NAG Fortran Library Routine Document

E02CBF

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

E02CBF evaluates a bivariate polynomial from the rectangular array of coefficients in its double Chebyshev-series representation.

2 Specification

```
      SUBROUTINE E02CBF(MFIRST, MLAST, K, L, X, XMIN, XMAX, Y, YMIN, YMAX, FF,
     1                  A, NA, WORK, NWORK, IFAIL)
      INTEGER           MFIRST, MLAST, K, L, NA, NWORK, IFAIL
      real              X(MLAST), XMIN, XMAX, Y, YMIN, YMAX, FF(MLAST), A(NA),
     1                  WORK(NWORK)
```

3 Description

This subroutine evaluates a bivariate polynomial (represented in double Chebyshev form) of degree k in one variable, $\bar{x}$, and degree l in the other, $\bar{y}$. The range of both variables is -1 to $+1$. However, these normalised variables will usually have been derived (as when the polynomial has been computed by E02CAF, for example) from the user's original variables x and y by the transformations

$$\bar{x} = \frac{2x - (x_{\max} + x_{\min})}{(x_{\max} - x_{\min})} \quad \text{and} \quad \bar{y} = \frac{2y - (y_{\max} + y_{\min})}{(y_{\max} - y_{\min})}.$$

(Here $x_{\min}$ and $x_{\max}$ are the ends of the range of x which has been transformed to the range -1 to $+1$ of $\bar{x}$. $y_{\min}$ and $y_{\max}$ are correspondingly for y. See Section 8). For this reason, the subroutine has been designed to accept values of x and y rather than $\bar{x}$ and $\bar{y}$, and so requires values of $x_{\min}$, etc. to be supplied by the user. In fact, for the sake of efficiency in appropriate cases, the routine evaluates the polynomial for a sequence of values of x, all associated with the same value of y.

The double Chebyshev-series can be written as

$$\sum_{i=0}^{k} \sum_{j=0}^{l} a_{ij} T_i(\bar{x}) T_j(\bar{y}),$$

where $T_i(\bar{x})$ is the Chebyshev polynomial of the first kind of degree i and argument $\bar{x}$, and $T_j(\bar{y})$ is similarly defined. However the standard convention, followed in this subroutine, is that coefficients in the above expression which have either i or j zero are written $\frac{1}{2}a_{ij}$, instead of simply a_{ij}, and the coefficient with both i and j zero is written $\frac{1}{4}a_{0,0}$.

The subroutine first forms $c_i = \sum_{j=0}^{l} a_{ij} T_j(\bar{y})$, with $a_{i,0}$ replaced by $\frac{1}{2}a_{i,0}$, for each of $i = 0, 1, \ldots, k$. The value of the double series is then obtained for each value of x, by summing $c_i \times T_i(\bar{x})$, with c_0 replaced by $\frac{1}{2}c_0$, over $i = 0, 1, \ldots, k$. The Clenshaw three term recurrence (Clenshaw (1955)) with modifications due to Reinsch and Gentleman (1969) is used to form the sums.

4 References

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

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: MFIRST – INTEGER *Input*
2: MLAST – INTEGER *Input*

On entry: the index of the first and last x value in the array x at which the evaluation is required respectively (see Section 8).

Constraint: MLAST $\geq$ MFIRST.

3: K – INTEGER *Input*
4: L – INTEGER *Input*

On entry: the degree k of x and l of y, respectively, in the polynomial.

Constraint: K and L ≥ 0.

5: X(MLAST) – ***real*** array *Input*

On entry: X(i), for $i = \text{MFIRST}, \text{MFIRST} + 1, \ldots, \text{MLAST}$, must contain the x values at which the evaluation is required.

Constraint: XMIN $\leq$ X(i) $\leq$ XMAX, for all i.

6: XMIN – ***real*** *Input*
7: XMAX – ***real*** *Input*

On entry: the lower and upper ends, x_{min} and x_{max}, of the range of the variable x (see Section 3).

The values of XMIN and XMAX may depend on the value of y (e.g., when the polynomial has been derived using E02CAF).

