NAG Library Routine Document

E01SBF

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 (KIND=nag_wp) X(M), Y(M), F(M), GRADS(2,M), PX, PY, PF
```

3 Description

E01SBF takes as input the arguments 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 arguments, 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 (1984) and the code is derived from Renka (1984).

E01SBF must only be called after a call to E01SAF.

4 References

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

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

5 Arguments

1: M – INTEGER *Input*
2: X(M) – REAL (KIND=nag_wp) array *Input*
3: Y(M) – REAL (KIND=nag_wp) array *Input*
4: F(M) – REAL (KIND=nag_wp) array *Input*
5: TRIANG($7 \times$ M) – INTEGER array *Input*
6: GRADS(2, M) – REAL (KIND=nag_wp) array *Input*

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

7: PX – REAL (KIND=nag_wp) *Input*
8: PY – REAL (KIND=nag_wp) *Input*

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

9: PF – REAL (KIND=nag_wp) *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. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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 argument, 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: 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 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 < 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.

IFAIL $= -99$

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

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

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

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

Computational errors should be negligible in most practical situations.

8 Parallelism and Performance

E01SBF is not thread safe and should not be called from a multithreaded user program. Please see Section 3.12.1 in How to Use the NAG Library and its Documentation for more information on thread safety.

E01SBF is not threaded in any implementation.

9 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.

10 Example

See Section 10 in E01SAF.