NAG Library Routine Document

D06ABF

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

D06ABF generates a triangular mesh of a closed polygonal region in $\mathbb{R}^2$, given a mesh of its boundary. It uses a Delaunay–Voronoi process, based on an incremental method.

2 Specification

```
SUBROUTINE D06ABF (NVB, NVINT, NVMAX, NEDGE, EDGE, NV, NELT, COOR, CONN,    &
                   WEIGHT, NPROPA, ITRACE, RWORK, LRWORK, IWORK, LIWORK,    &
                   IFAIL)
INTEGER            NVB, NVINT, NVMAX, NEDGE, EDGE(3,NEDGE), NV, NELT,       &
                   CONN(3,2*NVMAX+5), NPROPA, ITRACE, LRWORK,               &
                   IWORK(LIWORK), LIWORK, IFAIL
REAL (KIND=nag_wp) COOR(2,NVMAX), WEIGHT(*), RWORK(LRWORK)
```

3 Description

D06ABF generates the set of interior vertices using a Delaunay–Voronoi process, based on an incremental method. It allows you to specify a number of fixed interior mesh vertices together with weights which allow concentration of the mesh in their neighbourhood. 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 Arguments

1: NVB – INTEGER *Input*

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

Constraint: NVB ≥ 3.

2: NVINT – INTEGER *Input*

On entry: the number of fixed interior mesh vertices to which a weight will be applied.

Constraint: NVINT ≥ 0.

3: NVMAX – INTEGER *Input*

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

Constraint: NVMAX $\geq$ NVB + NVINT.

4: NEDGE – INTEGER *Input*

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

Constraint: NEDGE ≥ 1.

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

On entry: the specification of the boundary edges. EDGE$(1, j)$ and EDGE$(2, j)$ contain the vertex numbers of the two end points of the jth boundary edge. EDGE$(3, j)$ is a user-supplied tag for the jth boundary edge and is not used by D06ABF.

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 $j = 1, 2, \ldots, \text{NEDGE}$.

6: NV – INTEGER *Output*

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

7: NELT – INTEGER *Output*

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

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

On entry: COOR$(1, i)$ contains the x coordinate of the ith input boundary mesh vertex, for $i = 1, 2, \ldots, \text{NVB}$. COOR$(1, i)$ contains the x coordinate of the $(i - \text{NVB})$th fixed interior vertex, for $i = \text{NVB} + 1, \ldots, \text{NVB} + \text{NVINT}$. For boundary and interior vertices, COOR$(2, i)$ contains the corresponding y coordinate, for $i = 1, 2, \ldots, \text{NVB} + \text{NVINT}$.

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

9: CONN$(3, 2 \times \text{NVMAX} + 5)$ – INTEGER array *Output*

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

10: WEIGHT$(*)$ – REAL (KIND=nag_wp) array *Input*

Note: the dimension of the array WEIGHT must be at least max$(1, \text{NVINT})$.

On entry: the weight of fixed interior vertices. It is the diameter of triangles (length of the longer edge) created around each of the given interior vertices.

Constraint: if NVINT > 0, WEIGHT$(i) > 0.0$, for $i = 1, 2, \ldots, \text{NVINT}$.

11: NPROPA – INTEGER *Input*

On entry: the propagation type and coefficient, the argument NPROPA is used when the internal points are created. They are distributed in a geometric manner if NPROPA is positive and in an arithmetic manner if it is negative. For more details see Section 9.

Constraint: NPROPA $\neq 0$.

12: ITRACE – INTEGER *Input*

On entry: the level of trace information required from D06ABF.

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.

13: RWORK(LRWORK) – REAL (KIND=nag_wp) array *Workspace*

14: LRWORK – INTEGER *Input*

On entry: the dimension of the array RWORK as declared in the (sub)program from which D06ABF is called.

Constraint: LRWORK $\geq 12 \times$ NVMAX $+ 15$.

15: IWORK(LIWORK) – INTEGER array *Workspace*

16: LIWORK – INTEGER *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D06ABF is called.

Constraint: LIWORK $\geq 6 \times$ NEDGE $+ 32 \times$ NVMAX $+ 2 \times$ NVB $+ 78$.

17: 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, NVB < 3,
or NVINT < 0,
or NVB $+$ NVINT $>$ NVMAX,
or NEDGE < 1,
or EDGE$(i, j) < 1$ or EDGE$(i, j) >$ NVB, for some $i = 1, 2$ and $j = 1, 2, \ldots,$ NEDGE,
or EDGE$(1, j) =$ EDGE$(2, j)$, for some $j = 1, 2, \ldots,$ NEDGE,
or NPROPA $= 0$;
or if NVINT > 0, WEIGHT$(i) \leq 0.0$, for some $i = 1, 2, \ldots,$ NVINT;
or LRWORK $< 12 \times$ NVMAX $+ 15$,
or LIWORK $< 6 \times$ NEDGE $+ 32 \times$ NVMAX $+ 2 \times$ NVB $+ 78$.

