nag_2d_scat_eval (e01sbc)

1. Purpose

nag_2d_scat_eval (e01sbc) evaluates at given points the two-dimensional interpolant function computed by nag_2d_scat_interpolant (e01sac).

2. Specification

```c
#include <nag.h>
#include <nage01.h>

void nag_2d_scat_eval(Nag_Scat_Struct *comm, Integer n, double px[],
                      double py[], double pf[], NagError *fail)
```

3. Description

This function 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 nag_2d_scat_interpolant (e01sac), and evaluates the interpolant at each of the points \( (px_k, py_k) \), for \( k = 1, 2, \ldots, n \).

When \textit{method} = \texttt{Nag_RC}, the derivatives stored in \texttt{comm} will be used to compute the interpolant if necessary. A triangle is sought which contains the point \( (px_k, py_k) \), and the vertices of the triangle along with the partial derivatives and \( f_r \) values at the vertices are used to compute the value \( F(px_k, py_k) \). If the point \( (px_k, py_k) \) 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 (1984) and the code is derived from Renka (1984).

Alternatively, if \textit{method} = \texttt{Nag_Shep}, then all points that are within distance of \( (px_k, py_k) \), along with the corresponding nodal functions stored in \texttt{comm}, will be used to compute a value of the interpolant, if necessary.

\texttt{nag_2d_scat_eval} must only be called after a call to \texttt{nag_2d_scat_interpolant} (e01sac).

4. Parameters

- \texttt{comm}
  
  Pointer to a communication structure of type \texttt{Nag_Scat_Struct} which must be unchanged from the previous call of \texttt{nag_2d_scat_interpolant} (e01sac).

- \texttt{n}
  
  Input: the number of points at which the evaluation of the interpolant is required. Constraint: \( n \geq 1 \).

- \texttt{px[n]}
  
  Input: the \( x \)- and \( y \)-coordinates of the \( k \)th point \( (px_k, py_k) \), for \( k = 1, 2, \ldots, n \), at which the interpolant is to be evaluated.

- \texttt{py[n]}
  
  Input: the \( x \)- and \( y \)-coordinates of the \( k \)th point \( (px_k, py_k) \), for \( k = 1, 2, \ldots, n \), at which the interpolant is to be evaluated.

- \texttt{pf[n]}
  
  Output: the values of the interpolant evaluated at the points \( (px_k, py_k) \), for \( k = 1, 2, \ldots, n \).

- \texttt{fail}
  
  The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

- **NE_NO_SETUP**
  
  The setup function \texttt{nag_2d_scat_interpolant} (e01sac) has not been called.

- **NE_SETUP_ERROR**
  
  The call to setup function \texttt{nag_2d_scat_interpolant} (e01sac) produced an error.
6. Further Comments

The time taken for a call of nag_2d_scat_eval is approximately proportional to the number of data points, \(m\), used by nag_2d_scat_interpolant (e01sac).

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

6.1. Accuracy

Computational errors should be negligible in most practical situations.

6.2. References


7. See Also

nag_2d_scat_interpolant (e01sac)

8. Example

See the example program for nag_2d_scat_interpolant (e01sac).