NAG Fortran Library Routine Document

D06BAF

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

D06BAF generates a boundary mesh on a closed connected subdomain Ω of $\mathbb{R}^2$.

2 Specification

```
 SUBROUTINE D06BAF (NLINES, COORCH, LINED, FBND, COORUS, NUS, RATE,
1                   NCOMP, NLCOMP, LCOMP, NVMAX, NEDMX, NVB, COOR, NEDGE,
2                   EDGE, ITRACE, RUSER, IUSER, RWORK, LRWORK, IWORK,
3                   LIWORK, IFAIL)
 INTEGER            NLINES, LINED(4,NLINES), NUS, NCOMP, NLCOMP(NCOMP),
1                   LCOMP(NLINES), NVMAX, NEDMX, NVB, NEDGE,
2                   EDGE(3,NEDMX), ITRACE, IUSER(*), LRWORK,
3                   IWORK(LIWORK), LIWORK, IFAIL
 double precision   COORCH(2,NLINES), COORUS(2,NUS), RATE(NLINES),
1                   COOR(2,NVMAX), RUSER(*), RWORK(LRWORK)
 EXTERNAL           FBND
```

3 Description

Given a closed connected subdomain Ω of $\mathbb{R}^2$, whose boundary $\partial\Omega$ is divided by characteristic points into m distinct line segments, D06BAF generates a boundary mesh on $\partial\Omega$. Each line segment may be a straight line, a curve defined by the equation $f(x,y) = 0$, or a polygonal curve defined by a set of given boundary mesh points.

This routine is primarily designed for use with either D06AAF (a simple incremental method) or D06ABF (Delaunay–Voronoi method) or D06ACF (Advancing Front method) to triangulate the interior of the domain Ω. For more details about the boundary and interior mesh generation, 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: NLINES – INTEGER *Input*

On entry: m, the number of lines that define the boundary of the closed connected subdomain (this equals the number of characteristic points which separate the entire boundary $\partial\Omega$ into lines).

Constraint: NLINES ≥ 1.

2: COORCH(2,NLINES) – ***double precision*** array *Input*

On entry: COORCH$(1,i)$ contains the x co-ordinate of the ith characteristic point, for $i = 1,\ldots,$NLINES; while COORCH$(2,i)$ contains the corresponding y co-ordinate.

3: LINED(4,NLINES) – INTEGER array *Input*

On entry: the description of the lines that define the boundary domain. The line i, for $i = 1, \ldots, m$, is defined as follows:

LINED$(1, i)$

The number of points on the line, including two end points.

LINED$(2, i)$

The first end point of the line. If LINED$(2, i) = j$, then the co-ordinates of the first end point are those stored in COORCH$(:, j)$.

LINED$(3, i)$

The second end point of the line. If LINED$(3, i) = k$, then the co-ordinates of the second end point are those stored in COORCH$(:, k)$.

LINED$(4, i)$

This defines the type of line segment connecting the end points. Additional information is conveyed by the numerical value of LINED$(4, i)$ as follows:

(i) LINED$(4, i) > 0$, the line is described in the user-supplied function FBND with LINED$(4, i)$ as the index. In this case, the line must be described in the trigonometric (anticlockwise) direction;

(ii) LINED$(4, i) = 0$, the line is a straight line;

(iii) if LINED$(4, i) < 0$, say $(-p)$, then the line is a polygonal arc joining the end points and interior points specified in COORUS. In this case the line contains the points whose co-ordinates are stored in COORCH$(:, j)$,COORUS$(:, p)$, COORUS$(:, p + 1)$,. . .,COORUS$(:, p + r - 3)$, COORCH$(:, k)$, where $r =$ LINED$(1, i)$, $j =$ LINED$(2, i)$ and $k =$ LINED$(3, i)$.

Constraints:

$2 \leq$ LINED$(1, i)$;
$1 \leq$ LINED$(2, i) \leq$ NLINES;
$1 \leq$ LINED$(3, i) \leq$ NLINES;
LINED$(2, i) \neq$ LINED$(3, i)$, for $i = 1, 2, \ldots,$ NLINES.