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.

IFAIL $= 3$

An error has occurred during the generation of the boundary mesh. It appears that NVMAX is not large enough.

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

Not applicable.

8 Parallelism and Performance

D06ABF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The position of the internal vertices is a function position of the vertices on the given boundary. A fine mesh on the boundary results in a fine mesh in the interior. To dilute the influence of the data on the interior of the domain, the value of NPROPA can be changed. The propagation coefficient is calculated as: $\omega = 1 + \dfrac{a - 1.0}{20.0}$, where a is the absolute value of NPROPA. During the process vertices are generated on edges of the mesh $\mathcal{T}_i$ to obtain the mesh $\mathcal{T}_{i+1}$ in the general incremental method (consult the D06 Chapter Introduction or George and Borouchaki (1998)). This generation uses the coefficient ω, and it is geometric if NPROPA > 0, and arithmetic otherwise. But increasing the value of a may lead to failure of the process, due to precision, especially in geometries with holes. So you are advised to manipulate the argument NPROPA 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**.

10 Example

In this example, a geometry with two holes (two wings inside an exterior circle) is meshed using a Delaunay–Voronoi method. The exterior circle is centred at the point $(1.0, 0.0)$ with a radius 3. The main wing, using aerofoil RAE 2822 data, lies between the origin and the centre of the circle, while the secondary aerofoil is produced from the first by performing a translation, a scale reduction and a rotation. To be able to carry out some realistic computation on that geometry, some interior points have been introduced to have a finer mesh in the wake of those aerofoils.

The boundary mesh has 296 vertices and 296 edges (see Section 10.3 top). Note that the particular mesh generated could be sensitive to the ***machine precision*** and therefore may differ from one implementation to another. The interior meshes for different values of NPROPA are given in Section 10.3.

