NAG Library Routine Document

D06AAF

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

D06AAF generates a triangular mesh of a closed polygonal region in $\mathbb{R}^2$, given a mesh of its boundary. It uses a simple incremental method.

2 Specification

```
SUBROUTINE D06AAF (NVB, NVMAX, NEDGE, EDGE, NV, NELT, COOR, CONN, BSPACE,      &
                   SMOOTH, COEF, POWER, ITRACE, RWORK, LRWORK, IWORK,          &
                   LIWORK, IFAIL)

INTEGER            NVB, NVMAX, NEDGE, EDGE(3,NEDGE), NV, NELT,                 &
                   CONN(3,2*(NVMAX-1)), ITRACE, LRWORK, IWORK(LIWORK),         &
                   LIWORK, IFAIL
REAL (KIND=nag_wp) COOR(2,NVMAX), BSPACE(NVB), COEF, POWER, RWORK(LRWORK)
LOGICAL            SMOOTH
```

3 Description

D06AAF generates the set of interior vertices using a process based on a simple incremental method. A smoothing of the mesh is optionally available. For more details about the triangulation method, consult the D06 Chapter Introduction as well as George and Borouchaki (1998).

This routine is derived from material in the MODULEF package from INRIA (Institut National de Recherche en Informatique et Automatique).

4 References

George P L and Borouchaki H (1998) *Delaunay Triangulation and Meshing: Application to Finite Elements* Editions HERMES, Paris

5 Parameters

1: NVB – INTEGER *Input*

On entry: the number of vertices in the input boundary mesh.

Constraint: $3 \leq \text{NVB} \leq \text{NVMAX}$.

2: NVMAX – INTEGER *Input*

On entry: the maximum number of vertices in the mesh to be generated.

3: NEDGE – INTEGER *Input*

On entry: the number of boundary edges in the input mesh.

Constraint: $\text{NEDGE} \geq 1$.

4: EDGE(3,NEDGE) – INTEGER array *Input*

On entry: the specification of the boundary edges. $\text{EDGE}(1,j)$ and $\text{EDGE}(2,j)$ contain the vertex numbers of the two end points of the jth boundary edge. $\text{EDGE}(3,j)$ is a user-supplied tag for the jth boundary edge and is not used by D06AAF.

Constraint: $1 \leq \text{EDGE}(i,j) \leq \text{NVB}$ and $\text{EDGE}(1,j) \neq \text{EDGE}(2,j)$, for $i = 1,2$ and $j = 1,2,\ldots,\text{NEDGE}$.

5: NV – INTEGER *Output*

On exit: the total number of vertices in the output mesh (including both boundary and interior vertices). If $\text{NVB} = \text{NVMAX}$, no interior vertices will be generated and $\text{NV} = \text{NVB}$.

6: NELT – INTEGER *Output*

On exit: the number of triangular elements in the mesh.

7: COOR(2,NVMAX) – REAL (KIND=nag_wp) array *Input/Output*

On entry: $\text{COOR}(1,i)$ contains the x coordinate of the ith input boundary mesh vertex; while $\text{COOR}(2,i)$ contains the corresponding y coordinate, for $i = 1,2,\ldots,\text{NVB}$.

On exit: $\text{COOR}(1,i)$ will contain the x coordinate of the $(i - \text{NVB})$th generated interior mesh vertex; while $\text{COOR}(2,i)$ will contain the corresponding y coordinate, for $i = \text{NVB}+1,\ldots,\text{NV}$. The remaining elements are unchanged.

8: CONN($3,2 \times (\text{NVMAX} - 1)$) – INTEGER array *Output*

On exit: the connectivity of the mesh between triangles and vertices. For each triangle j, $\text{CONN}(i,j)$ gives the indices of its three vertices (in anticlockwise order), for $i = 1,2,3$ and $j = 1,2,\ldots,\text{NELT}$.

9: BSPACE(NVB) – REAL (KIND=nag_wp) array *Input*

On entry: the desired mesh spacing (triangle diameter, which is the length of the longer edge of the triangle) near the boundary vertices.

Constraint: $\text{BSPACE}(i) > 0.0$, for $i = 1,2,\ldots,\text{NVB}$.

10: SMOOTH – LOGICAL *Input*

On entry: indicates whether or not mesh smoothing should be performed.

If SMOOTH = .TRUE., the smoothing is performed; otherwise no smoothing is performed.

11: COEF – REAL (KIND=nag_wp) *Input*

On entry: the coefficient in the stopping criteria for the generation of interior vertices. This parameter controls the triangle density and the number of triangles generated is in $O(\text{COEF}^2)$. The mesh will be finer if COEF is greater than 0.7165 and 0.75 is a good value.

Suggested value: 0.75.

