NAG Fortran Library Routine Document

D06DBF

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

D06DBF joins together (restitches) two adjacent, or overlapping, meshes.

2 Specification

```
 SUBROUTINE D06DBF (EPS, NV1, NELT1, NEDGE1, COOR1, EDGE1, CONN1, REFT1,
1                   NV2, NELT2, NEDGE2, COOR2, EDGE2, CONN2, REFT2, NV3,
2                   NELT3, NEDGE3, COOR3, EDGE3, CONN3, REFT3, ITRACE,
3                   IWORK, LIWORK, IFAIL)
 INTEGER            NV1, NELT1, NEDGE1, EDGE1(3,NEDGE1), CONN1(3,NELT1),
1                   REFT1(NELT1), NV2, NELT2, NEDGE2, EDGE2(3,NEDGE2),
2                   CONN2(3,NELT2), REFT2(NELT2), NV3, NELT3, NEDGE3,
3                   EDGE3(3,*), CONN3(3,*), REFT3(*), ITRACE,
4                   IWORK(LIWORK), LIWORK, IFAIL
 double precision   EPS, COOR1(2,NV1), COOR2(2,NV2), COOR3(2,*)
```

3 Description

D06DBF joins together two adjacent, or overlapping, meshes. If the two meshes are adjacent then vertices belonging to the part of the boundary forming the common interface should coincide. If the two meshes overlap then vertices and triangles in the overlapping zone should coincide too.

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

4 References

None.

5 Parameters

1: EPS – ***double precision*** *Input*

On entry: the relative precision of the restitching of the two input meshes (see Section 8).

Suggested value: 0.001.

Constraint: EPS > 0.0.

2: NV1 – INTEGER *Input*

On entry: the total number of vertices in the first input mesh.

Constraint: NV1 ≥ 3.

3: NELT1 – INTEGER *Input*

On entry: the number of triangular elements in the first input mesh.

Constraint: NELT1 $\leq 2 \times \text{NV1} - 1$.

4: NEDGE1 – INTEGER *Input*

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

Constraint: NEDGE1 ≥ 1.

5: COOR1(2,NV1) – ***double precision*** array *Input*

On entry: COOR1$(1, i)$ contains the x co-ordinate of the ith vertex of the first input mesh, for $i = 1, \ldots,$ NV1; while COOR1$(2, i)$ contains the corresponding y co-ordinate.

6: EDGE1(3,NEDGE1) – INTEGER array *Input*

On entry: the specification of the boundary edges of the first input mesh. EDGE1$(1,j)$ and EDGE1$(2,j)$ contain the vertex numbers of the two end points of the jth boundary edge. EDGE1$(3,j)$ is a user-supplied tag for the jth boundary edge.

Constraint: $1 \leq$ EDGE1$(i, j) \leq$ NV1 and EDGE1$(1, j) \neq$ EDGE1$(2, j)$, for $i = 1, 2$ and $j = 1, 2, \ldots,$ NEDGE1.

7: CONN1(3,NELT1) – INTEGER array *Input*

On entry: the connectivity between triangles and vertices of the first input mesh. For each triangle j, CONN1(i,j) gives the indices of its three vertices (in anticlockwise order), for $i = 1, 2, 3$ and $j = 1, \ldots,$ NELT1.

Constraints:

$1 \leq$ CONN1$(i,j) \leq$ NV1;
CONN1$(1,j) \neq$ CONN1$(2,j)$;
CONN1$(1, j) \neq$ CONN1$(3, j)$ and CONN1$(2, j) \neq$ CONN1$(3, j)$, for $i = 1, 2, 3$ and $j = 1, 2, \ldots,$ NELT1.

8: REFT1(NELT1) – INTEGER array *Input*

On entry: REFT1(k) contains the user-supplied tag of the kth triangle from the first input mesh, for $k = 1, \ldots,$ NELT1.

9: NV2 – INTEGER *Input*

On entry: the total number of vertices in the second input mesh.

Constraint: NV2 ≥ 3.

10: NELT2 – INTEGER *Input*

On entry: the number of triangular elements in the second input mesh.

Constraint: NELT2 $\leq 2 \times$ NV2 $- 1$.

11: NEDGE2 – INTEGER *Input*

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

Constraint: NEDGE2 ≥ 1.

12: COOR2(2,NV2) – ***double precision*** array *Input*

On entry: COOR2$(1, i)$ contains the x co-ordinate of the ith vertex of the second input mesh, for $i = 1, \ldots,$ NV2; while COOR2$(2, i)$ contains the corresponding y co-ordinate.