10.1 Program Text

```
    Program d06abfe

!     D06ABF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: d06abf, f06epf, nag_wp, x01aaf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: meshout = 7, nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: c, d, dnvint, r, s, theta, theta_i
      Integer                          :: i, i1, ic, ifail, itrace, j, liwork, &
                                          lrwork, nearest, nedge, nelt,        &
                                          nelt_near, npropa, nrae, nv, nvb,    &
                                          nvint, nvmax, nv_near
      Character (1)                    :: pmesh
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: coor(:,:), rwork(:), weight(:)
      Real (Kind=nag_wp)               :: t(2)
      Integer, Allocatable             :: conn(:,:), edge(:,:), iwork(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: cos, real, sin, tan
!     .. Executable Statements ..
      Write (nout,*) 'D06ABF Example Program Results'

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

!     Reading of the geometry:
!          number of points on RAE aerofoil data;
!          number of points on circular boundary;
!          maximum number of vertices.

      Read (nin,*) nrae, nvint, nvmax
      Read (nin,*) pmesh

      nvb = nvint + 2*nrae
      nedge = nvb

      lrwork = 12*nvmax + 15
      liwork = 6*nedge + 32*nvmax + 2*nvb + 78
      Allocate (coor(2,nvmax),rwork(lrwork),weight(nvint),conn(3,2*nvmax+5),   &
        edge(3,nedge),iwork(liwork))

!     Circular outer boundary, radius 3 and centre (1,0)
      theta = 2.0_nag_wp*x01aaf(r)/real(nvint,kind=nag_wp)
      r = 3.0_nag_wp
      t(1) = 1.0_nag_wp
      Do i = 1, nvint
```

```
        theta_i = theta*real(i-1,kind=nag_wp)
        coor(1,i) = r*cos(theta_i) + t(1)
        coor(2,i) = r*sin(theta_i)
      End Do

!     Read data for aerofoil RAE 2822
      Do i = 1, nrae
        Read (nin,*) i1, coor(1,nvint+i), coor(2,nvint+i)
      End Do

!     Transform RAE 2822 for secondary foil
      theta = x01aaf(theta)/12.0_nag_wp
      c = cos(theta)
      s = sin(theta)
      ic = nvint + nrae
!     Copy and rotate coordinates by theta = pi/12
      coor(1:2,ic+1:ic+nrae) = coor(1:2,nvint+1:ic)
      Call f06epf(nrae,coor(1,ic+1),2,coor(2,ic+1),2,c,s)
!     Reduce by 0.4 and translate to distance 0.25 from intercept at (0.75,0)
      d = 0.4_nag_wp
      t(1) = 0.75_nag_wp + 0.25_nag_wp*c
      t(2) = -0.25_nag_wp*s
      Do i = 1, nrae
        coor(1:2,ic+i) = d*coor(1:2,ic+i) + t(1:2)
      End Do

!     Boundary edges
      Do i = 1, nedge
        edge(1,i) = i
        edge(2,i) = i + 1
        edge(3,i) = 0
      End Do
!     Tie up end of three boundary edges
      edge(2,nvint) = 1
      edge(2,nvint+nrae) = nvint + 1
      edge(2,nedge) = nvint + nrae + 1

!     Initialize mesh control parameters

      itrace = 0

!     Generation of interior vertices on the
!     RAE airfoil's wake

      dnvint = 2.5E0_nag_wp/real(nvint+1,kind=nag_wp)

      Do i = 1, nvint
        i1 = nvb + i
        coor(1,i1) = 1.38E0_nag_wp + real(i,kind=nag_wp)*dnvint
        coor(2,i1) = -tan(theta)*(coor(1,i1)-0.75_nag_wp)
      End Do

      weight(1:nvint) = 0.01E0_nag_wp

      Write (nout,*)

!      Loop on the propagation coef

pcoef: Do j = 1, 4

        nearest = 250
        Select Case (j)
        Case (1)
          npropa = -5
        Case (2)
          npropa = -1
        Case (3)
          npropa = 1
        Case Default
          npropa = 5
        End Select
```

```
!       Call to the 2D Delaunay-Voronoi mesh generator

        ifail = 0
        Call d06abf(nvb,nvint,nvmax,nedge,edge,nv,nelt,coor,conn,weight,       &
          npropa,itrace,rwork,lrwork,iwork,liwork,ifail)

        Write (nout,99999) 'Mesh characteristics with NPROPA =', npropa
        nv_near = ((nv+nearest/2)/nearest)*nearest
        nelt_near = ((nelt+nearest/2)/nearest)*nearest
        Write (nout,99998) 'NV  ', nv_near, nearest
        Write (nout,99998) 'NELT', nelt_near, nearest

        If (pmesh=='Y') Then
!         Output the mesh in a form suitable for printing

          If (j==1) Then
            Write (meshout,*) '# D06ABF Example Program Mesh results'
          End If
          Write (meshout,99999) '# Mesh line segments for NPROPA =', npropa
          Do i = 1, nelt
            Write (meshout,99997) coor(1,conn(1,i)), coor(2,conn(1,i))
            Write (meshout,99997) coor(1,conn(2,i)), coor(2,conn(2,i))
            Write (meshout,99997) coor(1,conn(3,i)), coor(2,conn(3,i))
            Write (meshout,99997) coor(1,conn(1,i)), coor(2,conn(1,i))
            Write (meshout,*)
          End Do
          Write (meshout,*)
        End If

      End Do pcoef

99999 Format (1X,A,I6)
99998 Format (1X,A5,' = ',I10,' to the nearest ',I3)
99997 Format (2(2X,E13.6))
    End Program d06abfe
```

10.2 Program Data

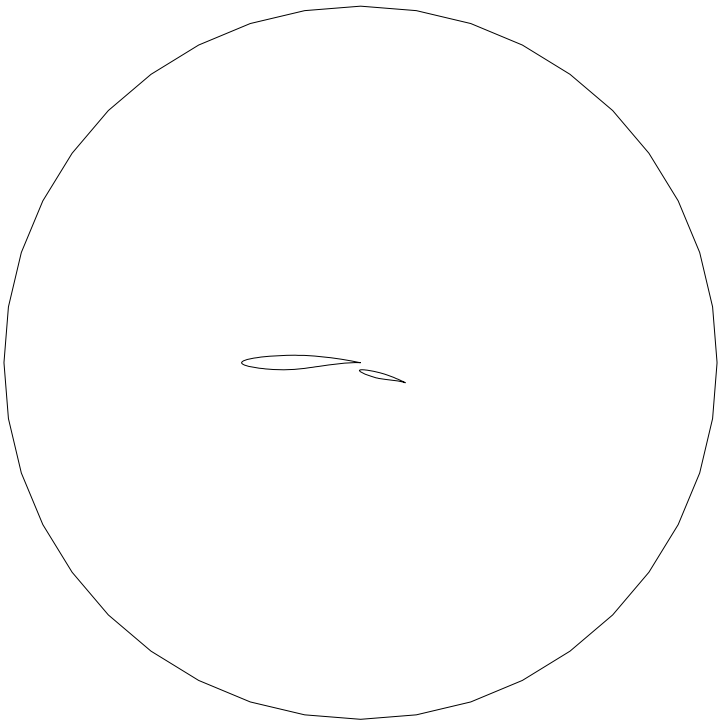
Note 1: since the data file for this example is quite large only a section of it is reproduced in this document. The full data file is distributed with your implementation.

```
D06ABF Example Program Data
       128       40       6000        : nrae  nvint  nvmax
       'N'                            : Printing mesh? 'Y' or 'N'

  01     0.000000E+00  0.000000E+00
            .
            .
            .
 128     0.602000E-03  -.316000E-02   : RAE Aerofoil 2822
```

