

NAG Fortran Library Routine Document

D06CCF

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

D06CCF rennumbers the vertices of a given mesh using a Gibbs method, in order the reduce the bandwidth of Finite Element matrices associated with that mesh.

2 Specification

```

SUBROUTINE D06CCF(NV, NELT, NEDGE, NNZMAX, NNZ, COOR, EDGE, CONN, IROW,
1          ICOL, ITRACE, IWORK, LIWORK, RWORK, LRWORK, IFAIL)
  INTEGER          NV, NELT, NEDGE, NNZMAX, NNZ, EDGE(3,NEDGE),
1          CONN(3,NELT), IROW(NNZMAX), ICOL(NNZMAX), ITRACE,
2          IWORK(LIWORK), LIWORK, LRWORK, IFAIL
  real            COOR(2,NV), RWORK(LRWORK)

```

3 Description

D06CCF uses a Gibbs method to renumber the vertices of a given mesh in order to reduce the bandwidth of the associated Finite Element matrix A . This matrix has elements a_{ij} such that:

$$a_{ij} \neq 0 \implies i \text{ and } j \text{ are vertices belonging to the same triangle.}$$

This routine reduces the bandwidth m , which is the smallest integer such that $a_{ij} \neq 0$ whenever $|i - j| > m$ (see Gibbs *et al.* (1976) for details about that method). D06CCF also returns the sparsity structure of the matrix associated with the renumbered mesh.

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

4 References

Gibbs N E, Poole W G Jr and Stockmeyer P K (1976) An algorithm for reducing the bandwidth and profile of a sparse matrix *SIAM J. Numer. Anal.* **13** 236–250

5 Parameters

- | | | |
|----|--|--------------|
| 1: | NV – INTEGER | <i>Input</i> |
| | <i>On entry:</i> the total number of vertices in the input mesh. | |
| | <i>Constraint:</i> NV ≥ 3. | |
| 2: | NELT – INTEGER | <i>Input</i> |
| | <i>On entry:</i> the number of triangles in the input mesh. | |
| | <i>Constraint:</i> NELT ≤ 2 × NV – 1. | |
| 3: | NEDGE – INTEGER | <i>Input</i> |
| | <i>On entry:</i> the number of the boundary edges in the input mesh. | |
| | <i>Constraint:</i> NEDGE ≥ 1. | |

- 4: NNZMAX – INTEGER *Input*
On entry: the maximum number of non-zero entries in the matrix based on the input mesh. It is the dimension of the arrays IROW and ICOL as declared in the (sub)program from which D06CCF is called.
Constraint: $4 \times \text{NELT} + \text{NV} \leq \text{NNZMAX} \leq \text{NV}^2$.
- 5: NNZ – INTEGER *Output*
On exit: the number of non-zero entries in the matrix based on the input mesh.
- 6: COOR(2,NV) – *real* array *Input/Output*
On entry: COOR(1,*i*) contains the *x*-coordinate of the *i*th input mesh vertex, for $i = 1, \dots, \text{NV}$; while COOR(2,*i*) contains the corresponding *y*-coordinate.
On exit: COOR(1,*i*) will contain the *x*-coordinate of the *i*th renumbered mesh vertex, for $i = 1, \dots, \text{NV}$; while COOR(2,*i*) will contain the corresponding *y*-coordinate.
- 7: EDGE(3,NEDGE) – INTEGER array *Input/Output*
On entry: the specification of the boundary or interface edges. EDGE(1 : 2, *j*) contains the vertex number of the two end-points of the *j*th boundary edge. EDGE(3, *j*) is a user-supplied tag for the *j*th boundary or interface edge: EDGE(3, *j*) = 0 for an interior edge and has a non-zero tag otherwise.
On exit: the renumbered specification of the boundary or interface edges.
Constraint: $1 \leq \text{EDGE}(i, j) \leq \text{NV}$ and $\text{EDGE}(1, j) \neq \text{EDGE}(2, j)$, for $i = 1, 2$ and $j = 1, \dots, \text{NEDGE}$.
- 8: CONN(3,NELT) – INTEGER array *Input/Output*
On entry: the connectivity of the mesh between triangles and vertices. For each triangle *j*, CONN(*i*, *j*) gives the indices in COOR of its three vertices (in anticlockwise order), for $i = 1, 2, 3$ and $j = 1, \dots, \text{NELT}$.
On exit: the renumbered connectivity of the mesh between triangles and vertices.
Constraints:
 $1 \leq \text{CONN}(i, j) \leq \text{NV}$,
 $\text{CONN}(1, j) \neq \text{CONN}(2, j)$,
 $\text{CONN}(1, j) \neq \text{CONN}(3, j)$ and $\text{CONN}(2, j) \neq \text{CONN}(3, j)$, for $i = 1, 2, 3$ and $j = 1, \dots, \text{NELT}$
- 9: IROW(NNZMAX) – INTEGER array *Output*
10: ICOL(NNZMAX) – INTEGER array *Output*
On exit: the first NNZ elements contain the row and column indices of the non-zero elements supplied in the Finite Element matrix *A*.
- 11: ITRACE – INTEGER *Input*
On entry: the level of trace information required from D06CCF as follows:
if ITRACE ≤ 0 , no output is generated;
if ITRACE = 1, then information about the effect of the renumbering on the Finite Element matrix are output. This information includes the half bandwidth and the sparsity structure of this matrix before and after renumbering;
if ITRACE > 1, then the output is similar to that produced when ITRACE = 1 but the sparsities (for each row of the matrix, indices of non-zero entries) of the matrix before and after renumbering are also output.

