NAG Fortran Library Routine Document

D06ACF

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

D06ACF generates a triangular mesh of a closed polygonal region in $\mathbb{R}^2$, given a mesh of its boundary. It uses an Advancing Front process, based on an incremental method.

2 Specification

```
 SUBROUTINE D06ACF(NVB, NVINT, NVMAX, NEDGE, EDGE, NV, NELT, COOR, CONN,
1                  WEIGHT, ITRACE, RWORK, LRWORK, IWORK, LIWORK, IFAIL)
 INTEGER           NVB, NVINT, NVMAX, NEDGE, EDGE(3,NEDGE), NV, NELT,
1                  CONN(3,2*NVMAX+5), ITRACE, LRWORK, IWORK(LIWORK),
2                  LIWORK, IFAIL
 real              COOR(2,NVMAX), WEIGHT(*), RWORK(LRWORK)
```

3 Description

D06ACF generates the set of interior vertices using an Advancing Front process, based on an incremental method. It allows the user 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 Parameters

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 the 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. $\text{EDGE}(1:2, j)$ contains the vertex number 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 this routine.

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

6: NV – INTEGER *Output*

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

7: NELT – INTEGER *Output*

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

8: COOR(2,NVMAX) – ***real*** array *Input/Output*

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

On exit: the input elements $\text{COOR}(1:2, 1:(\text{NVB} + \text{NVINT}))$ are unchanged. $\text{COOR}(1, i)$ 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 $\text{COOR}(2, i)$ will contain the corresponding y-coordinate.

9: CONN(3,2*NVMAX+5) – 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 in COOR of its three vertices (in anticlockwise order), for $i = 1, 2, 3$ and $j = 1, \ldots, \text{NELT}$.

10: WEIGHT(*) – ***real*** 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 $\text{NVINT} > 0$, $\text{WEIGHT}(i) > 0.0$, for $i = 1, \ldots, \text{NVINT}$.

11: ITRACE – INTEGER *Input*

On entry: the level of trace information required from D06ACF as follows:

if $\text{ITRACE} \leq 0$, no output is generated;

if $\text{ITRACE} \geq 1$, then 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.

Users are advised to set $\text{ITRACE} = 0$, unless they are experienced with the Finite Element meshes.

12: RWORK(LRWORK) – ***real*** array *Workspace*

13: LRWORK – INTEGER *Input*

On entry: the dimension of the array RWORK as declared in the (sub)program from which D06ACF is called.

Constraint: $\text{LRWORK} \geq 12 \times \text{NVMAX} + 30015$.

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

15: LIWORK – INTEGER *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D06ACF is called.

Constraint: LIWORK $\geq 8 \times$ NEDGE $+ 53 \times$ NVMAX $+ 2 \times$ NVB $+ 10078$.

16: 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 $= 1$

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 $j = 1, \ldots,$ NEDGE,
or EDGE$(1,j) =$ EDGE$(2,j)$, for some $j = 1, \ldots,$ NEDGE,
or if NVINT > 0, WEIGHT$(i) \leq 0.0$, for some $i = 1, \ldots,$ NVINT,
or LRWORK $< 12 \times$ NVMAX $+ 30015$,
or LIWORK $< 8 \times$ NEDGE $+ 53 \times$ NVMAX $+ 2 \times$ NVB $+ 10078$.

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 vertices positions on the given boundary. A fine mesh at the level of the boundary introduces a fine mesh of the interior. 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)).

To ensure correct functioning of the routine, the user is advised 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 wings inside an exterior circle) is meshed using a Delaunay-Voronoi method. The exterior circle is centred at the point $(1.5, 0.0)$ with a radius 4.5, the first wing begins at the origin and it is normalised, finally the last wing is also normalised and begins at the point $(0.8, -0.3)$. To be able to carry out some realistic computation on that kind of geometry, some interior points have been introduced to have a finer mesh in the wake of those airfoils.

The boundary mesh has 120 vertices and 120 edges (see Figure 1 top). Note that the particular mesh generated could be sensitive to the machine precision and therefore may differ from one implementation to another. Contains the generated mesh Figure 1.

