

## E02DEF – NAG Fortran Library Routine Document

**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

E02DEF calculates values of a bicubic spline from its B-spline representation.

### 2 Specification

```

SUBROUTINE E02DEF(M, PX, PY, X, Y, LAMDA, MU, C, FF, WRK, IWRK,
1          IFAIL)
  INTEGER      M, PX, PY, IWRK(PY-4), IFAIL
  real        X(M), Y(M), LAMDA(PX), MU(PY), C((PX-4)*(PY-4)),
1          FF(M), WRK(PY-4)

```

### 3 Description

This routine calculates values of the bicubic spline  $s(x, y)$  at prescribed points  $(x_r, y_r)$ , for  $r = 1, 2, \dots, m$ , from its augmented knot sets  $\{\lambda\}$  and  $\{\mu\}$  and from the coefficients  $c_{ij}$ , for  $i = 1, 2, \dots, PX - 4$ ;  $j = 1, 2, \dots, PY - 4$ , in its B-spline representation

$$s(x, y) = \sum_{ij} c_{ij} M_i(x) N_j(y).$$

Here  $M_i(x)$  and  $N_j(y)$  denote normalised cubic B-splines, the former defined on the knots  $\lambda_i$  to  $\lambda_{i+4}$  and the latter on the knots  $\mu_j$  to  $\mu_{j+4}$ .

This routine may be used to calculate values of a bicubic spline given in the form produced by E01DAF, E02DAF, E02DCF and E02DDF. It is derived from the routine B2VRE in Anthony *et al.* [1].

### 4 References

- [1] Anthony G T, Cox M G and Hayes J G (1982) *DASL – Data Approximation Subroutine Library* National Physical Laboratory
- [2] Cox M G (1978) The numerical evaluation of a spline from its B-spline representation *J. Inst. Math. Appl.* **21** 135–143

### 5 Parameters

1: M — INTEGER *Input*

*On entry:*  $m$ , the number of points at which values of the spline are required.

*Constraint:*  $M \geq 1$ .

2: PX — INTEGER *Input*

3: PY — INTEGER *Input*

*On entry:* PX and PY must specify the total number of knots associated with the variables  $x$  and  $y$  respectively. They are such that  $PX - 8$  and  $PY - 8$  are the corresponding numbers of interior knots.

*Constraint:*  $PX \geq 8$  and  $PY \geq 8$ .

- 4:** X(M) — *real* array *Input*
- 5:** Y(M) — *real* array *Input*
- On entry:* X and Y must contain  $x_r$  and  $y_r$ , for  $r = 1, 2, \dots, m$ , respectively. These are the co-ordinates of the points at which values of the spline are required. The order of the points is immaterial.
- Constraint:* X and Y must satisfy
- $$\text{LAMDA}(4) \leq X(r) \leq \text{LAMDA}(\text{PX} - 3)$$
- and
- $$\text{MU}(4) \leq Y(r) \leq \text{MU}(\text{PY} - 3), \text{ for } r = 1, 2, \dots, m.$$
- The spline representation is not valid outside these intervals.
- 6:** LAMDA(PX) — *real* array *Input*
- 7:** MU(PY) — *real* array *Input*
- On entry:* LAMDA and MU must contain the complete sets of knots  $\{\lambda\}$  and  $\{\mu\}$  associated with the  $x$  and  $y$  variables respectively.
- Constraint:* the knots in each set must be in non-decreasing order, with  $\text{LAMDA}(\text{PX} - 3) > \text{LAMDA}(4)$  and  $\text{MU}(\text{PY} - 3) > \text{MU}(4)$ .
- 8:** C((PX-4)\*(PY-4)) — *real* array *Input*
- On entry:* C((PY - 4) × (i - 1) + j) must contain the coefficient  $c_{ij}$  described in Section 3, for  $i = 1, 2, \dots, \text{PX} - 4$ ;  $j = 1, 2, \dots, \text{PY} - 4$ .
- 9:** FF(M) — *real* array *Output*
- On exit:* FF( $r$ ) contains the value of the spline at the point  $(x_r, y_r)$ , for  $r = 1, 2, \dots, m$ .
- 10:** WRK(PY-4) — *real* array *Workspace*
- 11:** IWRK(PY-4) — INTEGER array *Workspace*
- 12:** IFAIL — INTEGER *Input/Output*
- On entry:* IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.
- On exit:* IFAIL = 0 unless the routine detects an error (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 detected by the routine:

IFAIL = 1

On entry, M < 1,  
or PY < 8,  
or PX < 8.

IFAIL = 2

On entry, the knots in array LAMDA, or those in array MU, are not in non-decreasing order, or  $\text{LAMDA}(\text{PX} - 3) \leq \text{LAMDA}(4)$ , or  $\text{MU}(\text{PY} - 3) \leq \text{MU}(4)$ .

