NAG Library Routine Document

E01BAF

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

E01BAF determines a cubic spline interpolant to a given set of data.

2 Specification

```
SUBROUTINE E01BAF (M, X, Y, LAMDA, C, LCK, WRK, LWRK, IFAIL)
INTEGER              M, LCK, LWRK, IFAIL
REAL (KIND=nag_wp) X(M), Y(M), LAMDA(LCK), C(LCK), WRK(LWRK)
```

3 Description

E01BAF determines a cubic spline $s(x)$, defined in the range $x_1 \leq x \leq x_m$, which interpolates (passes exactly through) the set of data points (x_i, y_i), for $i = 1, 2, \ldots, m$, where $m \geq 4$ and $x_1 < x_2 < \cdots < x_m$. Unlike some other spline interpolation algorithms, derivative end conditions are not imposed. The spline interpolant chosen has $m - 4$ interior knots $\lambda_5, \lambda_6, \ldots, \lambda_m$, which are set to the values of $x_3, x_4, \ldots, x_{m-2}$ respectively. This spline is represented in its B-spline form (see Cox (1975)):

$$s(x) = \sum_{i=1}^{m} c_i N_i(x),$$

where $N_i(x)$ denotes the normalized B-spline of degree 3, defined upon the knots $\lambda_i, \lambda_{i+1}, \ldots, \lambda_{i+4}$, and c_i denotes its coefficient, whose value is to be determined by the routine.

The use of B-splines requires eight additional knots λ_1, λ_2, λ_3, λ_4, λ_{m+1}, λ_{m+2}, λ_{m+3} and λ_{m+4} to be specified; E01BAF sets the first four of these to x_1 and the last four to x_m.

The algorithm for determining the coefficients is as described in Cox (1975) except that QR factorization is used instead of LU decomposition. The implementation of the algorithm involves setting up appropriate information for the related routine E02BAF followed by a call of that routine. (See E02BAF for further details.)

Values of the spline interpolant, or of its derivatives or definite integral, can subsequently be computed as detailed in Section 9.

4 References

Cox M G (1975) An algorithm for spline interpolation *J. Inst. Math. Appl.* **15** 95–108

Cox M G (1977) A survey of numerical methods for data and function approximation *The State of the Art in Numerical Analysis* (ed D A H Jacobs) 627–668 Academic Press

5 Parameters

1: M – INTEGER *Input*

On entry: m, the number of data points.

Constraint: M ≥ 4.

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

On entry: $\mathrm{X}(i)$ must be set to x_i, the ith data value of the independent variable x, for $i = 1, 2, \ldots, m$.

Constraint: $\mathrm{X}(i) < \mathrm{X}(i+1)$, for $i = 1, 2, \ldots, \mathrm{M} - 1$.

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

On entry: $\mathrm{Y}(i)$ must be set to y_i, the ith data value of the dependent variable y, for $i = 1, 2, \ldots, m$.

4: LAMDA(LCK) – REAL (KIND=nag_wp) array *Output*

On exit: the value of λ_i, the ith knot, for $i = 1, 2, \ldots, m + 4$.

5: C(LCK) – REAL (KIND=nag_wp) array *Output*

On exit: the coefficient c_i of the B-spline $N_i(x)$, for $i = 1, 2, \ldots, m$. The remaining elements of the array are not used.

6: LCK – INTEGER *Input*

On entry: the dimension of the arrays LAMDA and C as declared in the (sub)program from which E01BAF is called.

Constraint: $\mathrm{LCK} \geq \mathrm{M} + 4$.

7: WRK(LWRK) – REAL (KIND=nag_wp) array *Workspace*

8: LWRK – INTEGER *Input*

On entry: the dimension of the array WRK as declared in the (sub)program from which E01BAF is called.

Constraint: $\mathrm{LWRK} \geq 6 \times \mathrm{M} + 16$.

9: 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.

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.**

On exit: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry $\mathrm{IFAIL} = 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = 1$

On entry, $\mathrm{M} < 4$,
or $\mathrm{LCK} < \mathrm{M} + 4$,
or $\mathrm{LWRK} < 6 \times \mathrm{M} + 16$.

IFAIL $= 2$

The X-values fail to satisfy the condition

$X(1) < X(2) < X(3) < \cdots < X(M)$.

IFAIL $= -99$

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

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

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

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

The rounding errors incurred are such that the computed spline is an exact interpolant for a slightly perturbed set of ordinates $y_i + \delta y_i$. The ratio of the root-mean-square value of the δy_i to that of the y_i is no greater than a small multiple of the relative ***machine precision***.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by E01BAF is approximately proportional to m.