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.

```
*     D06ACF Example Program Text
*     Mark 20 Release. NAG Copyright 2001.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NEDMX, NVMAX, NLINESX, NUS, NCOMPX, MAXCAN,
     +                 NVIMX, LRWORK, LIWORK
      PARAMETER        (NEDMX=200,NVMAX=2000,NLINESX=50,NUS=100,
     +                 NCOMPX=5,MAXCAN=10000,NVIMX=40,
     +                 LRWORK=12*NVMAX+3*MAXCAN+15,
     +                 LIWORK=8*NEDMX+55*NVMAX+MAXCAN+78)
*     .. Local Scalars ..
      real             DNVINT, RADIUS, X0, X1, Y0, Y1
      INTEGER          I, IFAIL, ITRACE, J, K, NCOMP, NEDGE, NELT,
     +                 NLINES, NV, NVB, NVINT, NVINT2, REFTK
      CHARACTER        PMESH
*     .. Local Arrays ..
      real             COOR(2,NVMAX), COORCH(2,NLINESX), COORUS(2,NUS),
     +                 RATE(NLINESX), RUSER(5), RWORK(LRWORK),
     +                 WEIGHT(NVIMX)
      INTEGER          CONN(3,2*NVMAX+5), EDGE(3,NEDMX), IUSER(1),
     +                 IWORK(LIWORK), LCOMP(NLINESX), LINE(4,NLINESX),
     +                 NLCOMP(NCOMPX)
*     .. External Functions ..
      real             FBND
      EXTERNAL         FBND
*     .. External Subroutines ..
      EXTERNAL         D06ACF, D06BAF
*     .. Intrinsic Functions ..
      INTRINSIC        ABS, real
*     .. Executable Statements ..
*
      WRITE (NOUT,*) 'D06ACF Example Program Results'
      WRITE (NOUT,*)
*
*     Skip heading in data file
*
      READ (NIN,*)
*
*     Initialise boundary mesh inputs:
*     the number of line and of the characteristic points of
*     the boundary mesh
*
      READ (NIN,*) NLINES
*
      READ (NIN,*) (COORCH(1,J),J=1,NLINES)
*
      READ (NIN,*) (COORCH(2,J),J=1,NLINES)
*
*     The Lines of the boundary mesh
*
```

```
      READ (NIN,*) ((LINE(I,J),I=1,4),RATE(J),J=1,NLINES)
*
*     The number of connected components to the boundary
*     and their informations
*
      READ (NIN,*) NCOMP
      J = 1
      DO 20 I = 1, NCOMP
         READ (NIN,*) NLCOMP(I)
*
         READ (NIN,*) (LCOMP(K),K=J,J+ABS(NLCOMP(I))-1)
         J = J + ABS(NLCOMP(I))
   20 CONTINUE
*
      READ (NIN,*) PMESH
*
*     Data passed to the user-supplied function
*
      X0 = 1.5e0
      Y0 = 0.e0
      RADIUS = 4.5e0
      X1 = 0.8e0
      Y1 = -0.3e0
*
      RUSER(1) = X0
      RUSER(2) = Y0
      RUSER(3) = RADIUS
      RUSER(4) = X1
      RUSER(5) = Y1
      IUSER(1) = 0
*
      ITRACE = 0
*
*     Call to the 2D boundary mesh generator
*
      IFAIL = 0
*
      CALL D06BAF(NLINES,COORCH,LINE,FBND,COORUS,NUS,RATE,NCOMP,NLCOMP,
     +            LCOMP,NVMAX,NEDMX,NVB,COOR,NEDGE,EDGE,ITRACE,RUSER,
     +            IUSER,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
      IF (PMESH.EQ.'N') THEN
         WRITE (NOUT,*) 'Boundary mesh characteristics'
         WRITE (NOUT,99999) 'NVB   =', NVB
         WRITE (NOUT,99999) 'NEDGE =', NEDGE
      ELSE IF (PMESH.EQ.'Y') THEN
*
*     Output the mesh to view it using the NAG Graphics Library
*
         WRITE (NOUT,99998) NVB, NEDGE
*
         DO 40 I = 1, NVB
            WRITE (NOUT,99997) I, COOR(1,I), COOR(2,I)
   40    CONTINUE
*
         DO 60 I = 1, NEDGE
            WRITE (NOUT,99996) I, EDGE(1,I), EDGE(2,I), EDGE(3,I)
   60    CONTINUE
      ELSE
         WRITE (NOUT,*) 'Problem with the printing option Y or N'
         STOP
      END IF
*
*     Initialise mesh control parameters