13: EDGE2(3,NEDGE2) – INTEGER array *Input*

On entry: the specification of the boundary edges of the second input mesh. EDGE2$(1,j)$ and EDGE2$(2,j)$ contain the vertex numbers of the two end points of the jth boundary edge. EDGE2$(3,j)$ is a user-supplied tag for the jth boundary edge.

Constraint: $1 \leq \mathrm{EDGE2}(i,j) \leq \mathrm{NV2}$ and $\mathrm{EDGE2}(1,j) \neq \mathrm{EDGE2}(2,j)$, for $i = 1,2$ and $j = 1,2,\ldots,\mathrm{NEDGE2}$.

14: CONN2(3,NELT2) – INTEGER array *Input*

On entry: the connectivity between triangles and vertices of the second input mesh. For each triangle j, CONN2(i,j) gives the indices of its three vertices (in anticlockwise order), for $i = 1,2,3$ and $j = 1,\ldots,\mathrm{NELT2}$.

Constraints:

$1 \leq \mathrm{CONN2}(i,j) \leq \mathrm{NV2}$;
$\mathrm{CONN2}(1,j) \neq \mathrm{CONN2}(2,j)$;
$\mathrm{CONN2}(1,j) \neq \mathrm{CONN2}(3,j)$ and $\mathrm{CONN2}(2,j) \neq \mathrm{CONN2}(3,j)$, for $i = 1,2,3$ and $j = 1,2,\ldots,\mathrm{NELT2}$.

15: REFT2(NELT2) – INTEGER array *Input*

On entry: REFT2(k) contains the user-supplied tag of the kth triangle from the second input mesh, for $k = 1,\ldots\mathrm{NELT2}$.

16: NV3 – INTEGER *Output*

On exit: the total number of vertices in the resulting mesh.

17: NELT3 – INTEGER *Output*

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

18: NEDGE3 – INTEGER *Output*

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

19: COOR3(2,*) – ***double precision*** array *Output*

Note: the second dimension of the array COOR3 must be at least $\mathrm{NV1} + \mathrm{NV2}$. This may be reduced to NV3 once that value is known.

On exit: COOR3$(1,i)$ will contain the x co-ordinate of the ith vertex of the resulting mesh, for $i = 1,\ldots,\mathrm{NV3}$; while COOR3$(2,i)$ will contain the corresponding y co-ordinate.

20: EDGE3(3,*) – INTEGER array *Output*

Note: the second dimension of the array EDGE3 must be at least $\mathrm{NEDGE1} + \mathrm{NEDGE2}$. This may be reduced to NEDGE3 once that value is known.

On exit: the specification of the boundary edges of the resulting mesh. EDGE3(i,j) will contain the vertex number of the ith end point ($i = 1,2$) of the jth boundary or interface edge.

If the two meshes overlap, EDGE3$(3,j)$ will contain the same tag as the corresponding edge belonging to the first and/or the second input mesh.

If the two meshes are adjacent,

(i) if the jth edge is part of the partition interface, then EDGE3$(3,j)$ will contain the value $1000 \times k_1 + k_2$ where k_1 and k_2 are the tags for the same edge of the first and the second mesh respectively;

(ii) otherwise, EDGE3$(3,j)$ will contain the same tag as the corresponding edge belonging to the first and/or the second input mesh.

21: CONN3(3,∗) – INTEGER array *Output*

Note: the second dimension of the array CONN3 must be at least NELT1 + NELT2. This may be reduced to NELT3 once that value is known.

On exit: the connectivity between triangles and vertices of the resulting mesh. CONN3(i,j) will give the indices of its three vertices (in anticlockwise order), for $i = 1, 2, 3$ and $j = 1, \ldots, \mathrm{NELT3}$.

22: REFT3(∗) – INTEGER array *Output*

Note: the dimension of the array REFT3 must be at least NELT1 + NELT2. This may be reduced to NELT3 once that value is known.

On exit: if the two meshes form a partition, REFT3(k) will contain the same tag as the corresponding triangle belonging to the first or the second input mesh, for $k = 1, \ldots, \mathrm{NELT3}$. If the two meshes overlap, then REFT3(k) will contain the value $1000 \times k_1 + k_2$ where k_1 and k_2 are the user-supplied tags for the same triangle of the first and the second mesh respectively, for $k = 1, \ldots, \mathrm{NELT3}$.

23: ITRACE – INTEGER *Input*

On entry: the level of trace information required from D06DBF.

ITRACE ≤ 0

No output is generated.

ITRACE ≥ 1

Details about the common vertices, edges and triangles to both meshes are printed on the current advisory message unit (see X04ABF).

24: IWORK(LIWORK) – INTEGER array *Workspace*
25: LIWORK – INTEGER *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D06DBF is called.