12: POWER – REAL (KIND=nag_wp) *Input*

On entry: controls the rate of change of the mesh size during the generation of interior vertices. The smaller the value of POWER, the faster the decrease in element size away from the boundary.

Suggested value: 0.25.

Constraint: $0.1 \leq \text{POWER} \leq 10.0$.

13: ITRACE – INTEGER *Input*

On entry: the level of trace information required from D06AAF.

ITRACE ≤ 0
: No output is generated.

ITRACE ≥ 1
: Output from the meshing solver is printed on the current advisory message unit (see X04ABF). This output contains details of the vertices and triangles generated by the process.

You are advised to set ITRACE $= 0$, unless you are experienced with finite element mesh generation.

14: RWORK(LRWORK) – REAL (KIND=nag_wp) array *Workspace*
15: LRWORK – INTEGER *Input*

On entry: the dimension of the array RWORK as declared in the (sub)program from which D06AAF is called.

Constraint: LRWORK $\geq$ NVMAX.

16: IWORK(LIWORK) – INTEGER array *Workspace*
17: LIWORK – INTEGER *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D06AAF is called.

Constraint: LIWORK $\geq 16 \times \mathrm{NVMAX} + 2 \times \mathrm{NEDGE} + \max(4 \times \mathrm{NVMAX} + 2, \mathrm{NEDGE}) - 14$.

18: 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: 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, NVB < 3 or NVB $>$ NVMAX,
or NEDGE < 1,
or EDGE$(i,j) < 1$ or EDGE$(i,j) >$ NVB, for some $i = 1, 2$ and $j = 1, 2, \ldots, \mathrm{NEDGE}$,
or EDGE$(1,j) =$ EDGE$(2,j)$, for some $j = 1, 2, \ldots, \mathrm{NEDGE}$,
or BSPACE$(i) \leq 0.0$, for some $i = 1, 2, \ldots, \mathrm{NVB}$,
or POWER < 0.1 or POWER > 10.0,
or LIWORK $< 16 \times \mathrm{NVMAX} + 2 \times \mathrm{NEDGE} + \max(4 \times \mathrm{NVMAX} + 2, \mathrm{NEDGE}) - 14$,
or LRWORK $<$ NVMAX.

IFAIL $= 2$

An error has occurred during the generation of the interior mesh. Check the definition of the boundary (arguments COOR and EDGE) as well as the orientation of the boundary (especially in the case of a multiple connected component boundary). Setting ITRACE > 0 may provide more details.

7 Accuracy

Not applicable.

8 Further Comments

The position of the internal vertices is a function of the positions of the vertices on the given boundary. A fine mesh on the boundary results in a fine mesh in the interior. The algorithm allows you to obtain a denser interior mesh by varying NVMAX, BSPACE, COEF and POWER. But you are advised to manipulate the last two parameters with care.

You are advised to take care to set the boundary inputs properly, especially for a boundary with multiply connected components. The orientation of the interior boundaries should be in **clockwise** order and opposite to that of the exterior boundary. If the boundary has only one connected component, its orientation should be **anticlockwise**.

9 Example

In this example, a geometry with two holes (two interior circles inside an exterior one) is meshed using the simple incremental method (see the D06 Chapter Introduction). The exterior circle is centred at the origin with a radius 1.0, the first interior circle is centred at the point $(-0.5, 0.0)$ with a radius 0.49, and the second one is centred at the point $(-0.5, 0.65)$ with a radius 0.15. Note that the points $(-1.0, 0.0)$ and $(-0.5, 0.5)$) are points of 'near tangency' between the exterior circle and the first and second circles.

The boundary mesh has 100 vertices and 100 edges (see Figure 1). Note that the particular mesh generated could be sensitive to the ***machine precision*** and therefore may differ from one implementation to another. Figure 2 contains the output mesh.

