_m)}$ in that order.

6: IP(M) – INTEGER array *Input*

On entry: p_i, the order of the highest-order derivative whose value is given at x_i, for $i = 1, 2, \ldots, m$. If the value of y only is given for some x_i then the corresponding value of IP(i) must be zero.

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

7: N – INTEGER *Input*

On entry: n, the total number of interpolating conditions.

Constraint: $\mathrm{N} = \mathrm{M} + \sum_{i=1}^{\mathrm{M}} \mathrm{IP}(i)$.

8: ITMIN – INTEGER *Input*
9: ITMAX – INTEGER *Input*

On entry: respectively the minimum and maximum number of iterations to be performed by the routine (for full details see Section 8.4). Setting ITMIN and/or ITMAX negative or zero invokes default value(s) of 2 and/or 10, respectively.

The default values will be satisfactory for most problems, but occasionally significant improvement will result from using higher values.

Suggested value: ITMIN $= 0$ and ITMAX $= 0$.

10: A(N) – ***double precision*** array *Output*

On exit: A$(i+1)$ contains the coefficient a_i in the Chebyshev series representation of $q(x)$, for $i = 0, 1, \ldots, n-1$.

11: WRK(LWRK) – ***double precision*** array *Output*

On exit: used as workspace, but see also Section 8.5.

12: LWRK – INTEGER *Input*

On entry: the dimension of the array WRK as declared in the (sub)program from which E01AEF is called.

Constraint: LWRK $\geq 7 \times$ N $+ 5 \times ipmax +$ M $+ 7$, where $ipmax$ is the largest element of IP(i), for $i = 1, 2, \ldots,$ M.

13: IWRK(LIWRK) – INTEGER array *Output*

On exit: used as workspace, but see also Section 8.5.

14: LIWRK – INTEGER *Input*

On entry: the dimension of the array IWRK as declared in the (sub)program from which E01AEF is called.

Constraint: LIWRK $\geq 2 \times$ M $+ 2$.

15: 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.

On exit: IFAIL $= 0$ unless the routine detects an error (see Section 6).

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.**

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 N $\neq$ M $+$ IP(1) $+$ IP(2) $+ \cdots +$ IP(M),
or LWRK $< 7 \times$ N $+ 5 \times ipmax +$ M $+ 7$ (see LWRK for the definition of $ipmax$),
or LIWRK $< 2 \times$ M $+ 2$.

IFAIL $= 2$

On entry, IP$(i) < 0$ for some i.

IFAIL $= 3$

On entry, XMIN $\geq$ XMAX,
or X$(i) <$ XMIN for some i,
or X$(i) >$ XMAX,
or X$(i) =$ X(j) for some $i \neq j$.

IFAIL $= 4$

Not all the performance indices are less than eight times the ***machine precision***, although ITMAX iterations have been performed. Parameters A, WRK and IWRK relate to the best polynomial determined. A more accurate solution may possibly be obtained by increasing ITMAX and recalling the routine. See also Sections 7, 8.4 and 8.5.

IFAIL $= 5$

The computation has been terminated because the iterative process appears to be diverging. (Parameters A, WRK and IWRK relate to the best polynomial determined.) Thus the problem specified by your data is probably too ill-conditioned for the solution to be satisfactory. This may result from some of the X(i) being very close together, or from the number of interpolating conditions, N, being large. If in such cases the conditions do not involve derivatives, you are likely to obtain a much more satisfactory solution to your problem either by cubic spline interpolation (see E01BAF) or by curve-fitting with a polynomial or spline in which the number of coefficients is less than N, preferably much less if N is large (see Chapter E02). But see Sections 7, 8.4 and 8.5.

7 Accuracy

A complete error analysis is not currently available, but the method gives good results for reasonable problems.

It is important to realise that for some sets of data, the polynomial interpolation problem is **ill-conditioned**. That is, a **small** perturbation in the data may induce **large** changes in the polynomial, **even in exact arithmetic**. Though by no means the worst example, interpolation by a single polynomial to a large number of function values given at points equally spaced across the range is notoriously ill-conditioned and the polynomial interpolating such a dataset is prone to exhibit enormous oscillations between the data points, especially near the ends of the range. These will be reflected in the Chebyshev coefficients being large compared with the given function values. A more familiar example of ill-conditioning occurs in the solution of certain systems of linear algebraic equations, in which a small change in the elements of the matrix and/or in the components of the right-hand side vector induces a relatively large change in the solution vector. The best that can be achieved in these cases is to make the residual vector small in some sense. If this is possible, the computed solution is exact for a slightly perturbed set of data. Similar considerations apply to the interpolation problem.