Constraint: XMAX $>$ XMIN.

8: Y – ***real*** *Input*

On entry: the value of the y co-ordinate of all the points at which the evaluation is required.

Constraint: YMIN $\leq$ Y $\leq$ YMAX.

9: YMIN – ***real*** *Input*
10: YMAX – ***real*** *Input*

On entry: the lower and upper ends, y_{min} and y_{max}, of the range of the variable y (see Section 3).

Constraint: YMAX $>$ YMIN.

11: FF(MLAST) – ***real*** array *Output*

On exit: FF(i) gives the value of the polynomial at the point (x_i, y), for $i = \text{MFIRST}, \text{MFIRST} + 1, \ldots, \text{MLAST}$.

12: A(NA) – ***real*** array *Input*

On entry: the Chebyshev coefficients of the polynomial. The coefficient a_{ij} defined according to the standard convention (see Section 3) must be in A$(i \times (l + 1) + j + 1)$.

13: NA – INTEGER *Input*

On entry: the dimension of the array A as declared in the (sub)program from which E02CBF is called.

Constraint: NA $\geq (\text{K} + 1) \times (\text{L} + 1)$, the number of coefficients in a polynomial of the specified degree.

14: WORK(NWORK) – ***real*** array *Workspace*

15: NWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which E02CBF is called.

Constraint: NWORK $\geq$ K + 1.

16: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 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, for users 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, MFIRST > MLAST,
or K < 0,
or L < 0,
or NA < (K + 1) × (L + 1),
or NWORK < K + 1.

IFAIL = 2

On entry, YMIN $\geq$ YMAX,
or Y < YMIN,
or Y > YMAX.

IFAIL = 3

On entry, XMIN $\geq$ XMAX,
or X(i) < XMIN, or X(i) > XMAX, for some i = MFIRST, MFIRST + 1, ..., MLAST.

7 Accuracy

The method is numerically stable in the sense that the computed values of the polynomial are exact for a set of coefficients which differ from those supplied by only a modest multiple of ***machine precision***.

8 Further Comments

The time taken by this routine is approximately proportional to $(k+1) \times (m+l+1)$, where m = MLAST − MFIRST + 1, the number of points at which the evaluation is required.

This subroutine is suitable for evaluating the polynomial surface fits produced by the subroutine E02CAF, which provides the ***real*** array A in the required form. For this use, the values of y_{min} and y_{max} supplied to the present subroutine must be the same as those supplied to E02CAF. The same applies to x_{min} and x_{max} if they are independent of y. If they vary with y, their values must be consistent with those supplied to E02CAF (see Section 8 of the document for E02CAF).

The parameters MFIRST and MLAST are intended to permit the selection of a segment of the array X which is to be associated with a particular value of y, when, for example, other segments of X are associated with other values of y. Such a case arises when, after using E02CAF to fit a set of data, the user wishes to evaluate the resulting polynomial at all the data values. In this case, if the parameters X, Y, MFIRST and MLAST of the present routine are set respectively (in terms of parameters of E02CAF) to $\mathrm{X}, \mathrm{Y(S)}, 1 + \sum_{i=1}^{\mathrm{S}-1} \mathrm{M}(i)$ and $\sum_{i=1}^{\mathrm{S}} \mathrm{M}(i)$, the routine will compute values of the polynomial surface at all data points which have Y(S) as their y co-ordinate (from which values the residuals of the fit may be derived).

9 Example

The example program reads data in the following order, using the notation of the parameter list above:

N K L
A(i), for $i = 1, 2, \ldots, (\mathrm{K}+1) \times (\mathrm{L}+1)$
YMIN YMAX
Y(i) M(i) XMIN(i) XMAX(i) X1(i) XM(i), for $i = 1, 2, \ldots, \mathrm{N}$.

For each line $\mathrm{Y} = \mathrm{Y}(i)$ the polynomial is evaluated at M(i) equispaced points between X1(i) and XM(i) inclusive.

