NAG 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, CRUS, SDCRUS, 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), SDCRUS, 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), FBND, CRUS(2,SDCRUS), 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 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 CRUS. In this case the line contains the points whose co-ordinates are stored in
COORCH$(:, j)$,
CRUS$(:, p)$,
CRUS$(:, p + 1), \ldots,$ CRUS$(:, p + r - 3)$,
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 FBND (lines with LINED$(4, i) > 0$, for $i = 1, 2, \ldots,$ NLINES) the two end points (LINED$(2, i)$ and LINED$(3, i)$) lie on the curve defined by index LINED$(4, i)$ in FBND, i.e.,

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

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

For all lines described as polygonal arcs (lines with LINED$(4, i) < 0$, for $i = 1, 2, \ldots,$ NLINES) the sets of intermediate points (i.e., $[-$LINED$(4, i) : -$LINED$(4, i) +$ LINED$(1, i) - 3]$ for all i such that LINED$(4, i) < 0$) are not overlapping. This can be expressed as:

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

or

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

for a j such that $j = 1, 2, \ldots,$ NLINES, $j \neq i$ and LINED$(4, j) < 0$.

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

FBND must be supplied to calculate the value of the function which describes the curve $\{(x, y) \in \mathbb{R}^2;$ 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 Library.

The specification of FBND 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 subroutine.

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: CRUS(2,SDCRUS) – ***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 $p = -\text{LINED}(4,i)$, then CRUS$(1,k)$, for $k = p, p+1, \ldots, p+\text{LINED}(1,i)-3$, must contain the x co-ordinate of the consecutive intermediate points for this line. Similarly CRUS$(2,k)$, for $k = p, p+1, \ldots, p+\text{LINED}(1,i)-3$, must contain the corresponding y co-ordinate.

6: SDCRUS – INTEGER *Input*

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

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

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

Constraints:

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

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

On entry: 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 $k = 1, 2, \ldots, \text{NCOMP}$.

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: NVMAX $\geq$ NLINES.

12: NEDMX – INTEGER *Input*

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

Constraint: NEDMX ≥ 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: COOR$(1, i)$ will contain the x co-ordinate of the ith boundary mesh vertex generated, for $i = 1, \ldots, \text{NVB}$; while 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. EDGE$(1, j)$ and EDGE$(2, j)$ will contain the vertex numbers of the two end points of the jth boundary edge. EDGE$(3, j)$ is a reference number for the jth boundary edge and

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

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

17: ITRACE – INTEGER *Input*

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

ITRACE $= 0$ or ITRACE < -1
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 $= -1$
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.

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.

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

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

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

You are free to use these arrays to supply information to this function from the calling subroutine.

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

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

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 Section 3.3 in the Essential Introduction 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 $< 2 \times (\text{NLINES} + \text{SDCRUS}) + 2 \times \max_{i=1,2,\ldots,m}\{\text{LINED}(1,i)\} \times \text{NLINES}$;

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

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 FBND and all the others are straight lines.

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, SDCRUS, NCOMPX, NVIMX,
     +                 MAXCAN, LRWORK, LIWORK
      PARAMETER        (NEDMX=1000,NVMAX=5000,NLINESX=50,SDCRUS=1000,
     +                 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), CRUS(2,SDCRUS),
     +                 RATE(NLINESX), RUSER(4), RWORK(LRWORK),
     +                 WEIGHT(NVIMX)
      INTEGER          CONN(3,2*NVMAX+5), EDGE(3,NEDMX), IUSER(1),
     +                 IWORK(LIWORK), LCOMP(NLINESX), LINED(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,*) (CRUS(1,J),J=1,4)
      READ (NIN,*) (CRUS(2,J),J=1,4)
*
*     The Lines of the boundary mesh
*
      READ (NIN,*) ((LINED(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 = 1
*
      CALL D06BAF(NLINES,COORCH,LINED,FBND,CRUS,SDCRUS,RATE,NCOMP,
     +            NLCOMP,LCOMP,NVMAX,NEDMX,NVB,COOR,NEDGE,EDGE,ITRACE,
     +            RUSER,IUSER,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,99993) ' ** D06BAF returned with IFAIL = ', IFAIL
         GO TO 160
      ELSE 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'
         GO TO 160
      END IF
*
*     Initialise mesh control parameters
*
```