Constraint:
LIWORK $\geq 2 \times \mathrm{NV1} + 3 \times \mathrm{NV2} + \mathrm{NELT1} + \mathrm{NELT2} + \mathrm{NEDGE1} + \mathrm{NEDGE2} + 1024$.

26: 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, EPS ≤ 0.0,
or NV1 < 3,
or NELT1 $> 2 \times \mathrm{NV1} - 1$,

or NEDGE1 < 1,
or $\text{EDGE1}(i,j) < 1$ or $\text{EDGE1}(i,j) > \text{NV1}$ for some $i = 1, 2$ and $j = 1, \ldots, \text{NEDGE1}$,
or $\text{EDGE1}(1,j) = \text{EDGE1}(2,j)$ for some $j = 1, \ldots, \text{NEDGE1}$,
or $\text{CONN1}(i,j) < 1$ or $\text{CONN1}(i,j) > \text{NV1}$ for some $i = 1, 2, 3$ and $j = 1, \ldots, \text{NELT1}$,
or $\text{CONN1}(1,j) = \text{CONN1}(2,j)$ or $\text{CONN1}(1,j) = \text{CONN1}(3,j)$ or $\text{CONN1}(2,j) = \text{CONN1}(3,j)$ for some $j = 1, \ldots, \text{NELT1}$,
or NV2 < 3,
or $\text{NELT2} > 2 \times \text{NV2} - 1$,
or NEDGE2 < 1,
or $\text{EDGE2}(i,j) < 1$ or $\text{EDGE2}(i,j) > \text{NV2}$ for some $i = 1, 2$ and $j = 1, \ldots, \text{NEDGE2}$,
or $\text{EDGE2}(1,j) = \text{EDGE2}(2,j)$ for some $j = 1, \ldots, \text{NEDGE2}$,
or $\text{CONN2}(i,j) < 1$ or $\text{CONN2}(i,j) > \text{NV2}$ for some $i = 1, 2, 3$ and $j = 1, \ldots, \text{NELT2}$,
or $\text{CONN2}(1,j) = \text{CONN2}(2,j)$ or $\text{CONN2}(1,j) = \text{CONN2}(3,j)$ or $\text{CONN2}(2,j) = \text{CONN2}(3,j)$ for some $j = 1, \ldots, \text{NELT2}$,
or $\text{LIWORK} < 2 \times \text{NV1} + 3 \times \text{NV2} + \text{NELT1} + \text{NELT2} + \text{NEDGE1} + \text{NEDGE2} + 1024$.

IFAIL $= 2$

Using the input precision EPS, the routine has detected fewer than two coincident vertices between the two input meshes. You are advised to try another value of EPS; if this error still occurs the two meshes are probably not stitchable.

IFAIL $= 3$

A serious error has occurred in an internal call to the restitching routine. You should check the input of the two meshes, especially the edge/vertex and/or the triangle/vertex connectivities. If the problem persists, contact NAG.

IFAIL $= 4$

The routine has detected a different number of coincident triangles from the two input meshes in the overlapping zone. You should check the input of the two meshes, especially the triangle/vertex connectivities.

IFAIL $= 5$

The routine has detected a different number of coincident edges from the two meshes on the partition interface. You should check the input of the two meshes, especially the edge/vertex connectivities.

7 Accuracy

Not applicable.

8 Further Comments

D06DBF finds all the common vertices between the two input meshes using the relative precision of the restitching parameter EPS. You are advised to vary the value of EPS in the neighbourhood of 0.001 with ITRACE ≥ 1 to get the optimal value for the meshes under consideration.

9 Example

For this routine two examples are presented. There is a single example program for D06DBF, with a main program and the code to solve the two example problems is given in the (sub)programs EX1 and EX2.

Example 1 (EX1)

This example involves the unit square $[0, 1]^2$ meshed uniformly, and then translated by a vector $\vec{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$ (using D06DAF). This translated mesh is then restitched with the original mesh. Two cases are considered:

(a) overlapping meshes ($u_1 = 15.0$, $u_2 = 17.0$),

(b) partitioned meshes ($u_1 = 19.0$, $u_2 = 0.0$).

The mesh on the unit square has 400 vertices, 722 triangles and its boundary has 76 edges. In the overlapping case the resulting geometry is shown in Figures 1 and 2. The resulting geometry for the partitioned meshes is shown in Figure 3.

Example 2 (EX2)

This example restitches three geometries by calling the routine D06DBF twice. The result is a mesh with three partitions. The first geometry is meshed by the Delaunay–Voronoi process (using D06ABF), the second one meshed by an Advancing Front algorithm (using D06ACF), while the third one is the result of a rotation (by $-\pi/2$) of the second one (using D06DAF). The resulting geometry is shown in Figures 4 and 5.