- 12: IWORK(LIWORK) – INTEGER array Workspace
 13: LIWORK – INTEGER Input
- On entry:* the dimension of the array IWORK as declared in the (sub)program from which D06CCF is called.
- Constraint:* $LIWORK \geq \max(NNZMAX, 20 \times NV)$.
- 14: RWORK(LRWORK) – *real* array Workspace
 15: LRWORK – INTEGER Input
- On entry:* the dimension of the array RWORK as declared in the (sub)program from which D06CCF is called.
- Constraint:* $LRWORK \geq NV$.
- 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, NV < 3,
 or NELT > 2 × NV - 1,
 or NEDGE < 1,
 or NNZMAX < 1 or NNZMAX > NV²
 or CONN(*i*, *j*) < 1 or CONN(*i*, *j*) > NV for some *i* = 1, 2, 3 and *j* = 1, ..., NELT,
 or CONN(1, *j*) = CONN(2, *j*) or CONN(1, *j*) = CONN(3, *j*) or
 CONN(2, *j*) = CONN(3, *j*) for some *j* = 1, ..., NELT,
 or EDGE(*i*, *j*) < 1 or EDGE(*i*, *j*) > NV for some *i* = 1, 2 and *j* = 1, ..., NEDGE,
 or EDGE(1, *j*) = EDGE(2, *j*) for some *j* = 1, ..., NEDGE,
 or LIWORK < max(NNZMAX, 20 × NV),
 or LRWORK < NV.

IFAIL = 2

A serious error has occurred during the computation of the compact sparsity of the Finite Element matrix or in an internal call to the renumbering routine. Check the input mesh, especially the connectivity between triangles and vertices (the argument CONN). If the problem persists, contact NAG.

7 Accuracy

Not applicable.

8 Further Comments

Not applicable.

9 Example

In this example, a geometry with two holes (two interior circles inside an exterior one) is considered. The geometry has been meshed using the simple incremental method (D06AAF) and it has 250 vertices and 402 triangles (see Figure 1). The routine D06BAF is used to renumber the vertices, and one can see the benefit in terms of the sparsity of the Finite Element matrix based on the renumbered mesh (see Figure 2).