The residuals $y_i^{(k)} - q^{(k)}(x_i)$ are available for inspection (see Section 8.5). To assess whether these are reasonable, however, it is necessary to relate them to the largest function and derivative values taken by $q(x)$ over the interval $[x_{\min}, x_{\max}]$. The following performance indices aim to do this. Let the kth derivative of q with respect to the normalized variable $\bar{x}$ be given by the Chebyshev series

$$\tfrac{1}{2}a_0^k T_0(\bar{x}) + a_1^k T_1(\bar{x}) + \cdots + a_{n-1-k}^k T_{n-1-k}(\bar{x}).$$

Let A_k denote the sum of the moduli of these coefficients (this is an upper bound on the kth derivative in the interval and is taken as a measure of the maximum size of this derivative), and define

$$S_k = \max_{i \leq k} A_i.$$

Then if the root-mean-square value of the residuals of $q^{(k)}$, scaled so as to relate to the normalized variable $\bar{x}$, is denoted by r_k, the performance indices are defined by

$$P_k = r_k/S_k, \qquad \text{for } k = 0, 1, \ldots, \max_i(p_i).$$

It is expected that, in reasonable cases, they will all be less than (say) 8 times the ***machine precision*** (this is the accuracy criterion mentioned in Section 3), and in many cases will be of the order of ***machine precision*** or less.

8 Further Comments

8.1 Timing

Computation time is approximately proportional to $it \times n^3$, where *it* is the number of iterations actually used. (See Section 8.5.)

8.2 Divided-difference Strategy

In constructing each new coefficient in the Newton form of the polynomial, a new x_i must be brought into the computation. The x_i chosen is that which yields the smallest new coefficient. This strategy increases the stability of the divided-difference technique, sometimes quite markedly, by reducing errors due to cancellation.

8.3 Conversion to Chebyshev Form

Conversion from the Newton form to Chebyshev series form is effected by evaluating the former at the n values of $\bar{x}$ at which $T_{n-1}(x)$ takes the value ± 1, and then interpolating these n function values by a call of E02AFF, which provides the Chebyshev series representation of the polynomial with very small additional relative error.

8.4 Iterative Refinement

The iterative refinement process is performed as follows.

Firstly, an initial approximation, $q_1(x)$ say, is found by the technique described in Section 3. The rth step of the refinement process then consists of evaluating the residuals of the rth approximation $q_r(x)$, and constructing an interpolant, $dq_r(x)$, to these residuals. The next approximation $q_{r+1}(x)$ to the interpolating polynomial is then obtained as

$$q_{r+1}(x) = q_r(x) + dq_r(x).$$

This completes the description of the rth step.

The iterative process is terminated according to the following criteria. When a polynomial is found whose performance indices (as defined in Section 7) are all less than 8 times the ***machine precision***, the process terminates after ITMIN further iterations (or after a total of ITMAX iterations if that occurs earlier). This will occur in most reasonable problems. The extra iterations are to allow for the possibility of further improvement. If no such polynomial is found, the process terminates after a total of ITMAX iterations. Both these criteria are over-ridden, however, in two special cases. Firstly, if for some value of r the sum of the moduli of the Chebyshev coefficients of $dq_r(x)$ is greater than that of $q_r(x)$, it is concluded that the process is diverging and the process is terminated at once ($q_{r+1}(x)$ is not computed).

Secondly, if at any stage, the performance indices are all computed as zero, again the process is terminated at once.

As the iterations proceed, a record is kept of the best polynomial. Subsequently, at the end of each iteration, the new polynomial replaces the current best polynomial if it satisfies two conditions (otherwise the best polynomial remains unchanged). The first condition is that at least one of its root-mean-square residual values, r_k (see Section 7) is smaller than the corresponding value for the current best polynomial. The second condition takes two different forms according to whether or not the performance indices (see Section 7) of the current best polynomial are all less than 8 times the ***machine precision***. If they are, then the largest performance index of the new polynomial is required to be less than that of the current best polynomial. If they are not, the number of indices which are less than 8 times the ***machine precision*** must

not be smaller than for the current best polynomial. When the iterative process is terminated, it is the polynomial then recorded as best, which is returned to you as $q(x)$.

8.5 Workspace Information

On successful exit, and also if IFAIL $= 4$ or 5 on exit, the following information is contained in the workspace arrays WRK and IWRK:

WRK$(k+1)$, for $k = 0, 1, \ldots, ipmax$ where $ipmax = \max_i p_i$, contains the ratio of p_k, the performance index relating to the kth derivative of the $q(x)$ finally provided, to 8 times the ***machine precision***.

WRK$(ipmax+1+j)$, for $j = 1, 2, \ldots, n$, contains the jth residual, i.e., the value of $y_i^{(k)} - q^{(k)}(x_i)$, where i and k are the appropriate values corresponding to the jth element in the array Y (see the description of Y in Section 5).

IWRK(1) contains the number of iterations actually performed in deriving $q(x)$.

If, on exit, IFAIL $= 4$ or 5, the $q(x)$ finally provided may still be adequate for your requirements. To assess this you should examine the residuals contained in WRK$(ipmax+1+j)$, for $j = 1, 2, \ldots, n$, to see whether they are acceptably small.