9.1 Program Text

```
*     D06DBF Example Program Text
*     Mark 20 Release. NAG Copyright 2001.
*     .. Parameters ..
      INTEGER           NOUT
      PARAMETER         (NOUT=6)
*     .. External Subroutines ..
      EXTERNAL          EX1, EX2
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D06DBF Example Program Results'
      CALL EX1
      CALL EX2
      STOP
      END
*
      SUBROUTINE EX1
*     .. Parameters ..
      INTEGER           NIN, NOUT
      PARAMETER         (NIN=5,NOUT=6)
      INTEGER           NBEDMX, NVMAX, NELTMAX, LIWORK, NTRANS, LRWORK
      PARAMETER         (NBEDMX=200,NVMAX=900,NELTMAX=2*(NVMAX-1),
     +                  LIWORK=4*NVMAX+2*NELTMAX+2*NBEDMX,NTRANS=1,
     +                  LRWORK=12*NTRANS)
*     .. Local Scalars ..
      DOUBLE PRECISION EPS
      INTEGER           I, IFAIL, IMAX, ITRACE, ITRANS, JMAX, JTRANS, K,
     +                  KTRANS, NEDGE1, NEDGE2, NEDGE3, NELT1, NELT2,
     +                  NELT3, NV1, NV2, NV3, REFTK
      CHARACTER         PMESH
*     .. Local Arrays ..
      DOUBLE PRECISION COOR1(2,NVMAX), COOR2(2,NVMAX), COOR3(2,NVMAX),
     +                  RWORK(LRWORK), TRANS(6,NTRANS)
      INTEGER           CONN1(3,NELTMAX), CONN2(3,NELTMAX),
     +                  CONN3(3,NELTMAX), EDGE1(3,NBEDMX),
     +                  EDGE2(3,NBEDMX), EDGE3(3,NBEDMX), ITYPE(NTRANS),
     +                  IWORK(LIWORK), REFT1(NELTMAX), REFT2(NELTMAX),
     +                  REFT3(NELTMAX)
*     .. External Subroutines ..
      EXTERNAL          D06DAF, D06DBF
*     .. Intrinsic Functions ..
      INTRINSIC         DBLE
*     .. Executable Statements ..
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Example 1'
      WRITE (NOUT,*)
      IMAX = 20
      JMAX = IMAX
*
*     Skip heading in data file
*
      READ (NIN,*)
      READ (NIN,*)
```

```
*
*     Read the mesh : coordinates and connectivity of the 1st domain
*
      READ (NIN,*) NV1, NELT1
      NV2 = NV1
      NELT2 = NELT1
      DO 20 I = 1, NV1
         READ (NIN,*) COOR1(1,I), COOR1(2,I)
   20 CONTINUE
*
      DO 40 K = 1, NELT1
         READ (NIN,*) CONN1(1,K), CONN1(2,K), CONN1(3,K), REFTK
         REFT1(K) = 1
         REFT2(K) = 2
   40 CONTINUE
*
      READ (NIN,*) PMESH
*
*     the Edges of the boundary
*
      NEDGE1 = 0
      DO 60 I = 1, IMAX - 1
         NEDGE1 = NEDGE1 + 1
         EDGE1(1,NEDGE1) = I
         EDGE1(2,NEDGE1) = I + 1
         EDGE1(3,NEDGE1) = 1
   60 CONTINUE
*
      DO 80 I = 1, JMAX - 1
         NEDGE1 = NEDGE1 + 1
         EDGE1(1,NEDGE1) = I*IMAX
         EDGE1(2,NEDGE1) = (I+1)*IMAX
         EDGE1(3,NEDGE1) = 1
   80 CONTINUE
*
      DO 100 I = 1, IMAX - 1
         NEDGE1 = NEDGE1 + 1
         EDGE1(1,NEDGE1) = IMAX*JMAX - I + 1
         EDGE1(2,NEDGE1) = IMAX*JMAX - I
         EDGE1(3,NEDGE1) = 1
  100 CONTINUE
*
      DO 120 I = 1, JMAX - 1
         NEDGE1 = NEDGE1 + 1
         EDGE1(1,NEDGE1) = (JMAX-I)*IMAX + 1
         EDGE1(2,NEDGE1) = (JMAX-I-1)*IMAX + 1
         EDGE1(3,NEDGE1) = 1
  120 CONTINUE
*
      NEDGE2 = NEDGE1
*
      DO 220 KTRANS = 1, 2
*
*     Translation of the 1st domain to obtain the 2nd domain
*     KTRANS = 1 leading to a domains overlapping
*     KTRANS = 2 leading to a domains partition
*
         IF (KTRANS.EQ.1) THEN
            ITRANS = IMAX - 5
            JTRANS = JMAX - 3
         ELSE
            ITRANS = IMAX - 1
            JTRANS = 0
         END IF
*
         ITYPE(1) = 1
         TRANS(1,1) = DBLE(ITRANS)/DBLE(IMAX-1)
         TRANS(2,1) = DBLE(JTRANS)/DBLE(JMAX-1)
         ITRACE = 0
         IFAIL = 0
*
```

```
         CALL D06DAF(NV2,NEDGE2,NELT2,NTRANS,ITYPE,TRANS,COOR1,EDGE1,
     +               CONN1,COOR2,EDGE2,CONN2,ITRACE,RWORK,LRWORK,IFAIL)
*
         DO 140 I = 1, NEDGE2
            EDGE2(3,I) = 2
  140    CONTINUE
*
*     Call to the restitching driver
*
         ITRACE = 0
         IFAIL = 0
         EPS = 1.D-2
*
         CALL D06DBF(EPS,NV1,NELT1,NEDGE1,COOR1,EDGE1,CONN1,REFT1,NV2,