For each line described by the user-supplied function (lines with LINED$(4, i) > 0$, $i = 1, \ldots,$ NLINES) the two end points (LINED$(2, i)$ and LINED$(3, i)$) lie on the curve defined by index LINED$(4, i)$ in the user-supplied function FBND, i.e.,

$\mathrm{FBND}(\mathrm{LINED}(4, i), \mathrm{COORCH}(1, \mathrm{LINED}2i)), \mathrm{COORCH}(2, \mathrm{LINED}2i)), \mathrm{RUSER}, \mathrm{IUSER}) = 0;$

$\mathrm{FBND}(\mathrm{LINED}(4, i), \mathrm{COORCH}(1, \mathrm{LINED}3i)), \mathrm{COORCH}(2, \mathrm{LINED}3i)), \mathrm{RUSER}, \mathrm{IUSER}) = 0,$ for $i = 1, 2, \ldots,$ NLINES.

For all lines described as polygonal arcs (lines with LINED$(4, i) < 0$, $i = 1, \ldots,$ NLINES) the sets of intermediate points (i.e., $[-\mathrm{LINED}(4, i) : -\mathrm{LINED}(4, i) + \mathrm{LINED}(1, i) - 3]$ for all i such that LINED$(4, i) < 0$) are not overlapping. This can be expressed as:

$$-\mathrm{LINED}(4, i) + \mathrm{LINED}(1, i) - 3 = \sum_{\{i, \mathrm{LINED}(4,i)<0\}} \{\mathrm{LINED}(1, i) - 2\}$$

or

$$-\mathrm{LINED}(4, i) + \mathrm{LINED}(1, i) - 2 = -\mathrm{LINED}(4, j),$$

for a j such that $j = 1, \ldots,$ NLINES, $j \neq i$ and LINED$(4, j) < 0$.

4: FBND – ***double precision*** FUNCTION, supplied by the user *External Procedure*

FBND must be supplied by you to calculate the value of the function which describes the curve $\{(x, y) \in \mathbb{R}^2; \text{ such that } f(x, y) = 0\}$ on segments of the boundary for which LINED$(4, i) > 0$. If there are no boundaries for which LINED$(4,i) > 0$ FBND will never be referenced by D06BAF and FBND may be the dummy function D06BAD. (D06BAD is included in the NAG Fortran Library and so need not be supplied by you. Its name may be implementation-dependent: see the Users' Note for your implementation for details.)

Its specification is:

```
double precision FUNCTION FBND (I, X, Y, RUSER, IUSER)

INTEGER                          I, IUSER(*)
double precision                 X, Y, RUSER(*)
```

1: I – INTEGER *Input*

On entry: LINED$(4,i)$, the reference index of the line (portion of the contour) i described.

2: X – ***double precision*** *Input*
3: Y – ***double precision*** *Input*

On entry: the values of x and y at which $f(x,y)$ is to be evaluated.

4: RUSER$(*)$ – ***double precision*** array *User Workspace*
5: IUSER$(*)$ – INTEGER array *User Workspace*

You are free to use these arrays to supply information to this function from the calling (sub)program.

FBND must be declared as EXTERNAL in the (sub)program from which D06BAF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

5: COORUS(2,NUS) – ***double precision*** array *Input*

On entry: the co-ordinates of the intermediate points for polygonal arc lines. For a line i defined as a polygonal arc (i.e., LINED$(4,i) < 0$), if $p = -$LINED$(4,i)$, then COORUS$(1,k)$, $k = p, p+1, \ldots, p+$LINED$(1,i) - 3$ must contain the x co-ordinate of the consecutive intermediate points for this line. Similarly COORUS$(2,k)$, $k = p, p+1, \ldots, p+$LINED$(1,i) - 3$ must contain the corresponding y co-ordinate.

6: NUS – INTEGER *Input*

On entry: the second dimension of the array COORUS as declared in the (sub)program from which D06BAF is called.

Constraint: $\text{NUS} \geq \sum_{\{i,\text{LINED}(4,i)<0\}} \{\text{LINED}(1,i) - 2\}$.

