NAG Fortran Library Routine Document

D03MAF

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

D03MAF places a triangular mesh over a given two-dimensional region. The region may have any shape, including one with holes.

2 Specification

```
 SUBROUTINE D03MAF(H, M, N, NB, NPTS, PLACES, INDEX, IDIM, IN, DIST, LD,
1                  IFAIL)
 INTEGER           M, N, NB, NPTS, INDEX(4,IDIM), IDIM, IN, LD, IFAIL
 real              H, PLACES(2,IDIM), DIST(4,LD)
 EXTERNAL          IN
```

3 Description

This subroutine begins with a uniform triangular grid as shown in Figure 1 and assumes that the region to be triangulated lies within the rectangle given by the inequalities

$$0 < x < \sqrt{3}(m-1)h, \quad 0 < y < (n-1)h.$$

This rectangle is drawn in bold in Figure 1. The region is specified by the user's function IN which must determine whether any given point (x, y) lies in the region. The uniform grid is processed columnwise, with (x_1, y_1) preceding (x_2, y_2) if $x_1 < x_2$ or $x_1 = x_2$, $y_1 < y_2$. Points near the boundary are moved onto it and points well outside the boundary are omitted. The direction of movement is chosen to avoid pathologically thin triangles. The points accepted are numbered in exactly the same order as the corresponding points of the uniform grid were scanned. The output consists of the x, y co-ordinates of all grid points and integers indicating whether they are internal and to which other points they are joined by triangle sides.

The mesh size h must be chosen small enough for the essential features of the region to be apparent from testing all points of the original uniform grid for being inside the region. For instance if any hole is within $2h$ of another hole or the outer boundary then a triangle may be found with all vertices within $\frac{1}{2}h$ of a boundary. Such a triangle is taken to be external to the region so the effect will be to join the hole to another hole or to the external region.

Further details of the algorithm are given in the references.

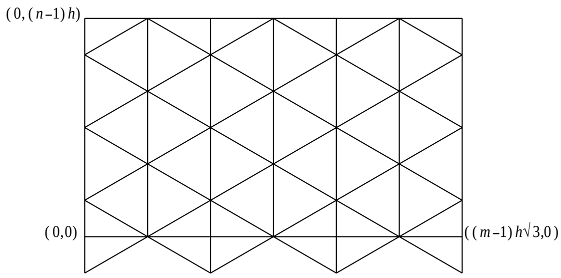

Figure 1

4 References

Reid J K (1970) Fortran subroutines for the solutions of Laplace's equation over a general routine in two dimensions *Harwell Report TP422*

Reid J K (1972) On the construction and convergence of a finite-element solution of Laplace's equation *J. Instr. Math. Appl.* **9** 1–13

5 Parameters

1: H – ***real*** *Input*

On entry: the required length, h, for the sides of the triangles of the uniform mesh.

2: M – INTEGER *Input*
3: N – INTEGER *Input*

On entry: values m and n such that all points (x, y) inside the region satisfy the inequalities

$$\begin{array}{ccccc} 0 & \leq & x & \leq & \sqrt{3}(m-1)h, \\ 0 & \leq & y & \leq & (n-1)h. \end{array}$$

Constraint: M, N > 2.

4: NB – INTEGER *Input*

On entry: the number of times a triangle side is bisected to find a point on the boundary. A value of 10 is adequate for most purposes (see Section 7).

Constraint: NB ≥ 1.

5: NPTS – INTEGER *Output*

On exit: the number of points in the triangulation.

6: PLACES(2,IDIM) – ***real*** array *Output*

On exit: the x and y co-ordinates respectively of the ith point of the triangulation.

7: INDEX(4,IDIM) – INTEGER array *Output*

On exit: INDEX$(1, i)$ contains i if point i is inside the region and $-i$ if it is on the boundary. For each triangle side between points i and j with $j > i$, INDEX(k, i), $k > 1$, contains j or $-j$ according to whether point j is internal or on the boundary. There can never be more than three such points. If there are less, then some values INDEX(k, i), $k > 1$, are zero.

8: IDIM – INTEGER *Input*

On entry: the second dimension of the arrays PLACES and INDEX as declared in the (sub)program from which D03MAF is called.