     +               NELT2,NEDGE2,COOR2,EDGE2,CONN2,REFT2,NV3,NELT3,
     +               NEDGE3,COOR3,EDGE3,CONN3,REFT3,ITRACE,IWORK,LIWORK,
     +               IFAIL)
*
         IF (PMESH.EQ.'N') THEN
            IF (KTRANS.EQ.1) THEN
               WRITE (NOUT,*) 'The restitched mesh characteristics'
               WRITE (NOUT,*) 'in the overlapping case'
            ELSE
               WRITE (NOUT,*) 'in the partition case'
            END IF
            WRITE (NOUT,99999) 'NV    =', NV3
            WRITE (NOUT,99999) 'NELT  =', NELT3
            WRITE (NOUT,99999) 'NEDGE =', NEDGE3
         ELSE IF (PMESH.EQ.'Y') THEN
*
*     Output the mesh to view it using the NAG Graphics Library
*
            WRITE (NOUT,99998) NV3, NELT3, NEDGE3
            DO 160 I = 1, NV3
               WRITE (NOUT,99997) COOR3(1,I), COOR3(2,I)
  160       CONTINUE
*
            DO 180 K = 1, NELT3
               WRITE (NOUT,99996) CONN3(1,K), CONN3(2,K), CONN3(3,K),
     +           REFT3(K)
  180       CONTINUE
*
            DO 200 K = 1, NEDGE3
               WRITE (NOUT,99998) EDGE3(1,K), EDGE3(2,K), EDGE3(3,K)
  200       CONTINUE
         ELSE
            WRITE (NOUT,*) 'Problem with the printing option Y or N'
            STOP
         END IF
  220 CONTINUE
*
99999 FORMAT (1X,A,I6)
99998 FORMAT (1X,3I10)
99997 FORMAT (2(2X,E12.6))
99996 FORMAT (1X,4I10)
      END
*
      SUBROUTINE EX2
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NEDMX, NVMAX, NLINESX, NUS, NCOMPX, NVIMX,
     +                 MAXCAN, LRWORK, LIWORK
      PARAMETER        (NEDMX=200,NVMAX=700,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 EPS
      INTEGER          I, IFAIL, ITRACE, J, K, NCOMP, NEDGE1, NEDGE2,
     +                 NEDGE3, NEDGE4, NEDGE5, NELT1, NELT2, NELT3,
     +                 NELT4, NELT5, NLINES, NPROPA, NQINT, NTRANS, NV1,
```

```
     +                 NV2, NV3, NV4, NV5, NVB1, NVB2, NVFIX, NVINT
      CHARACTER        PMESH
*     .. Local Arrays ..
      DOUBLE PRECISION COOR1(2,NVMAX), COOR2(2,NVMAX), COOR3(2,NVMAX),
     +                 COOR4(2,NVMAX), COOR5(2,NVMAX),
     +                 COORCH(2,NLINESX), COORUS(2,NUS), RATE(NLINESX),
     +                 RUSER(1), RWORK(LRWORK), TRANS(6,1),
     +                 WEIGHT(NVIMX)
      INTEGER          CONN1(3,2*NVMAX+5), CONN2(3,2*NVMAX+5),
     +                 CONN3(3,2*NVMAX+5), CONN4(3,2*NVMAX+5),
     +                 CONN5(3,2*NVMAX+5), EDGE1(3,NEDMX),
     +                 EDGE2(3,NEDMX), EDGE3(3,NEDMX), EDGE4(3,NEDMX),
     +                 EDGE5(3,NEDMX), ITYPE(1), IUSER(1),
     +                 IWORK(LIWORK), LCOMP(NLINESX), LINE(4,NLINESX),
     +                 NLCOMP(NCOMPX), NUMFIX(1), REFT1(2*NVMAX+5),
     +                 REFT2(2*NVMAX+5), REFT3(2*NVMAX+5),
     +                 REFT4(2*NVMAX+5), REFT5(2*NVMAX+5)
*     .. External Functions ..
      DOUBLE PRECISION FBND
      EXTERNAL         FBND
*     .. External Subroutines ..
      EXTERNAL         D06ABF, D06ACF, D06BAF, D06CAF, D06DAF, D06DBF
*     .. Intrinsic Functions ..
      INTRINSIC        ABS
*     .. Executable Statements ..
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Example 2'
      WRITE (NOUT,*)
*
*     Skip heading in data file
*
      READ (NIN,*)
*
*     Build the mesh of the 1st domain
*
*     Initialise boundary mesh inputs:
*     the number of line and of the characteristic points of
*     the boundary mesh
*
*
      READ (NIN,*) NLINES
*
*     Characteristic points of the boundary geometry
*
      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
*
      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,NVB1,COOR1,NEDGE1,EDGE1,ITRACE,
```

```
     +             RUSER,IUSER,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
*     mesh it using Delaunay-Voronoi method
*
*     Initialise mesh control parameters
*
      ITRACE = 0
      NPROPA = 1
      NVINT = 0
      IFAIL = 0
*
*     Call to the 2D Delaunay-Voronoi mesh generator
*