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.

```
*     E02CBF Example Program Text.
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          MMAX, KMAX, NWORK, LMAX, NA
      PARAMETER        (MMAX=100,KMAX=9,NWORK=KMAX+1,LMAX=9,NA=(KMAX+1)
     +                 *(LMAX+1))
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             X1, XM, XMAX, XMIN, Y, YMAX, YMIN
      INTEGER          I, IFAIL, J, K, L, M, N, NCOEF
*     .. Local Arrays ..
      real             A(NA), FF(MMAX), WORK(NWORK), X(MMAX)
*     .. External Subroutines ..
      EXTERNAL         E02CBF
*     .. Intrinsic Functions ..
      INTRINSIC        real
*     .. Executable Statements ..
      WRITE (NOUT,*) 'E02CBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
   20 READ (NIN,*,END=100) N, K, L
      IF (K.LE.KMAX .AND. L.LE.LMAX) THEN
         NCOEF = (K+1)*(L+1)
         READ (NIN,*) (A(I),I=1,NCOEF)
         READ (NIN,*) YMIN, YMAX
         DO 80 I = 1, N
            READ (NIN,*) Y, M, XMIN, XMAX, X1, XM
            IF (M.LE.MMAX) THEN
               DO 40 J = 1, M
                  X(J) = X1 + ((XM-X1)*real(J-1))/real(M-1)
   40          CONTINUE
               IFAIL = 0
*
               CALL E02CBF(1,M,K,L,X,XMIN,XMAX,Y,YMIN,YMAX,FF,A,NA,WORK,
     +                     NWORK,IFAIL)
*
               WRITE (NOUT,*)
               WRITE (NOUT,99999) 'Y = ', Y
```

```
               WRITE (NOUT,*)
               WRITE (NOUT,*) '  I     X(I)      Poly(X(I),Y)'
               DO 60 J = 1, M
                  WRITE (NOUT,99998) J, X(J), FF(J)
   60          CONTINUE
            END IF
   80    CONTINUE
         GO TO 20
      END IF
  100 STOP
*
99999 FORMAT (1X,A,e13.4)
99998 FORMAT (1X,I3,1P,2e13.4)
      END
```

9.2 Program Data

```
E02CBF Example Program Data
   3   3   2
15.34820
 5.15073
 0.10140
 1.14719
 0.14419
-0.10464
 0.04901
-0.00314
-0.00699
 0.00153
-0.00033
-0.00022
   0.0         4.0
   1.0         9   0.1          4.5          0.5          4.5
   1.5         8   0.225        4.25         0.5          4.0
   2.0         8   0.4          4.0          0.5          4.0
```

9.3 Program Results

```
 E02CBF Example Program Results

 Y =    0.1000E+01

   I     X(I)      Poly(X(I),Y)
   1   5.0000E-01   2.0812E+00
   2   1.0000E+00   2.1888E+00
   3   1.5000E+00   2.3018E+00
   4   2.0000E+00   2.4204E+00
   5   2.5000E+00   2.5450E+00
   6   3.0000E+00   2.6758E+00
   7   3.5000E+00   2.8131E+00
   8   4.0000E+00   2.9572E+00
   9   4.5000E+00   3.1084E+00

 Y =    0.1500E+01

   I     X(I)      Poly(X(I),Y)
   1   5.0000E-01   2.6211E+00
   2   1.0000E+00   2.7553E+00
   3   1.5000E+00   2.8963E+00
   4   2.0000E+00   3.0444E+00
   5   2.5000E+00   3.2002E+00
   6   3.0000E+00   3.3639E+00
   7   3.5000E+00   3.5359E+00
   8   4.0000E+00   3.7166E+00

 Y =    0.2000E+01

   I     X(I)      Poly(X(I),Y)
   1   5.0000E-01   3.1700E+00
   2   1.0000E+00   3.3315E+00
```

```
   3   1.5000E+00   3.5015E+00
   4   2.0000E+00   3.6806E+00
   5   2.5000E+00   3.8692E+00
   6   3.0000E+00   4.0678E+00
   7   3.5000E+00   4.2769E+00
   8   4.0000E+00   4.4971E+00
```