7: RATE(NLINES) – ***double precision*** array *Input*

On entry: RATE(i) is the geometric progression ratio between the points to be generated on the line i, for $i = 1, \ldots, m$ and LINED$(4,i) \geq 0$.

If LINED$(4,i) < 0$, RATE(i) is not referenced.

Constraint: if LINED$(4,i) \geq 0$, RATE$(i) > 0$, for $i = 1, 2, \ldots,$ NLINES.

8: NCOMP – INTEGER *Input*

On entry: n, the number of separately connected components of the boundary.

Constraint: NCOMP ≥ 1.

9: NLCOMP(NCOMP) – INTEGER array *Input*

On entry: $|\text{NLCOMP}(k)|$ is the number of line segments in component k of the contour. The line i of component k runs in the direction $\text{LINED}(2,i)$ to $\text{LINED}(3,i)$ if $\text{NLCOMP}(k) > 0$, and in the opposite direction otherwise; for $k = 1,\ldots,n$.

Constraints:

$$1 \leq |\text{NLCOMP}(k)| \leq \text{NLINES}, \text{ for } k = 1,2,\ldots,\text{NCOMP};$$

$$\sum_{k=1}^{n} |\text{NLCOMP}(k)| = \text{NLINES}.$$

10: LCOMP(NLINES) – INTEGER array *Input*

On entry: $\text{LCOMP}(l1:,l2)$, where $l2 = \sum_{i=1}^{k} |\text{NLCOMP}(i)|$ and $l1 = l2 + 1 - |\text{NLCOMP}(k)|$ is the list of line numbers for the kth components of the boundary, for $k = 1,\ldots,\text{NCOMP}$.

Constraint: LCOMP must hold a valid permutation of the integers $[1,\text{NLINES}]$.

11: NVMAX – INTEGER *Input*

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

Constraint: $\text{NVMAX} \geq \text{NLINES}$.

12: NEDMX – INTEGER *Input*

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

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

13: NVB – INTEGER *Output*

On exit: the total number of boundary mesh vertices generated.

14: COOR(2,NVMAX) – ***double precision*** array *Output*

On exit: $\text{COOR}(1,i)$ will contain the x co-ordinate of the ith boundary mesh vertex generated, for $i = 1,\ldots,\text{NVB}$; while $\text{COOR}(2,i)$ will contain the corresponding y co-ordinate.

15: NEDGE – INTEGER *Output*

On exit: the total number of boundary edges in the boundary mesh.

16: EDGE(3,NEDMX) – INTEGER array *Output*

On exit: the specification of the boundary edges. $\text{EDGE}(1,j)$ and $\text{EDGE}(2,j)$ will contain the vertex numbers of the two end points of the jth boundary edge. $\text{EDGE}(3,j)$ is a reference number for the jth boundary edge and

$\text{EDGE}(3,j) = \text{LINED}(4,i)$, where i and j are such that the jth edges is part of the ith line of the boundary and $\text{LINED}(4,i) \geq 0$;

$\text{EDGE}(3,j) = 100 + |\text{LINED}(4,i)|$, where i and j are such that the jth edges is part of the ith line of the boundary and $\text{LINED}(4,i) < 0$.

17: ITRACE – INTEGER *Input*

On entry: the level of trace information required from D06BAF.

$\text{ITRACE} \leq 0$

No output is generated.

ITRACE $= 1$

Output from the boundary mesh generator is printed on the current advisory message unit (see X04ABF). This output contains the input information of each line and each connected component of the boundary.

ITRACE > 1

The output is similar to that produced when ITRACE $= 1$, but the co-ordinates of the generated vertices on the boundary are also output.

ITRACE $= -1$

An analysis of the output boundary mesh is printed on the current advisory message unit. This analysis includes the orientation (clockwise or anticlockwise) of each connected component of the boundary. This information could be of interest to you, especially if an interior meshing is carried out using the output of this routine, calling either D06AAF, D06ABF or D06ACF.

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

18: RUSER($*$) – ***double precision*** array *User Workspace*
19: IUSER($*$) – INTEGER array *User Workspace*

Note: the dimension of the array RUSER and IUSER must be at least 1.