      CALL D06ABF(NVB1,NVINT,NVMAX,NEDGE1,EDGE1,NV1,NELT1,COOR1,CONN1,
     +            WEIGHT,NPROPA,ITRACE,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
      DO 40 K = 1, NELT1
         REFT1(K) = 1
   40 CONTINUE
*
*     Call the smoothing routine
*
      NVFIX = 0
      NQINT = 10
      IFAIL = 0
*
      CALL D06CAF(NV1,NELT1,NEDGE1,COOR1,EDGE1,CONN1,NVFIX,NUMFIX,
     +            ITRACE,NQINT,IWORK,LIWORK,RWORK,LRWORK,IFAIL)
*
*     build the mesh of the 2nd domain
*
*     Initialise boundary mesh inputs:
*     the number of line and of the characteristic points of
*     the boundary mesh
*
      READ (NIN,*) NLINES
*
*     Characteristic points of the boundary geometry
*
      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 60 I = 1, NCOMP
         READ (NIN,*) NLCOMP(I)
*
         READ (NIN,*) (LCOMP(K),K=J,J+ABS(NLCOMP(I))-1)
         J = J + ABS(NLCOMP(I))
   60 CONTINUE
*
      READ (NIN,*) PMESH
*
      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,NVB2,COOR2,NEDGE2,EDGE2,ITRACE,
     +            RUSER,IUSER,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
*     mesh it using the advancing front method
```

```
*
*     Initialise mesh control parameters
*
      ITRACE = 0
      NVINT = 0
      IFAIL = 0
*
*     Call to the 2D Advancing front mesh generator
*
      CALL D06ACF(NVB2,NVINT,NVMAX,NEDGE2,EDGE2,NV2,NELT2,COOR2,CONN2,
     +            WEIGHT,ITRACE,RWORK,LRWORK,IWORK,LIWORK,IFAIL)
*
      DO 80 K = 1, NELT2
         REFT2(K) = 2
   80 CONTINUE
*
*     Rotation of the 2nd domain mesh to produce
*     the 3rd mesh domain
*
      NTRANS = 1
      ITYPE(1) = 3
      TRANS(1,1) = 6.D0
      TRANS(2,1) = -1.D0
      TRANS(3,1) = -90.D0
      ITRACE = 0
      IFAIL = 0
*
      CALL D06DAF(NV2,NEDGE2,NELT2,NTRANS,ITYPE,TRANS,COOR2,EDGE2,CONN2,
     +            COOR3,EDGE3,CONN3,ITRACE,RWORK,LRWORK,IFAIL)
*
      NV3 = NV2
      NELT3 = NELT2
      NEDGE3 = NEDGE2
*
      DO 100 K = 1, NELT3
         REFT3(K) = 3
  100 CONTINUE
*
*     restitching of the mesh 1 and 2 to form the mesh 4
*
      EPS = 1.D-3
      ITRACE = 0
      IFAIL = 0
*
      CALL D06DBF(EPS,NV1,NELT1,NEDGE1,COOR1,EDGE1,CONN1,REFT1,NV2,
     +            NELT2,NEDGE2,COOR2,EDGE2,CONN2,REFT2,NV4,NELT4,NEDGE4,
     +            COOR4,EDGE4,CONN4,REFT4,ITRACE,IWORK,LIWORK,IFAIL)
*
*     restitching of the mesh 3 and 4 to form the mesh 5
*
      ITRACE = 0
      IFAIL = 0
*
      CALL D06DBF(EPS,NV4,NELT4,NEDGE4,COOR4,EDGE4,CONN4,REFT4,NV3,
     +            NELT3,NEDGE3,COOR3,EDGE3,CONN3,REFT3,NV5,NELT5,NEDGE5,
     +            COOR5,EDGE5,CONN5,REFT5,ITRACE,IWORK,LIWORK,IFAIL)
*
      IF (PMESH.EQ.'N') THEN
         WRITE (NOUT,*) 'The restitched mesh characteristics'
         WRITE (NOUT,99999) 'NV    =', NV5
         WRITE (NOUT,99999) 'NELT  =', NELT5
         WRITE (NOUT,99999) 'NEDGE =', NEDGE5
      ELSE IF (PMESH.EQ.'Y') THEN
*
*     Output the mesh to view it using the NAG Graphics Library
*
         WRITE (NOUT,99998) NV5, NELT5, NEDGE5
         DO 120 I = 1, NV5
            WRITE (NOUT,99997) COOR5(1,I), COOR5(2,I)
  120    CONTINUE
*
```

```
         DO 140 K = 1, NELT5
            WRITE (NOUT,99996) CONN5(1,K), CONN5(2,K), CONN5(3,K),
     +        REFT5(K)
  140    CONTINUE
*
         DO 160 K = 1, NEDGE5
            WRITE (NOUT,99998) EDGE5(1,K), EDGE5(2,K), EDGE5(3,K)
  160    CONTINUE
      ELSE
         WRITE (NOUT,*) 'Problem with the printing option Y or N'
         STOP
      END IF
*
99999 FORMAT (1X,A,I6)
99998 FORMAT (1X,3I10)
99997 FORMAT (2(2X,E12.6))
99996 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, Y0
*     .. Executable Statements ..
      FBND = 0.D0
      IF (I.EQ.1) THEN
*
*     inner circle
*
         X0 = 0.D0
         Y0 = 0.D0
         RADIUS2 = 1.D0
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      ELSE IF (I.EQ.2) THEN
*
*     outer circle
*
         X0 = 0.D0
         Y0 = 0.D0
         RADIUS2 = 5.D0
         FBND = (X-X0)**2 + (Y-Y0)**2 - RADIUS2
      END IF
*
      RETURN
      END
```