IFAIL = 3

On entry, at least one of the prescribed points  $(x_r, y_r)$  lies outside the rectangle defined by  $\text{LAMDA}(4)$ ,  $\text{LAMDA}(\text{PX} - 3)$  and  $\text{MU}(4)$ ,  $\text{MU}(\text{PY} - 3)$ .

## 7 Accuracy

The method used to evaluate the B-splines is numerically stable, in the sense that each computed value of  $s(x_r, y_r)$  can be regarded as the value that would have been obtained in exact arithmetic from slightly perturbed B-spline coefficients. See Cox [2] for details.

## 8 Further Comments

Computation time is approximately proportional to the number of points,  $m$ , at which the evaluation is required.

## 9 Example

This program reads in knot sets LAMDA(1),..., LAMDA(PX) and MU(1),..., MU(PY), and a set of bicubic spline coefficients  $c_{ij}$ . Following these are a value for  $m$  and the co-ordinates  $(x_r, y_r)$ , for  $r = 1, 2, \dots, m$ , at which the spline is to be evaluated.

### 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.

```

*      E02DEF Example Program Text
*      Mark 14 Release.  NAG Copyright 1989.
*      .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MMAX, PXMAX, PYMAX
      PARAMETER        (MMAX=20,PXMAX=MMAX,PYMAX=PXMAX)
*      .. Local Scalars ..
      INTEGER          I, IFAIL, M, PX, PY
*      .. Local Arrays ..
      real             C((PXMAX-4)*(PYMAX-4)), FF(MMAX), LAMDA(PXMAX),
+                    MU(PYMAX), WRK(PYMAX-7), X(MMAX), Y(MMAX)
      INTEGER          IWRK(PYMAX-7)
*      .. External Subroutines ..
      EXTERNAL         E02DEF
*      .. Executable Statements ..
      WRITE (NOUT,*) 'E02DEF Example Program Results'
*      Skip heading in data file
      READ (NIN,*)
*      Read PX and PY, the number of knots in the X and Y directions.
      READ (NIN,*) PX, PY
      IF (PX.LE.PXMAX .AND. PY.LE.PYMAX) THEN
*          Read the knots LAMDA(1) .. LAMDA(PX) and MU(1) .. MU(PY).
          READ (NIN,*) (LAMDA(I),I=1,PX)
          READ (NIN,*) (MU(I),I=1,PY)
*          Read C, the bicubic spline coefficients.
          READ (NIN,*) (C(I),I=1,(PX-4)*(PY-4))
*          Read M, the number of spline evaluation points.
          READ (NIN,*) M
          IF (M.LE.MMAX) THEN
*              Read the X and Y co-ordinates of the evaluation points.
              DO 20 I = 1, M
                  READ (NIN,*) X(I), Y(I)
20          CONTINUE
              IFAIL = 0

```

```

*
*       Evaluate the spline at the M points.
*       CALL E02DEF(M,PX,PY,X,Y,LAMDA,MU,C,FF,WRK,IWRK,IFAIL)
*
*       Print the results.
*       WRITE (NOUT,*)
*       WRITE (NOUT,*) '      I      X(I)      Y(I)      FF(I)'
*       WRITE (NOUT,99999) (I,X(I),Y(I),FF(I),I=1,M)
*       END IF
*       END IF
*       STOP
*
* 99999 FORMAT (1X,I7,3F11.3)
*       END

```

## 9.2 Program Data

E02DEF Example Program Data

11	10										PX	PY
1.0	1.0	1.0	1.0	1.3	1.5	1.6	2.0	2.0	2.0	2.0	LAMDA(1) ..	LAMDA(PX)
0.0	0.0	0.0	0.0	0.4	0.7	1.0	1.0	1.0	1.0	1.0	MU(1) ..	MU(PY)
1.0000	1.1333	1.3667	1.7000	1.9000	2.0000							
1.2000	1.3333	1.5667	1.9000	2.1000	2.2000							
1.5833	1.7167	1.9500	2.2833	2.4833	2.5833							
2.1433	2.2767	2.5100	2.8433	3.0433	3.1433							
2.8667	3.0000	3.2333	3.5667	3.7667	3.8667							
3.4667	3.6000	3.8333	4.1667	4.3667	4.4667							
4.0000	4.1333	4.3667	4.7000	4.9000	5.0000							
7											Spline coefficients, C	
1.0	0.0										M	
1.1	0.1										X(1), Y(1)	
1.5	0.7											
1.6	0.4											
1.9	0.3											
1.9	0.8											
2.0	1.0										X(M), Y(M)	

## 9.3 Program Results

E02DEF Example Program Results

I	X(I)	Y(I)	FF(I)
1	1.000	0.000	1.000
2	1.100	0.100	1.310
3	1.500	0.700	2.950
4	1.600	0.400	2.960
5	1.900	0.300	3.910
6	1.900	0.800	4.410
7	2.000	1.000	5.000