You are free to use these arrays to supply information to this function from the calling (sub)program.

20: RWORK(LRWORK) – ***double precision*** array *Workspace*
21: LRWORK – INTEGER *Input*

On entry: the dimension of the array RWORK as declared in the (sub)program from which D06BAF is called.

Constraint: LRWORK $\geq 2 \times (\text{NLINES} + \text{NUS}) + 2 \times \max_{i=1,\ldots,m}\{\text{LINED}(1,i)\} \times \text{NLINES}$.

22: IWORK(LIWORK) – INTEGER array *Workspace*
23: LIWORK – INTEGER *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D06BAF is called.

Constraint: LIWORK $\geq \sum_{\{i,\text{LINED}(4,i)<0\}} \{\text{LINED}(1,i) - 2\} + 8 \times \text{NLINES} + \text{NVMAX} + 3 \times \text{NEDMX} + 2 \times \text{NUS}$.

24: 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 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, 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.**

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, NLINES < 1;
or NVMAX $<$ NLINES;
or NEDMX < 1;
or NCOMP < 1;
or LRWORK $< 2 \times (\text{NLINES} + \text{NUS}) + 2 \times \max_{i=1,\dots,m}\{\text{LINED}(1,i)\} \times \text{NLINES}$;
or LIWORK $< \sum_{\{i,\text{LINED}(4,i)<0\}} \{\text{LINED}(1,i) - 2\} + 8 \times \text{NLINES} + \text{NVMAX} + 3 \times \text{NEDMX} + 2 \times \text{NUS}$;
or NUS $< \sum_{\{i,\text{LINED}(4,i)<0\}} \{\text{LINED}(1,i) - 2\}$;
or RATE$(i) < 0.0$ for some $i = 1,\dots,$NLINES with LINED$(4,i) \geq 0$;
or LINED$(1,i) < 2$ for some $i = 1,\dots,$NLINES;
or LINED$(2,i) < 1$ or LINED$(2,i) >$ NLINES for some $i = 1,\dots,$NLINES;
or LINED$(3,i) < 1$ or LINED$(3,i) >$ NLINES for some $i = 1,\dots,$NLINES;
or LINED$(2,i) =$ LINED$(3,i)$ for some $i = 1,\dots,$NLINES;
or NLCOMP$(k) = 0$, or $|\text{NLCOMP}(k)| >$ NLINES for a $k = 1,\dots,$NCOMP;
or $\sum_{k=1}^{n} |\text{NLCOMP}(k)| \neq \text{NLINES}$;
or LCOMP does not represent a valid permutation of the integers in [1, NLINES];
or one of the end points for a line i described by the user-supplied function (lines with LINED$(4,i) > 0$, for $i = 1,\dots,$NLINES) does not belong to the corresponding curve in the user-supplied function FBND;
or the intermediate points for the lines described as polygonal arcs (lines with LINED$(i) < 0$, for $i = 1,\dots,$NLINES) are overlapping.

IFAIL $= 2$

An error has occurred during the generation of the boundary mesh. It appears that NEDMX is not large enough, so you are advised to increase the value of NEDMX.

IFAIL $= 3$

An error has occurred during the generation of the boundary mesh. It appears that NVMAX is not large enough, so you are advised to increase the value of NVMAX.

IFAIL $= 4$

An error has occurred during the generation of the boundary mesh. Check the definition of each line (the parameter LINED) and each connected component of the boundary (the arguments NLCOMP, and LCOMP, as well as the co-ordinates of the characteristic points. Setting ITRACE > 0 may provide more details.

7 Accuracy

Not applicable.

8 Further Comments

The boundary mesh generation technique in this routine has a 'tree' structure. The boundary should be partitioned into geometrically simple segments (straight lines or curves) delimited by characteristic points. Then, the lines should be assembled into connected component of the boundary domain.

Using this strategy, the inputs to that routine can be built up, following the requirements stated in Section 5:

the characteristic and the user-supplied intermediate points:

NLINES, NUS, COORCH and COORUS;

the characteristic lines:

LINED, FBND, RATE;

finally the assembly of lines into the connected components of the boundary:

NCOMP, and

NLCOMP, LCOMP.

