E01SBF – 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

E01SBF evaluates at a given point the two-dimensional interpolant function computed by E01SAF.

2 Specification

```
SUBROUTINE E01SBF(M, X, Y, F, TRIANG, GRADS, PX, PY, PF, IFAIL)
INTEGER          M, TRIANG(7*M), IFAIL
real             X(M), Y(M), F(M), GRADS(2,M), PX, PY, PF
```

3 Description

This routine takes as input the parameters defining the interpolant $F(x,y)$ of a set of scattered data points (x_r, y_r, f_r), for $r = 1,2,\ldots,m$, as computed by E01SAF, and evaluates the interpolant at the point (px, py).

If (px, py) is equal to (x_r, y_r) for some value of r, the returned value will be equal to f_r.

If (px, py) is not equal to (x_r, y_r) for any r, the derivatives in GRADS will be used to compute the interpolant. A triangle is sought which contains the point (px, py), and the vertices of the triangle along with the partial derivatives and f_r values at the vertices are used to compute the value $F(px, py)$. If the point (px, py) lies outside the triangulation defined by the input parameters, the returned value is obtained by extrapolation. In this case, the interpolating function F is extended linearly beyond the triangulation boundary. The method is described in more detail in Renka and Cline [2] and the code is derived from Renka [1].

E01SBF must only be called after a call to E01SAF.

4 References

[1] Renka R L (1984) Algorithm 624: Triangulation and interpolation of arbitrarily distributed points in the plane *ACM Trans. Math. Software* **10** 440–442

[2] Renka R L and Cline A K (1984) A triangle-based C^1 interpolation method *Rocky Mountain J. Math.* **14** 223–237

5 Parameters

1: M — INTEGER *Input*

2: X(M) — ***real*** array *Input*

3: Y(M) — ***real*** array *Input*

4: F(M) — ***real*** array *Input*

5: TRIANG(7∗M) — INTEGER array *Input*

6: GRADS(2,M) — ***real*** array *Input*

On entry: M, X, Y, F, TRIANG and GRADS must be unchanged from the previous call of E01SAF.

7: PX — ***real*** *Input*

8: PY — ***real*** *Input*

On entry: the point (px, py) at which the interpolant is to be evaluated.

9: PF — ***real*** *Output*

On exit: the value of the interpolant evaluated at the point (px, py).

10: 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 < 3.

IFAIL = 2

On entry, the triangulation information held in the array TRIANG does not specify a valid triangulation of the data points. TRIANG may have been corrupted since the call to E01SAF.

IFAIL = 3

The evaluation point (PX,PY) lies outside the nodal triangulation, and the value returned in PF is computed by extrapolation.

7 Accuracy

Computational errors should be negligible in most practical situations.

8 Further Comments

The time taken for a call of E01SBF is approximately proportional to the number of data points, m.

The results returned by this routine are particularly suitable for applications such as graph plotting, producing a smooth surface from a number of scattered points.

9 Example

See the example for Section 9 of the document for E01SAF.