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.

```
D06DBF Example Program Data
Example 1
       400       722     :NV1 NELT1
  0.000000E+00  0.000000E+00
          .
          .
          .
  0.100000E+01  0.100000E+01     :COOR1(1:2,1:NV1)
         1         2        22         0
          .
          .
          .
       379       400       399         0 :(CONN1(:,K), REFT, K = 1,...,NELT1)
```

```
'N'                                        : Printing option 'Y' or 'N'
Example 2
 9                                  :1st geometry NLINES (m)
  2.0000   2.0000   1.0000
 -1.0000  -2.2361   0.0000
  0.0000   0.0000   0.0000                  :(COORCH(1,1:m))
 -1.0000   1.0000   0.0000
  0.0000   0.0000  -2.2361
 -1.0000   1.0000   2.2361                  :(COORCH(2,1:m))
10  1  2  0   1.0000 10  2  9  2   1.0000
10  9  5  2   1.0000 10  5  6  2   1.0000
10  6  1  2   1.0000 10  3  8  1   1.0000
10  8  4  1   1.0000 10  4  7  1   1.0000
10  7  3  1   1.0000           :(LINE(:,j),RATE(j),j=1,m)
 2                          :NCOMP (n, number of contours)
 5                           :number of lines in contour 1
 1 2 3 4 5                             :lines of contour 1
-4                           :number of lines in contour 2
 9 8 7 6                               :lines of contour 2
 4                                  :2nd geometry NLINES (m)
  2.0000   6.0000   6.0000   2.0000       :(COORCH(1,1:m))
 -1.0000  -1.0000   1.0000   1.0000       :(COORCH(2,1:m))
19  1  2  0   1.0000 10  2  3  0   1.0000
19  3  4  0   1.0000 10  4  1  0   1.0000 :(LINE(:,j),RATE(j),j=1,m)
 1                          :NCOMP (n, number of contours)
 4                           :number of lines in contour 1
 1 2 3 4                               :lines of contour 1
'N'                            :Printing option 'Y' or 'N'
```