The example below details the use of this strategy.

9 Example

The NAG logo is taken as an example of a geometry with holes. The boundary has been partitioned in 40 lines characteristic points; including 4 for the exterior boundary and 36 for the logo itself. All line geometry specifications have been considered, see the description of LINED, including 4 lines defined as polygonal arc, 4 defined by a user-supplied function and all the others are straight lines.

Figure 1 top represents the boundary mesh of the NAG logo; there are 259 nodes and 259 edges. The Figure 1 middle and bottom represent the final mesh built using respectively the Delaunay–Voronoi (D06ABF) and the Advancing front (D06ACF) method.

9.1 Program Text

```
*     D06BAF Example Program Text
*     Mark 20 Release. NAG Copyright 2001.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NEDMX, NVMAX, NLINESX, NUS, NCOMPX, NVIMX,
     +                 MAXCAN, LRWORK, LIWORK
      PARAMETER        (NEDMX=300,NVMAX=1000,NLINESX=50,NUS=100,
     +                 NCOMPX=5,NVIMX=20,MAXCAN=10000,
     +                 LRWORK=12*NVMAX+3*MAXCAN+15,
     +                 LIWORK=8*NEDMX+55*NVMAX+MAXCAN+78)
*     .. Local Scalars ..
      DOUBLE PRECISION X0, XA, XB, XMAX, XMIN, Y0, YMAX, YMIN
      INTEGER          I, IFAIL, ITRACE, J, K, NCOMP, NEDGE, NELT,
     +                 NLINES, NPROPA, NV, NVB, NVINT, REFTK
      CHARACTER        PMESH
*     .. Local Arrays ..
      DOUBLE PRECISION COOR(2,NVMAX), COORCH(2,NLINESX), COORUS(2,NUS),
     +                 RATE(NLINESX), RUSER(4), 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 ..
      DOUBLE PRECISION FBND
      EXTERNAL         FBND
*     .. External Subroutines ..
      EXTERNAL         D06ABF, D06ACF, D06BAF
*     .. Intrinsic Functions ..
*
      INTRINSIC        ABS
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D06BAF 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
*
*     The ellipse boundary which envelops the NAg Logo
*     the N, the A and the G
*
      READ (NIN,*) (COORCH(1,J),J=1,NLINES)
      READ (NIN,*) (COORCH(2,J),J=1,NLINES)
*
      READ (NIN,*) (COORUS(1,J),J=1,4)
      READ (NIN,*) (COORUS(2,J),J=1,4)
*
*     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
*
      XMIN = COORCH(1,4)
      XMAX = COORCH(1,2)
      YMIN = COORCH(2,1)
      YMAX = COORCH(2,3)
*
      XA = (XMAX-XMIN)/2.D0
      XB = (YMAX-YMIN)/2.D0
*
      X0 = (XMIN+XMAX)/2.D0
      Y0 = (YMIN+YMAX)/2.D0
*
      RUSER(1) = XA
      RUSER(2) = XB
      RUSER(3) = X0
      RUSER(4) = Y0
      IUSER(1) = 0
*
      ITRACE = -1
*
*     Call to the 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
      NPROPA = 1
      NVINT = 0
      IFAIL = 0
*
*     Call to the 2D Delaunay-Voronoi mesh generator
*
      CALL D06ABF(NVB,NVINT,NVMAX,NEDGE,EDGE,NV,NELT,COOR,CONN,WEIGHT,
     +            NPROPA,ITRACE,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
      IF (PMESH.EQ.'N') THEN
         WRITE (NOUT,*)
     +     'Complete mesh (via the 2D Delaunay-Voronoi mesh'
         WRITE (NOUT,*) 'generator) 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 80 I = 1, NV
            WRITE (NOUT,99995) COOR(1,I), COOR(2,I)
   80    CONTINUE
*
         REFTK = 0
         DO 100 K = 1, NELT
            WRITE (NOUT,99994) CONN(1,K), CONN(2,K), CONN(3,K), REFTK
  100    CONTINUE
      END IF
*
*     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 (via the 2D Advancing front mesh'
         WRITE (NOUT,*) 'generator) 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
*
      DOUBLE PRECISION FUNCTION FBND(I,X,Y,RUSER,IUSER)
*     .. Scalar Arguments ..
      DOUBLE PRECISION                X, Y
      INTEGER                         I
*     .. Array Arguments ..
      DOUBLE PRECISION                RUSER(*)
      INTEGER                         IUSER(*)
*     .. Local Scalars ..
*
      DOUBLE PRECISION                RADIUS2, X0, XA, XB, Y0
*     .. Executable Statements ..
      XA = RUSER(1)
      XB = RUSER(2)
      X0 = RUSER(3)
      Y0 = RUSER(4)
*
      FBND = 0.D0
      IF (I.EQ.1) THEN
*
*     line 1,2,3, and 4: ellipse centred in (X0,Y0) with
*     XA and XB as coefficients
*
         FBND = ((X-X0)/XA)**2 + ((Y-Y0)/XB)**2 - 1.D0
      ELSE IF (I.EQ.2) THEN
*
*     line 24, 27, 33 and 38 are a circle centred in (X0,Y0)
*     with radius SQRT(RADIUS2)
*
         X0 = 20.5D0
         Y0 = 4.D0
         RADIUS2 = 4.25D0
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      ELSE IF (I.EQ.3) THEN
         X0 = 17.D0
         Y0 = 8.5D0
         RADIUS2 = 5.D0
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      ELSE IF (I.EQ.4) THEN
         X0 = 17.D0
         Y0 = 8.5D0
         RADIUS2 = 5.D0
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2

      ELSE IF (I.EQ.5) THEN
         X0 = 19.5D0
         Y0 = 4.D0
         RADIUS2 = 1.25D0
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      END IF
*
      RETURN
      END
```