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.

```
*      D06CCF Example Program Text
*      Mark 20 Release. NAG Copyright 2001.
*      .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NBEDMX, NVMAX, NNZMAX, LRWORK, LIWORK
      PARAMETER        (NBEDMX=100,NVMAX=300,NNZMAX=10*NVMAX,
+                      LRWORK=NVMAX,LIWORK=20*NVMAX)
*      .. Local Scalars ..
      INTEGER          I, I1, IFAIL, ITRACE, K, NEDGE, NELT, NNZ, NV,
+                      REFTK
      CHARACTER        PMESH
*      .. Local Arrays ..
      real             COOR(2,NVMAX), RWORK(LRWORK)
      INTEGER          CONN(3,2*NVMAX+5), EDGE(3,NBEDMX), ICOL(NNZMAX),
+                      IROW(NNZMAX), IWORK(LIWORK)
*      .. External Subroutines ..
      EXTERNAL         D06CBF, D06CCF
*      .. Executable Statements ..
*
      WRITE (NOUT,*) 'D06CCF Example Program Results'
      WRITE (NOUT,*)
*
*      Skip heading in data file
*
      READ (NIN,*)
*
*      Reading of the geometry
*
      READ (NIN,*) NV, NELT, NEDGE
*
      IF (NV.GT.NVMAX .OR. NEDGE.GT.NBEDMX) THEN
        WRITE (NOUT,*) 'Problem with the array dimensions '
        WRITE (NOUT,99999) 'NV MAX ', NV, NVMAX
        WRITE (NOUT,99999) 'NEDGE MAX ', NEDGE, NBEDMX
        STOP
      END IF
*
      DO 20 I = 1, NV
        READ (NIN,*) COOR(1,I), COOR(2,I)
20  CONTINUE
*
      DO 40 K = 1, NELT
        READ (NIN,*) CONN(1,K), CONN(2,K), CONN(3,K), REFTK
40  CONTINUE
*
      DO 60 I = 1, NEDGE
        READ (NIN,*) I1, EDGE(1,I), EDGE(2,I), EDGE(3,I)
60  CONTINUE
*
      READ (NIN,*) PMESH
*
*      Compute the sparsity of the FE matrix
*      from the input geometry
*
      IFAIL = 0
      CALL D06CBF(NV,NELT,NNZMAX,CONN,NNZ,IROW,ICOL,IFAIL)
```

```

*
      IF (PMESH.EQ.'N') THEN
        WRITE (NOUT,*) 'The Matrix Sparsity characteristics'
        WRITE (NOUT,*) 'before the renumbering'
        WRITE (NOUT,99998) 'NV =', NV
        WRITE (NOUT,99998) 'NNZ =', NNZ
      ELSE IF (PMESH.EQ.'Y') THEN
*
*      Output the sparsity of the mesh to view it
*      using the NAG Graphics Library
*
        WRITE (NOUT,99997) NV, NNZ
        DO 80 I = 1, NNZ
          WRITE (NOUT,99997) IROW(I), ICOL(I)
80      CONTINUE
      ELSE
        WRITE (NOUT,*) 'Problem with the printing option Y or N'
        STOP
      END IF
*
*      Call the renumbering routine and get the new sparsity
*
      IFAIL = 0
      ITRACE = 1
      CALL D06CCF(NV,NELT,NEDGE,NNZMAX,NNZ,COOR,EDGE,CONN,IROW,ICOL,
+               ITRACE,IWORK,LIWORK,RWORK,LRWORK,IFAIL)
*
      IF (PMESH.EQ.'N') THEN
        WRITE (NOUT,*)
        WRITE (NOUT,*) 'The Matrix Sparsity characteristics'
        WRITE (NOUT,*) 'after the renumbering'
        WRITE (NOUT,99998) 'NV =', NV
        WRITE (NOUT,99998) 'NNZ =', NNZ
        WRITE (NOUT,99998) 'NELT =', NELT
      ELSE IF (PMESH.EQ.'Y') THEN
*
*      Output the sparsity of the renumbered mesh to view it
*      using the NAG Graphics Library
*
        WRITE (NOUT,99997) NV, NNZ
        DO 100 I = 1, NNZ
          WRITE (NOUT,99997) IROW(I), ICOL(I)
100      CONTINUE
*
*      Output the renumbered mesh to view it
*      using the NAG Graphics Library
*
        WRITE (NOUT,99997) NV, NELT
        DO 120 I = 1, NV
          WRITE (NOUT,99996) COOR(1,I), COOR(2,I)
120      CONTINUE
*
        REFTK = 0
        DO 140 K = 1, NELT
          WRITE (NOUT,99995) CONN(1,K), CONN(2,K), CONN(3,K), REFTK
140      CONTINUE
        END IF
*
      STOP
*
99999 FORMAT (1X,A,2I6)
99998 FORMAT (1X,A,I6)
99997 FORMAT (1X,2I10)
99996 FORMAT (2(2X,E12.6))
99995 FORMAT (1X,4I10)
      END

```

9.2 Program Data

Note: 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.

```
D06CCF Example Program Data
      250      402      100      :NV NELT NEDGE
      0.100000E+01  0.000000E+00
      .
      .
      .
      0.112781E+00  0.103479E+00      :COOR(1:2,1:NV)
      21      55      56      1
      .
      .
      .
      1      151      250      155      1      : (CONN(:,K), REFT, K=1,...,NELT)
      1      1      2      1
      .
      .
      .
      100 100 71      1      : (I1, EDGE(:,I), I=1,NEDGE)
      'N'      :Printing option 'Y' or 'N'
```

9.3 Program Results

D06CCF Example Program Results

The Matrix Sparsity characteristics
before the renumbering

NV = 250
NNZ = 1556

INITIAL HALF-BAND-WIDTH : 234 INITIAL PROFILE : 18233
FINAL HALF-BAND-WIDTH : 28 FINAL PROFILE : 4038

The Matrix Sparsity characteristics
after the renumbering

NV = 250
NNZ = 1556
NELT = 402

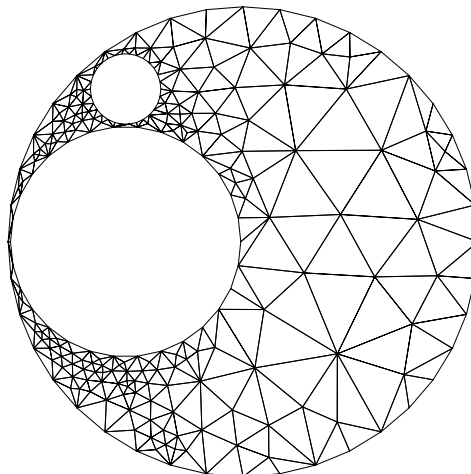


Figure 1
Mesh of the geometry



Figure 2
Sparsity of the matrix before (top) and after (bottom) the renumbering