```
      ITRACE = 0
      NPROPA = 1
      NVINT = 0
      IFAIL = 1
*
*     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 (IFAIL.NE.0) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,99993) ' ** D06ABF returned with IFAIL = ', IFAIL
         GO TO 160
      ELSE IF (PMESH.EQ.'N') THEN
         WRITE (NOUT,*) 'Complete mesh (via the 2D Delaunay-Voronoi'
         WRITE (NOUT,*) 'mesh 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 = 1
*
      CALL D06ACF(NVB,NVINT,NVMAX,NEDGE,EDGE,NV,NELT,COOR,CONN,WEIGHT,
     +            ITRACE,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,99993) ' ** D06ACF returned with IFAIL = ', IFAIL
         GO TO 160
      ELSE 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
*
  160 CONTINUE
*
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)
99993 FORMAT (1X,A,I5)
      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 7, lower arc on letter n, is a circle centred in (X0,Y0)
*        with radius SQRT(RADIUS2)
*
         X0 = 0.5D0
         Y0 = 6.25D0
         RADIUS2 = 20.3125D0
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      ELSE IF (I.EQ.3) THEN
*
*        line 11, upper arc on letter n, is a circle centred in (X0,Y0)
*        with radius SQRT(RADIUS2)
*
         X0 = 1.0D0
         Y0 = 4.0D0
         RADIUS2 = 9.0D0 + (11.0D0-Y0)**2
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      ELSE IF (I.EQ.4) THEN
*
*        line 15, upper arc on letter a, is a circle centred in (X0,Y0)
*        with radius SQRT(RADIUS2) touching point (5,11).
*
         X0 = 8.5D0
         Y0 = 2.75D0
         RADIUS2 = (X0-5.0D0)**2 + (11.0D0-Y0)**2
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      ELSE IF (I.EQ.5) THEN
*
*        line 25, lower arc on hat of 'a', is a circle centred in (X0,Y0)
*        with radius SQRT(RADIUS2) touching point (11,10).
*
         X0 = 8.5D0
         Y0 = 4.0D0
         RADIUS2 = 2.5D0**2 + (10.0D0-Y0)**2
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      ELSE IF (I.EQ.6) THEN
*
*        lines 20, 21 and 22, belly of letter a, is an ellipse centered
*        in (X0, Y0) with semi-axes 3.5 and 2.75.
*
         X0 = 8.5D0
         Y0 = 5.75D0
         FBND = ((X-X0)/3.5D0)**2 + ((Y-Y0)/2.75D0)**2 - 1.D0
```