*
      ITRACE = 0
*
*     Generation of interior vertices
*     for the wake of the first NACA
*
      NVINT = 40
```

```
      NVINT2 = 20
      DNVINT = 5.e0/real(NVINT2+1)
      DO 80 I = 1, NVINT2
         REFTK = NVB + I
         COOR(1,REFTK) = 1.e0 + real(I)*DNVINT
         COOR(2,REFTK) = 0.e0
         WEIGHT(I) = 0.05e0
   80 CONTINUE
*
*     for the wake of the second one
*
      DNVINT = 4.19e0/real(NVINT2+1)
      DO 100 I = NVINT2 + 1, NVINT
         REFTK = NVB + I
         COOR(1,REFTK) = 1.8e0 + real(I-NVINT2)*DNVINT
         COOR(2,REFTK) = -0.3e0
         WEIGHT(I) = 0.05e0
  100 CONTINUE
*
*     Call to the 2D Advancing front mesh generator
*
      IFAIL = 0
*
      CALL D06ACF(NVB,NVINT,NVMAX,NEDGE,EDGE,NV,NELT,COOR,CONN,WEIGHT,
     +            ITRACE,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
      IF (PMESH.EQ.'N') THEN
         WRITE (NOUT,*) 'Complete mesh characteristics'
         WRITE (NOUT,99999) 'NV   =', NV
         WRITE (NOUT,99999) 'NELT =', NELT
      ELSE IF (PMESH.EQ.'Y') THEN
*
*     Output the mesh to view it using the NAG Graphics Library
*
         WRITE (NOUT,99998) NV, NELT
         DO 120 I = 1, NV
            WRITE (NOUT,99995) COOR(1,I), COOR(2,I)
  120    CONTINUE
*
         REFTK = 0
         DO 140 K = 1, NELT
            WRITE (NOUT,99994) CONN(1,K), CONN(2,K), CONN(3,K), REFTK
  140    CONTINUE
      END IF
*
      STOP
*
99999 FORMAT (1X,A,I6)
99998 FORMAT (1X,2I10)
99997 FORMAT (2X,I4,2(2X,E12.6))
99996 FORMAT (1X,4I4)
99995 FORMAT (2(2X,E12.6))
99994 FORMAT (1X,4I10)
      END
*
      real FUNCTION FBND(I,X,Y,RUSER,IUSER)
*     .. Scalar Arguments ..
      real              X, Y
      INTEGER           I
*     .. Array Arguments ..
      real              RUSER(*)
      INTEGER           IUSER(*)
*     .. Local Scalars ..
      real              C, RADIUS, X0, X1, Y0, Y1
*     .. Intrinsic Functions ..
      INTRINSIC         SQRT
*     .. Executable Statements ..
      FBND = 0.e0
      IF (I.EQ.1) THEN
*
*     upper NACA0012 wing beginning at the origin
```

```
*
          C = 1.008930411365e0
          FBND = 0.6e0*(0.2969e0*SQRT(C*X)-0.126e0*C*X-0.3516e0*(C*X)
     +            **2+0.2843e0*(C*X)**3-0.1015e0*(C*X)**4) - C*Y
       ELSE IF (I.EQ.2) THEN
*
*      lower NACA0012 wing beginning at the origin
*
          C = 1.008930411365e0
          FBND = 0.6e0*(0.2969e0*SQRT(C*X)-0.126e0*C*X-0.3516e0*(C*X)
     +            **2+0.2843e0*(C*X)**3-0.1015e0*(C*X)**4) + C*Y
       ELSE IF (I.EQ.3) THEN
          X0 = RUSER(1)
          Y0 = RUSER(2)
          RADIUS = RUSER(3)
          FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS**2
       ELSE IF (I.EQ.4) THEN
*
*      upper NACA0012 wing beginning at (X1;Y1)
*
          C = 1.008930411365e0
          X1 = RUSER(4)
          Y1 = RUSER(5)
          FBND = 0.6e0*(0.2969e0*SQRT(C*(X-X1))-0.126e0*C*(X-X1)
     +            -0.3516e0*(C*(X-X1))**2+0.2843e0*(C*(X-X1))
     +            **3-0.1015e0*(C*(X-X1))**4) - C*(Y-Y1)
       ELSE IF (I.EQ.5) THEN
*
*      lower NACA0012 wing beginning at (X1;Y1)
*
          C = 1.008930411365e0
          X1 = RUSER(4)
          Y1 = RUSER(5)
          FBND = 0.6e0*(0.2969e0*SQRT(C*(X-X1))-0.126e0*C*(X-X1)
     +            -0.3516e0*(C*(X-X1))**2+0.2843e0*(C*(X-X1))
     +            **3-0.1015e0*(C*(X-X1))**4) + C*(Y-Y1)
       END IF
*
       RETURN
       END
```