9.2 Program Data

```
D06BAF Example Program Data
40                                                             :NLINES (m)
  9.5000  33.0000   9.5000 -14.0000  -4.0000  -2.0000   2.0000
  4.0000   2.0000  -2.0000  -4.0000  -2.0000   2.0000   4.0000
  7.0000   9.0000  13.0000  16.0000   9.0000  12.0000   7.0000
 10.0000  18.0000  21.0000  17.0000  20.0000  16.0000  20.0000
 15.5000  16.0000  18.0000  21.0000  16.0000  18.0000  18.5811
 21.0000  17.0000  20.0000  20.5000  23.0000          :(COORCH(1,1:m))
 -1.0000   7.5000  16.0000   7.5000   3.0000   3.0000   3.0000
  3.0000   7.0000   8.0000  12.0000  12.0000  12.0000  12.0000
  3.0000   3.0000   3.0000   3.0000   5.0000   5.0000  12.0000
 12.0000   2.0000   2.0000   3.0000   3.0000   5.0000   5.0000
  6.0000   6.0000   6.0000   6.0000   6.5000   6.5000  10.0811
 10.0811  10.7361  10.7361  12.0000  12.0000          :(COORCH(2,1:m))
 -2.6667  -3.3333   3.3333   2.6667                   :(COORUS(1,1:4))
  3.0000   3.0000   3.0000   3.0000                   :(COORUS(2,1:4))
15  1  2  1   0.9500   15  2  3  1   1.0500
15  3  4  1   0.9500   15  4  1  1   1.0500
 4  6  5 -1   1.0000   10 10  6  0   1.0000
10  7 10  0   1.0000    4  8  7 -3   1.0000
15 14  8  0   1.0000    4 13 14  0   1.0000
10  9 13  0   1.0000   10 12  9  0   1.0000
 4 11 12  0   1.0000   15  5 11  0   1.0000
 4 16 15  0   1.0000    7 19 16  0   1.0000
 4 20 19  0   1.0000    7 17 20  0   1.0000
 4 18 17  0   1.0000   13 22 18  0   1.0000
 5 21 22  0   1.0000   13 15 21  0   1.0000
 4 24 23  0   1.0000   10 24 32  2   1.0000
 4 31 32  0   1.0000    4 34 31  0   1.0000
10 34 35  3   1.0000    4 36 35  0   1.0000
 4 40 36  0   1.0000    4 39 40  0   1.0000
 4 38 39  0   1.0000    4 37 38  0   1.0000
10 37 33  4   1.0000    4 30 33  0   1.0000
 4 29 30  0   1.0000    4 27 29  0   1.0000
 4 28 27  0   1.0000   10 26 28  5   1.0000
 4 25 26  0   1.0000    4 23 25  0   1.0000 :(LINE(:,j),RATE(j),j=1,m)
 4                                        :NCOMP (n, number of contours)
 4                                         :number of lines in contour 1
 1  2  3  4                                          :lines of contour 1
10                                         :number of lines in contour 2
14 13 12 11 10  9  8  7  6  5                        :lines of contour 2
 8                                         :number of lines in contour 3
22 21 20 19 18 17 16 15                              :lines of contour 3
18                                         :number of lines in contour 4
30 29 28 27 26 25 24 23 40 39
38 37 36 35 34 33 32 31                              :lines of contour 4
'N'                                           :Printing option 'Y' or 'N'
```