9 Example

This example constructs an interpolant $q(x)$ to the following data:

$$\begin{array}{llll} m = 4, & x_{\min} = 2, & x_{\max} = 6, & \\ x_1 = 2, & p_1 = 0, & y_1 = 1, & \\ x_2 = 4, & p_2 = 1, & y_2 = 2, & y_2^{(1)} = -1, \\ x_3 = 5, & p_3 = 0, & y_3 = 1, & \\ x_4 = 6, & p_4 = 2, & y_4 = 2, & y_4^{(1)} = 4, \quad y_4^{(2)} = -2. \end{array}$$

The coefficients in the Chebyshev series representation of $q(x)$ are printed, and also the residuals corresponding to each of the given function and derivative values.

This program is written in a generalized form which can read any number of data-sets.

9.1 Program Text

```
*     E01AEF Example Program Text
*     Mark 20 Revised. NAG Copyright 2001.
*     .. Parameters ..
      INTEGER          MMAX, NMAX, IPMX, LWRK, LIWRK
      PARAMETER        (MMAX=4,NMAX=8,IPMX=2,LWRK=7*NMAX+5*IPMX+MMAX+7,
     +                 LIWRK=2*MMAX+2)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      DOUBLE PRECISION XMAX, XMIN
      INTEGER          I, IFAIL, IP1, IPMAX, IRES, ITMAX, ITMIN, IY, J,
     +                 M, N
*     .. Local Arrays ..
      DOUBLE PRECISION A(NMAX), WRK(LWRK), X(MMAX), Y(NMAX)
      INTEGER          IP(MMAX), IWRK(LIWRK)
*     .. External Subroutines ..
      EXTERNAL         E01AEF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'E01AEF Example Program Results'
*
      ITMIN = -1
      ITMAX = -1
*     Skip heading in data file
      READ (NIN,*)
   20 READ (NIN,*,END=120) M, XMIN, XMAX
      IF (M.GT.0 .AND. M.LE.MMAX) THEN
```

```
         N = 0
         IPMAX = 0
         DO 40 I = 1, M
            READ (NIN,*) IP(I), X(I), (Y(J),J=N+1,N+IP(I)+1)
            IPMAX = MAX(IPMAX,IP(I))
            N = N + IP(I) + 1
   40    CONTINUE
         IF (N.LE.NMAX .AND. IPMAX.LE.IPMX) THEN
            IFAIL = 1
*
            CALL E01AEF(M,XMIN,XMAX,X,Y,IP,N,ITMIN,ITMAX,A,WRK,LWRK,
     +                  IWRK,LIWRK,IFAIL)
*
            WRITE (NOUT,*)
            IF (IFAIL.GE.0) THEN
               IF (IFAIL.EQ.0 .OR. IFAIL.GE.4) THEN
                  WRITE (NOUT,99999)
     +              'Total number of interpolating conditions =', N
                  WRITE (NOUT,*)
                  WRITE (NOUT,*) 'Interpolating polynomial'
                  WRITE (NOUT,*)
                  WRITE (NOUT,*) '   I    Chebyshev Coefficient A(I+1)'
                  DO 60 I = 1, N
                     WRITE (NOUT,99998) I - 1, A(I)
   60             CONTINUE
                  WRITE (NOUT,*)
                  WRITE (NOUT,*) '  X    R   Rth derivative    Residual'
                  IY = 0
                  IRES = IPMAX + 1
                  DO 100 I = 1, M
                     IP1 = IP(I) + 1
                     DO 80 J = 1, IP1
                        IY = IY + 1
                        IRES = IRES + 1
                        IF (J-1.NE.0) THEN
                           WRITE (NOUT,99997) J - 1, Y(IY), WRK(IRES)
                        ELSE
                           WRITE (NOUT,99996) X(I), '   0', Y(IY),
     +                       WRK(IRES)
                        END IF
   80                CONTINUE
  100             CONTINUE
               ELSE
                  WRITE (NOUT,99995) 'E01AEF exits with IFAIL =', IFAIL
               END IF
            ELSE
               WRITE (NOUT,99994) IFAIL
               GO TO 120
            END IF
         END IF
         GO TO 20
      END IF
  120 CONTINUE
*
99999 FORMAT (1X,A,I4)
99998 FORMAT (1X,I4,F20.3)
99997 FORMAT (5X,I4,F12.1,F17.6)
99996 FORMAT (1X,F4.1,A,F12.1,F17.6)
99995 FORMAT (1X,A,I5)
99994 FORMAT (1X,' ** E01AEF returned with IFAIL = ',I5)
      END
```

9.2 Program Data

```
E01AEF Example Program Data
    4       2.0       6.0
    0       2.0       1.0
    1       4.0       2.0      -1.0
    0       5.0       1.0
    2       6.0       2.0       4.0      -2.0
```

9.3 Program Results

```
 E01AEF Example Program Results

 Total number of interpolating conditions =   7

 Interpolating polynomial

    I    Chebyshev Coefficient A(I+1)
    0               9.125
    1              -4.578
    2               0.461
    3               2.852
    4              -2.812
    5               2.227
    6              -0.711

   X    R   Rth derivative    Residual
 2.0    0         1.0         0.000000
 4.0    0         2.0         0.000000
        1        -1.0         0.000000
 5.0    0         1.0         0.000000
 6.0    0         2.0         0.000000
        1         4.0         0.000000
        2        -2.0         0.000000
```