```
      ELSE IF (I.EQ.7) THEN
*
*        lines 43, 44 and 45, outer curve on bottom of 'g', is an ellipse
*        centered in (X0, Y0) with semi-axes 3.5 and 2.5.
*
         X0 = 17.5D0
         Y0 = 2.5D0
         FBND = ((X-X0)/3.5D0)**2 + ((Y-Y0)/2.5D0)**2 - 1.D0
      ELSE IF (I.EQ.8) THEN
*
*        lines 28, 29 and 30, inner curve on bottom of 'g', is an ellipse
*        centered in (X0, Y0) with semi-axes 2.0 and 1.5.
*
         X0 = 17.5D0
         Y0 = 2.5D0
         FBND = ((X-X0)/2.0D0)**2 + ((Y-Y0)/1.5D0)**2 - 1.D0
      ELSE IF (I.EQ.9) THEN
*
*        line 42, inner curve on lower middle of 'g', is an ellipse
*        centered in (X0, Y0) with semi-axes 1.5 and 0.5.
*
         X0 = 17.5D0
         Y0 = 5.5D0
         FBND = ((X-X0)/1.5D0)**2 + ((Y-Y0)/0.5D0)**2 - 1.D0
      ELSE IF (I.EQ.10) THEN
*
*        line 31, outer curve on lower middle of 'g', is an ellipse
*        centered in (X0, Y0) with semi-axes 2.0 and 1.5.
*
         X0 = 17.5D0
         Y0 = 5.5D0
         FBND = ((X-X0)/3.0D0)**2 + ((Y-Y0)/1.5D0)**2 - 1.D0
      ELSE IF (I.EQ.11) THEN
*
*        line 41, inner curve on upper middle of 'g', is an ellipse
*        centered in (X0, Y0) with semi-axes 1.0 and 1.0.
*
         X0 = 17.0D0
         Y0 = 5.5D0
         FBND = ((X-X0)/1.0D0)**2 + ((Y-Y0)/1.0D0)**2 - 1.D0
      ELSE IF (I.EQ.12) THEN
*
*        line 32, outer curve on upper middle of 'g', is an ellipse
*        centered in (X0, Y0) with semi-axes 1.5 and 1.1573.
*
         X0 = 16.0D0
         Y0 = 5.5D0
         FBND = ((X-X0)/1.5D0)**2 + ((Y-Y0)/1.1573D0)**2 - 1.D0
      ELSE IF (I.EQ.13) THEN
*
*        lines 33, 33, 34, 39 and 40, upper portion of 'g', is an ellipse
*        centered in (X0, Y0) with semi-axes 3.0 and 2.75.
         X0 = 17.0D0
         Y0 = 9.25D0
         FBND = ((X-X0)/3.0D0)**2 + ((Y-Y0)/2.75D0)**2 - 1.D0
      END IF
*
      RETURN
      END
```

9.2 Program Data