9.3 Program Results

```
 D06BAF Example Program Results

 ANALYSIS OF THE BOUNDARY CREATED:
 THE BOUNDARY MESH CONTAINS    259 VERTEX AND    259 EDGES
 THERE ARE      4 COMPONENTS CONNECTED THE BOUNDARY
 THE  1-st COMPONENT CONTAINS      4 LINES IN ANTICLOCKWISE ORIENTATION
 THE  2-nd COMPONENT CONTAINS     10 LINES IN CLOCKWISE ORIENTATION
 THE  3-rd COMPONENT CONTAINS      8 LINES IN CLOCKWISE ORIENTATION
 THE  4-th COMPONENT CONTAINS     18 LINES IN CLOCKWISE ORIENTATION
 Boundary mesh characteristics
 NVB   =   259
 NEDGE =   259
 Complete mesh (via the 2D Delaunay-Voronoi mesh
 generator) characteristics
 NV    =   653
 NELT =  1051
 Complete mesh (via the 2D Advancing front mesh
 generator) characteristics
```

```
 NV   =   661
 NELT =  1067
```

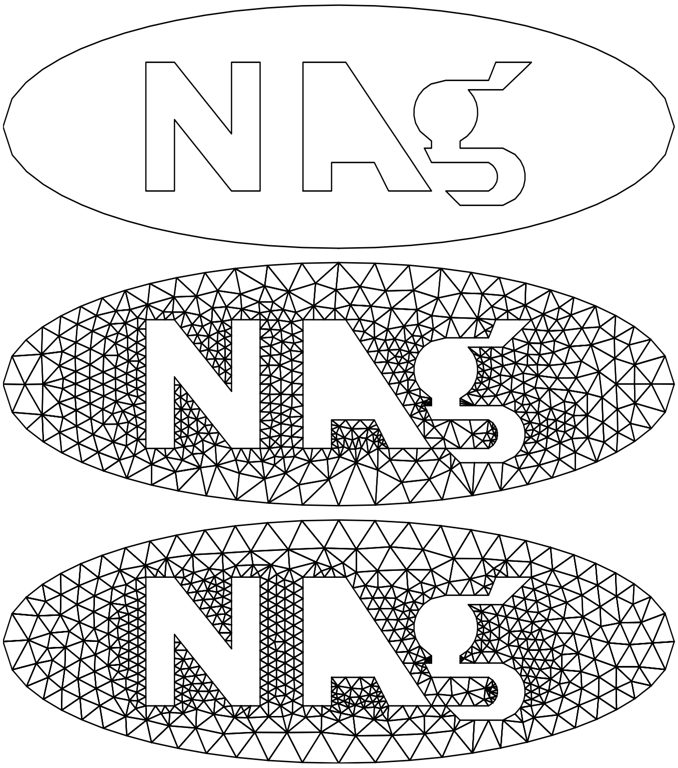

Figure 1
The boundary mesh (top), the Delaunay–Voronoi mesh (middle) and the Advancing Front mesh (bottom) of the NAG logo geometry