9.2 Program Data

```
D06ACF Example Program Data
 8                                                         :NLINES (m)
  0.0000   1.0000  -3.0000   6.0000   0.8000
  1.8000   1.5000   1.5000                            :(COORCH(1,1:m))
  0.0000   0.0000   0.0000   0.0000  -0.3000
 -0.3000   4.5000  -4.5000                            :(COORCH(2,1:m))
21  2  1  1   1.0000 21  1  2  2   1.0000
11  3  8  3   1.0000 11  4  7  3   1.0000
21  6  5  4   1.0000 21  5  6  5   1.0000
11  7  3  3   1.0000 11  8  4  3   1.0000 :(LINE(:,j),RATE(j),j=1,m)
 3                                      :NCOMP (n, number of contours)
-2                                       :number of lines in contour 1
 1  2                                              :lines of contour 1
 4                                       :number of lines in contour 2
 3  8  4  7                                        :lines of contour 2
-2                                       :number of lines in contour 3
 5  6                                              :lines of contour 3
'N'                                        :Printing option 'Y' or 'N'
```

9.3 Program Results

```
D06ACF Example Program Results

Boundary mesh characteristics
NVB   =   120
NEDGE =   120
Complete mesh characteristics
NV   =  1892
NELT =  3666
```

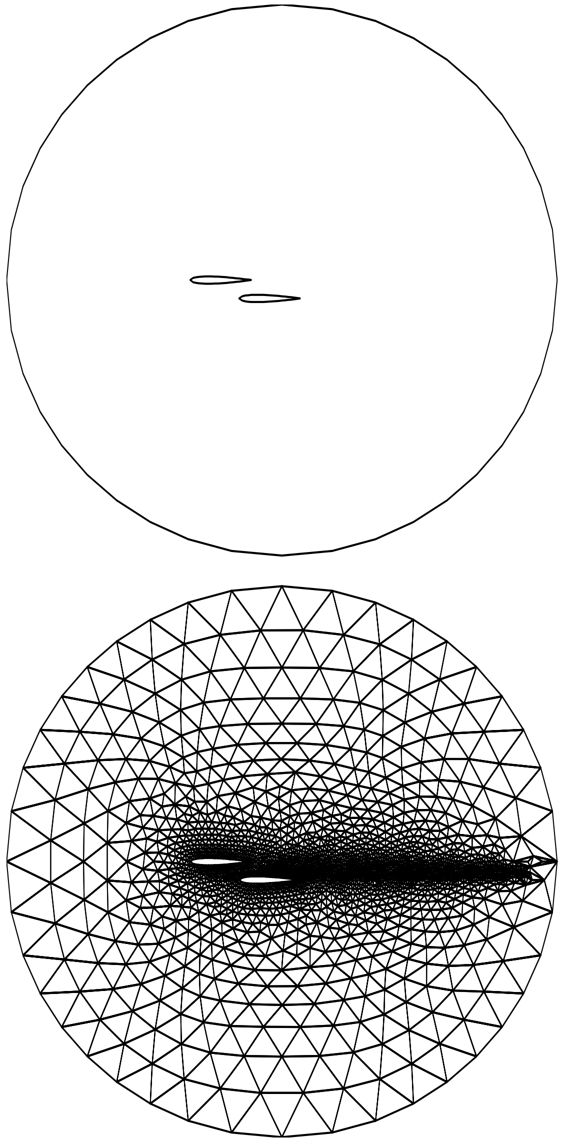

Figure 1
The boundary mesh (top), the interior mesh (bottom) of a double wing inside a circle geometry