```
D06BAF Example Program Data
45                                                      :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   4.0000   2.0000
  5.0000   6.0000  11.0000  11.0000   8.5000   5.0000   8.5000
 11.5000  13.0000  14.0000  13.0000  13.0000
 14.0000  15.5000  17.5000  17.5000  21.0000  19.5000  17.5000
```

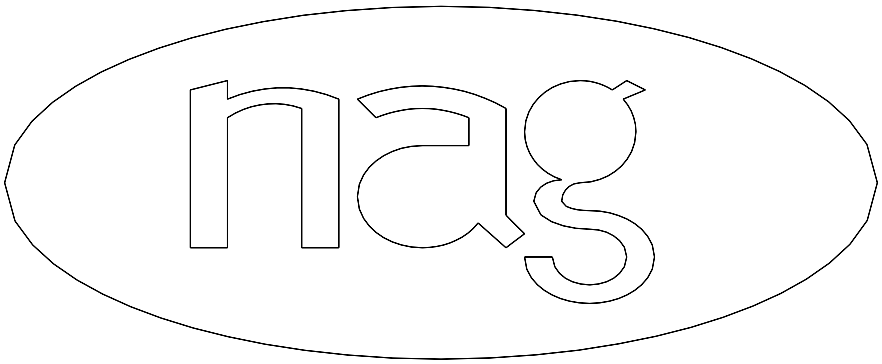
```
 17.5000  16.0000  14.5000  17.0000  16.0000  20.0000  14.0000
 19.3142  17.0000  20.5000  18.7249  19.5000           : End of X coords
 -3.0000   6.5000  16.0000   6.5000
  3.0000   3.0000   3.0000   3.0000  11.0000  10.0000  11.5000
 12.0000  11.0000  10.5000
 11.0000  10.0000  10.0000   8.5000   8.5000   5.7500   3.0000
  4.3335   3.0000   3.7500   4.7500  10.5000
  2.5000   2.5000   0.0000   1.0000   2.5000   2.5000   5.0000
  4.0000   5.5000   5.5000   6.5000   6.6573   9.2500   9.2500
 11.0000  12.000   11.5000  11.5000  12.0000           : End of Y coords
 -2.6667  -3.3333   3.3333   2.6667                    :(Poly (X))
  3.0000   3.0000   3.0000   3.0000                    :(Poly (Y))
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 14 10  2   1.0000   10  7 14  0   1.0000
 4  8  7  0   1.0000   10 13  8  0   1.0000
10 13  9  3   1.0000   10 12  9  0   1.0000
 4 11 12  0   1.0000   15  5 11  0   1.0000
15 26 15  4   1.0000   10 26 25  0   1.0000
 4 25 24  0   1.0000    4 24 23  0   1.0000
 4 23 22  0   1.0000   10 21 22  6   1.0000
10 20 21  6   1.0000   10 19 20  6   1.0000
 4 19 18  0   1.0000    5 18 17  0   1.0000
15 17 16  5   1.0000    4 16 15  0   1.0000
 4 27 28  0   1.0000    7 28 30  8   1.0000
 7 30 32  8   1.0000    7 32 34  8   1.0000
 6 36 34 10   1.0000    6 38 36 12   1.0000
10 40 38 13   1.0000   10 42 40 13   1.0000
 8 44 42 13   1.0000    4 44 45  0   1.0000
 4 45 43  0   1.0000    4 43 41  0   1.0000
 6 39 41 13   1.0000   10 37 39 13   1.0000
 6 37 35 11   1.0000    6 35 33  9   1.0000
10 31 33  7   1.0000   10 29 31  7   1.0000
10 27 29  7   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    (Ellipse)
10                                    :number of lines in contour 2
14 13 12 11 10  9  8  7  6  5         :lines of contour 2   (Letter N)
12                                    :number of lines in contour 3
18 19 20 21 22 23 24 25 26 15 16 17   :lines of contour 3   (Letter A)
19                                    :number of lines in contour 4
27 28 29 30 31 32 33 34 35 36 37 38
39 40 41 42 43 44 45                  :lines of contour 4   (Letter G)
'N'                                      :Printing option 'Y' or 'N'
```

9.3 Program Results

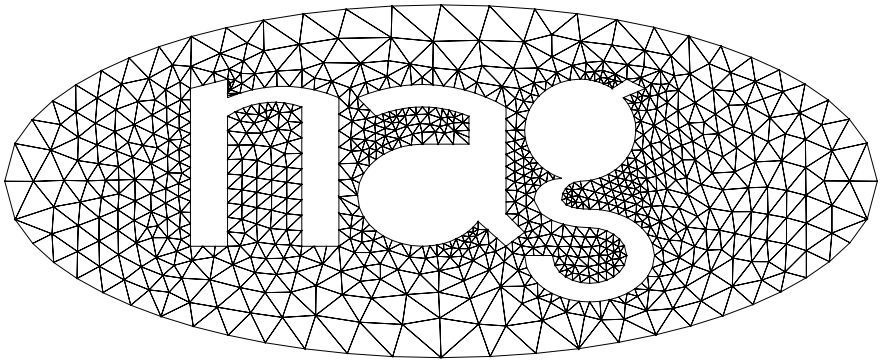
```
 D06BAF Example Program Results

 ANALYSIS OF THE BOUNDARY CREATED:
 THE BOUNDARY MESH CONTAINS    332 VERTEX AND    332 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     12 LINES IN ANTICLOCKWISE ORIENTATION
 THE  4-th COMPONENT CONTAINS     19 LINES IN CLOCKWISE ORIENTATION
 Boundary mesh characteristics
 NVB   =   332
 NEDGE =   332
 Complete mesh (via the 2D Delaunay-Voronoi
 mesh generator) characteristics
 NV   =   903
 NELT =  1478
 Complete mesh (via the 2D Advancing front mesh
 generator) characteristics
 NV   =   924
 NELT =  1520
```

Example Program
Boundary Mesh of the NAG Logo with 259 Nodes and 259 Edges



Final Mesh Built Using the Delaunay-Voronoi Method



Final Mesh Built Using the Advancing Front Method