Constraint: IDIM $\geq$ NPTS.

9: IN – INTEGER FUNCTION, supplied by the user. *External Procedure*

IN must return the value 1 if the given point (X,Y) lies inside the region, and 0 if it lies outside.

Its specification is:

```
INTEGER FUNCTION IN(X, Y)
real                X, Y
```

1: X – ***real*** *Input*

2: Y – ***real*** *Input*

On entry: the co-ordinates of the given point.

IN must be declared as EXTERNAL in the (sub)program from which D03MAF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

10: DIST(4,LD) – ***real*** array *Workspace*

11: LD – INTEGER *Input*

On entry: the second dimension of the array DIST as declared in the (sub)program from which D03MAF is called.

Constraint: LD $\geq$ 4N.

12: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL $= 0$ unless the routine detects an error (see Section 6).

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, for users 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.**

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 $= 1$

IDIM is too small.

IFAIL $= 2$

A point inside the region violates one of the constraints (see parameters M and N above).

IFAIL $= 3$

LD is too small.

IFAIL $= 4$

M $\leq$ 2.

IFAIL $= 5$

N $\leq$ 2.

IFAIL $= 6$

NB $\leq$ 0.

7 Accuracy

Points are moved onto the boundary by bisecting a triangle side NB times. The accuracy is therefore $h \times 2^{-\mathrm{NB}}$.

8 Further Comments

The time taken by the routine is approximately proportional to $m \times n$.

9 Example

The following program triangulates the circle with centre (7.0,7.0) and radius 6.0 using a basic grid size $h = 4.0$.

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.

```
*     D03MAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          IDIM, LD
      PARAMETER        (IDIM=100,LD=20)
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Local Scalars ..
      real             H
      INTEGER          I, IFAIL, J, M, N, NB, NPTS
*     .. Local Arrays ..
      real             DIST(4,LD), PLACES(2,IDIM)
      INTEGER          INDEX(4,IDIM)
*     .. External Functions ..
      INTEGER          IN1
      EXTERNAL         IN1
*     .. External Subroutines ..
      EXTERNAL         D03MAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D03MAF Example Program Results'
      WRITE (NOUT,*)
      H = 4.0e0
      M = 3
      N = 5
      NB = 10
      IFAIL = 0
*
      CALL D03MAF(H,M,N,NB,NPTS,PLACES,INDEX,IDIM,IN1,DIST,LD,IFAIL)
*
      WRITE (NOUT,*) '  I    X(I)      Y(I)'
      DO 20 I = 1, NPTS
         WRITE (NOUT,99999) I, PLACES(1,I), PLACES(2,I)
   20 CONTINUE
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Index'
      DO 40 I = 1, NPTS
         WRITE (NOUT,99998) (INDEX(J,I),J=1,4)
   40 CONTINUE
      STOP
*
99999 FORMAT (1X,I3,2F10.6)
99998 FORMAT (1X,4I5)
      END
*
      INTEGER FUNCTION IN1(X,Y)
*     Circular domain
*     .. Scalar Arguments ..
```

```
      real                    X, Y
*     .. Executable Statements ..
      IF ((X-7.0e0)**2+(Y-7.0e0)**2.LE.36.0e0) THEN
         IN1 = 1
      ELSE
         IN1 = 0
      END IF
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 D03MAF Example Program Results

   I    X(I)      Y(I)
   1  1.013182  6.584961
   2  1.412366  9.184570
   3  2.268242  3.309570
   4  3.464102  8.000000
   5  3.584195 11.930664
   6  6.928203  1.001953
   7  6.928203  6.000000
   8  6.928203 10.000000
   9  6.928203 12.998047
  10 11.686269  3.252930
  11 10.392305  8.000000
  12 10.392305 11.947266
  13 12.978541  6.506836
  14 12.562443  9.252930

 Index
    -1   -3    4   -2
    -2    4   -5    0
    -3   -6    7    4
     4    7    8   -5
    -5    8   -9    0
    -6    0  -10    7
     7  -10   11    8
     8   11  -12   -9
    -9  -12    0    0
   -10    0  -13   11
    11  -13  -14  -12
   -12  -14    0    0
   -13    0    0  -14
   -14    0    0    0
```