9.3 Program Results

```
 D06DBF Example Program Results

 Example 1

 The restitched mesh characteristics
 in the overlapping case
 NV    =   785
 NELT  =  1428
 NEDGE =   152
 in the partition case
 NV    =   780
 NELT  =  1444
 NEDGE =   133

 Example 2

 The restitched mesh characteristics
 NV    =   643
 NELT  =  1133
 NEDGE =   171
```

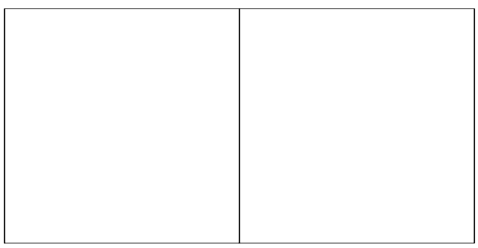

Figure 1
The boundary and the interior interfaces of the two partitioned squares geometry

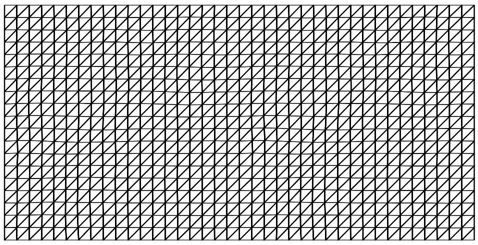

Figure 2
The interior mesh of the two partitioned squares geometry

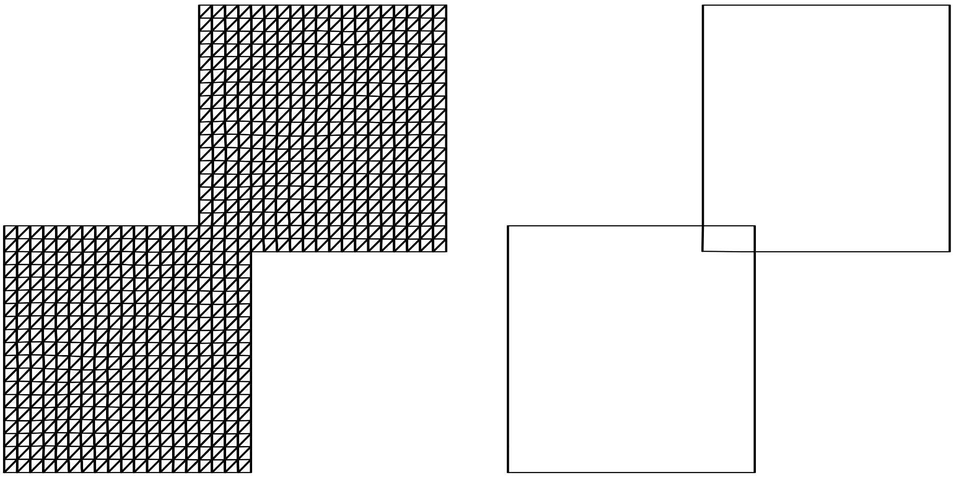

Figure 3
The boundary and the interior interfaces (left); the interior mesh (right)
of the two overlapping squares geometry

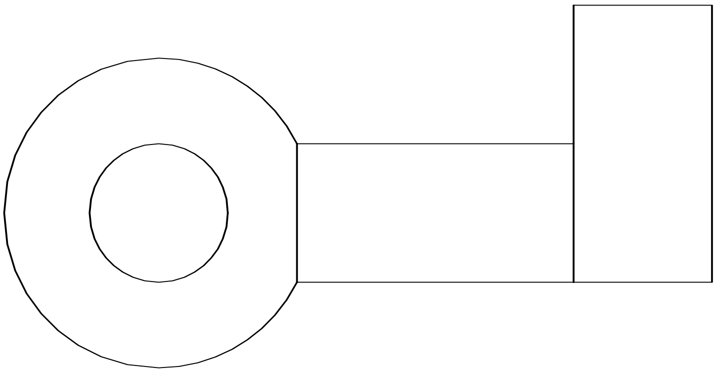

Figure 4
The boundary and the interior interfaces of the double restitched geometry

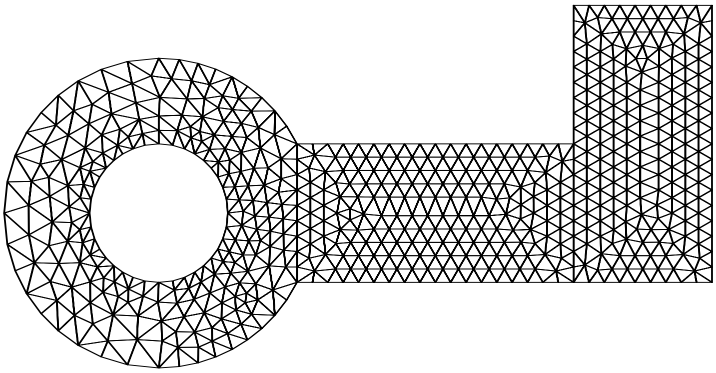

Figure 5
The interior mesh of the double restitched geometry