All the x_i are used as knot positions except x_2 and x_{m-1}. This choice of knots (see Cox (1977)) means that $s(x)$ is composed of $m - 3$ cubic arcs as follows. If $m = 4$, there is just a single arc space spanning the whole interval x_1 to x_4. If $m \geq 5$, the first and last arcs span the intervals x_1 to x_3 and x_{m-2} to x_m respectively. Additionally if $m \geq 6$, the ith arc, for $i = 2, 3, \ldots, m - 4$, spans the interval x_{i+1} to x_{i+2}.

After the call

```
CALL E01BAF (M, X, Y, LAMDA, C, LCK, WRK, LWRK, IFAIL)
```

the following operations may be carried out on the interpolant $s(x)$.

The value of $s(x)$ at $x = \mathrm{X}$ can be provided in the real variable S by the call

```
CALL E02BBF (M+4, LAMDA, C, X, S, IFAIL)
```

(see E02BBF).

The values of $s(x)$ and its first three derivatives at $x = \mathrm{X}$ can be provided in the real array S of dimension 4, by the call

```
CALL E02BCF (M+4, LAMDA, C, X, LEFT, S, IFAIL)
```

(see E02BCF).

Here LEFT must specify whether the left- or right-hand value of the third derivative is required (see E02BCF for details).

The value of the integral of $s(x)$ over the range x_1 to x_m can be provided in the real variable DINT by

```
CALL E02BDF (M+4, LAMDA, C, DINT, IFAIL)
```

(see E02BDF).

10 Example

This example sets up data from 7 values of the exponential function in the interval 0 to 1. E01BAF is then called to compute a spline interpolant to these data.

The spline is evaluated by E02BBF, at the data points and at points halfway between each adjacent pair of data points, and the spline values and the values of e^x are printed out.

10.1 Program Text

```
    Program e01bafe

!     E01BAF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: e01baf, e02bbf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: m = 7, nout = 6
      Integer, Parameter               :: lck = m + 4
      Integer, Parameter               :: lwrk = 6*m + 16
      Real (Kind=nag_wp), Parameter    :: x(m) = (/0.0E0_nag_wp,0.2E0_nag_wp,  &
            0.4E0_nag_wp,0.6E0_nag_wp,0.75E0_nag_wp,0.9E0_nag_wp,1.0E0_nag_wp/)
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: fit, xarg
      Integer                          :: ifail, j, r
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: c(lck), lamda(lck), wrk(lwrk), y(m)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: exp
!     .. Executable Statements ..
      Write (nout,*) 'E01BAF Example Program Results'

      y(1:m) = exp(x(1:m))

      ifail = 0
      Call e01baf(m,x,y,lamda,c,lck,wrk,lwrk,ifail)

      Write (nout,*)
      Write (nout,*) '   J    Knot LAMDA(J+2)   B-spline coeff C(J)'
      Write (nout,*)

      j = 1
      Write (nout,99998) j, c(1)

      Do j = 2, m - 1
        Write (nout,99999) j, lamda(j+2), c(j)
      End Do

      Write (nout,99998) m, c(m)
      Write (nout,*)
      Write (nout,*) &
        '   R        Abscissa             Ordinate             Spline'
      Write (nout,*)

      Do r = 1, m

        ifail = 0
        Call e02bbf(m+4,lamda,c,x(r),fit,ifail)

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

        If (r<m) Then
          xarg = 0.5E0_nag_wp*(x(r)+x(r+1))

          ifail = 0
```

```
          Call e02bbf(m+4,lamda,c,xarg,fit,ifail)

          Write (nout,99997) xarg, fit
        End If

      End Do

99999 Format (1X,I4,F15.4,2F20.4)
99998 Format (1X,I4,F35.4)
99997 Format (1X,F19.4,F40.4)
    End Program e01bafe
```

10.2 Program Data

None.

10.3 Program Results

```
 E01BAF Example Program Results

    J    Knot LAMDA(J+2)   B-spline coeff C(J)

    1                             1.0000
    2         0.0000              1.1336
    3         0.4000              1.3726
    4         0.6000              1.7827
    5         0.7500              2.1744
    6         1.0000              2.4918
    7                             2.7183

    R        Abscissa           Ordinate              Spline

    1         0.0000              1.0000              1.0000
              0.1000                                  1.1052
    2         0.2000              1.2214              1.2214
              0.3000                                  1.3498
    3         0.4000              1.4918              1.4918
              0.5000                                  1.6487
    4         0.6000              1.8221              1.8221
              0.6750                                  1.9640
    5         0.7500              2.1170              2.1170
              0.8250                                  2.2819
    6         0.9000              2.4596              2.4596
              0.9500                                  2.5857
    7         1.0000              2.7183              2.7183
```