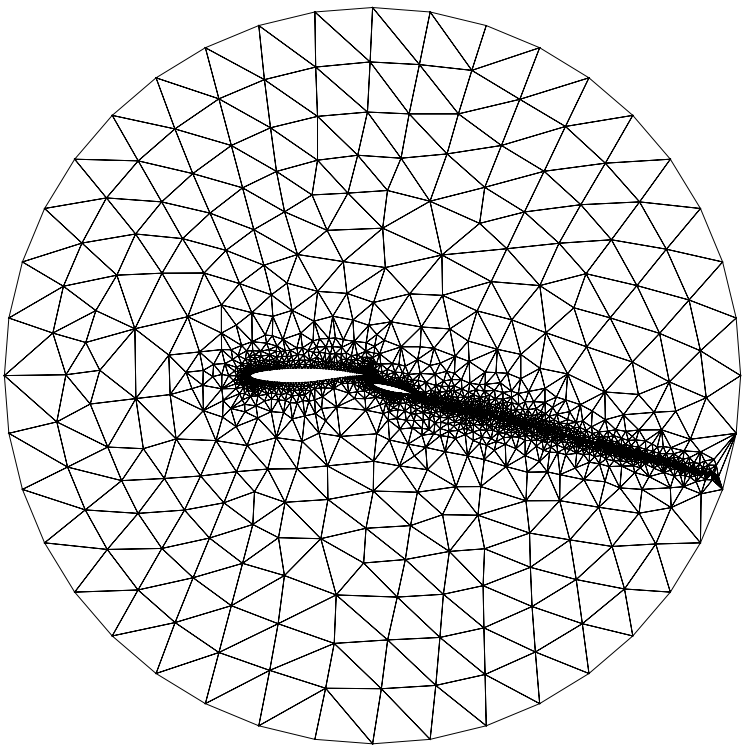
10.3 Program Results

```
 D06ABF Example Program Results

 Mesh characteristics with NPROPA =    -5
  NV   =       2250 to the nearest 250
  NELT =       4250 to the nearest 250
 Mesh characteristics with NPROPA =    -1
  NV   =       4500 to the nearest 250
  NELT =       8500 to the nearest 250
 Mesh characteristics with NPROPA =     1
  NV   =       5250 to the nearest 250
  NELT =      10000 to the nearest 250
 Mesh characteristics with NPROPA =     5
  NV   =       2000 to the nearest 250
  NELT =       3750 to the nearest 250
```

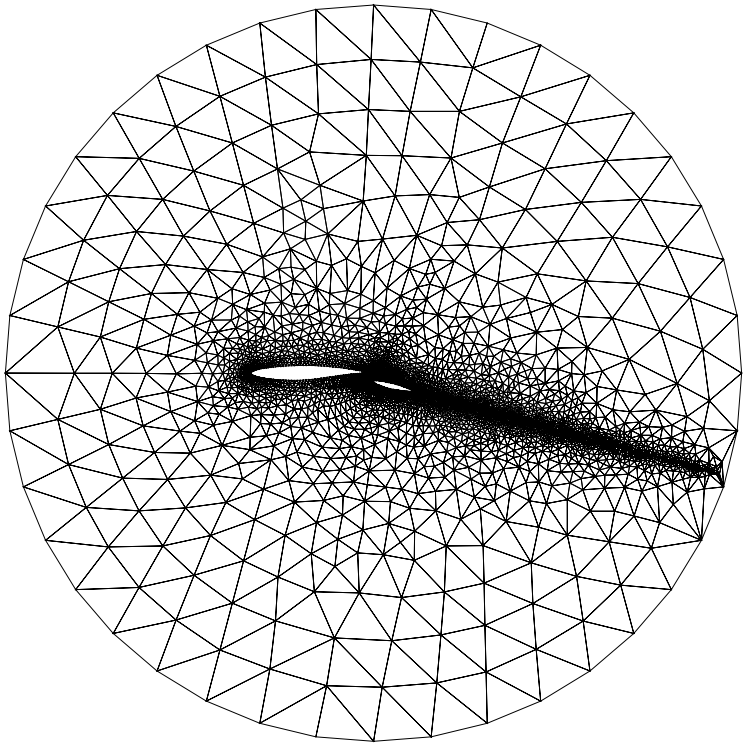
Example Program
Geometry for Generating Meshes



Mesh Generated Using Arithmetic Coefficient ω=1.2



Mesh Generated Using Arithmetic Coefficient ω=1.0



Mesh Generated Using Geometric Coefficient ω=1.0