9.1 Program Text

```
    Program d06aafe

!     D06AAF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: d06aaf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: coef, power
      Integer                          :: i, i1, ifail, itrace, k, liwork,     &
                                          lrwork, nedge, nelt, nv, nvb, nvmax, &
                                          reftk
      Logical                          :: smooth
      Character (1)                    :: pmesh
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: bspace(:), coor(:,:), rwork(:)
      Integer, Allocatable             :: conn(:,:), edge(:,:), iwork(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max
!     .. Executable Statements ..
      Write (nout,*) 'D06AAF Example Program Results'
```

```
!     Skip heading in data file
      Read (nin,*)

!     Reading of the geometry
!     Coordinates of the boundary mesh vertices and
!     edges references.

      Read (nin,*) nvb, nvmax, nedge
      lrwork = nvmax
      liwork = 16*nvmax + 2*nedge + max(4*nvmax+2,nedge-14)
      Allocate (bspace(nvb),coor(2,nvmax),rwork(lrwork),conn(3,2*(nvmax- &
        1)),edge(3,nedge),iwork(liwork))

      Do i = 1, nvb
        Read (nin,*) i1, coor(1,i1), coor(2,i1)
      End Do

!     Boundary edges

      Do i = 1, nedge
        Read (nin,*) i1, edge(1,i1), edge(2,i1), edge(3,i1)
      End Do

!     Initialise mesh control parameters

      bspace(1:nvb) = 0.05E0_nag_wp
      smooth = .True.
      itrace = 0
      coef = 0.75E0_nag_wp
      power = 0.25E0_nag_wp

!     Call to the mesh generator

      ifail = 0
      Call d06aaf(nvb,nvmax,nedge,edge,nv,nelt,coor,conn,bspace,smooth,coef, &
        power,itrace,rwork,lrwork,iwork,liwork,ifail)

      Write (nout,*)
      Read (nin,*) pmesh

      Select Case (pmesh)
      Case ('N')
        Write (nout,99999) 'NV   =', nv
        Write (nout,99999) 'NELT =', nelt
      Case ('Y')

!       Output the mesh

        Write (nout,99998) nv, nelt

        Do i = 1, nv
          Write (nout,99997) coor(1,i), coor(2,i)
        End Do

        reftk = 0

        Do k = 1, nelt
          Write (nout,99996) conn(1,k), conn(2,k), conn(3,k), reftk
        End Do

      Case Default
        Write (nout,*) 'Problem with the printing option Y or N'
      End Select

99999 Format (1X,A,I6)
99998 Format (1X,2I10)
99997 Format (2(2X,E13.6))
99996 Format (1X,4I10)
    End Program d06aafe
```

9.2 Program Data

Note 1: since the data file for this example is quite large only a section of it is reproduced in this document. The full data file is distributed with your implementation.

```
D06AAF Example Program Data
       100       100     :NVB NEDGE
     1  0.100000E+01  0.000000E+00
            .
            .
            .
   100  -.353278E+00  0.681187E+00 :(I1, COOR(:,I),I=1,...,NVB)
   1   1   2   1
            .
            .
            .
  99  99 100   1
 100 100  71   1  :(I1, EDGE(:,I), I=1,...,NEDGE)
'N'               :Printing option 'Y' or 'N'
```

9.3 Program Results

```
 D06AAF Example Program Results

 NV   =   250
 NELT =   402
```

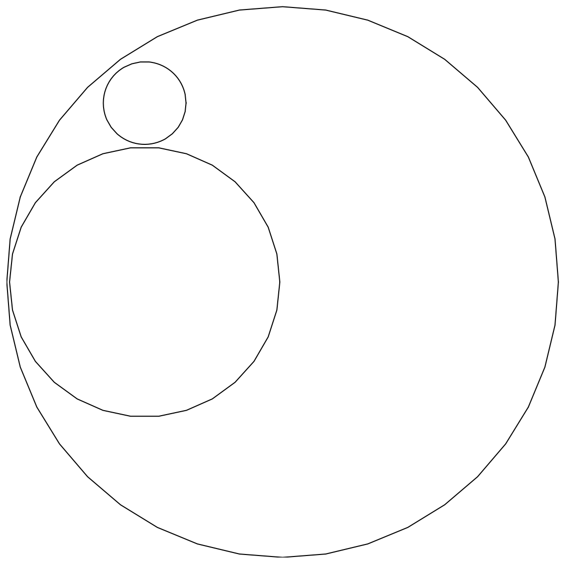

Figure 1
The boundary mesh of the geometry with two holes

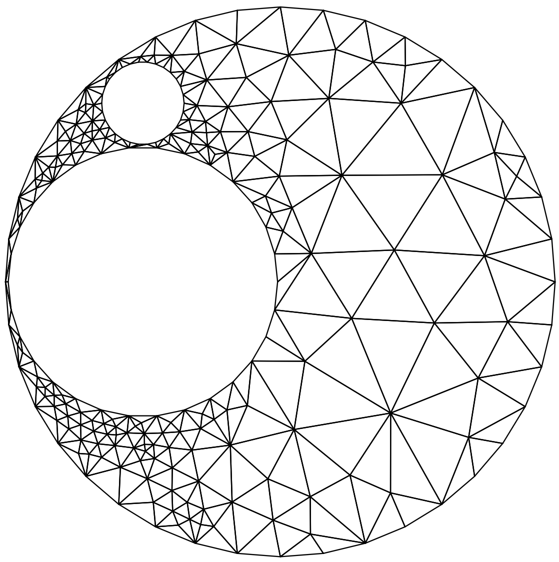

Figure 2
Interior mesh of the geometry with two holes