NAG Library Function Document

nag_mesh2d_renum (d06ccc)

1 Purpose

nag_mesh2d_renum (d06ccc) renumbers 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

```
#include <nag.h>
#include <nagd06.h>
void nag_mesh2d_renum (Integer nv, Integer nelt, Integer nedge,
     Integer nnzmax, Integer *nnz, double coor[], Integer edge[],
     Integer conn[], Integer irow[], Integer icol[], Integer itrace,
     const char *outfile, NagError *fail)
```

3 Description

nag_mesh2d_renum (d06ccc) 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:

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

This function 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). nag_mesh2d_renum (d06ccc) also returns the sparsity structure of the matrix associated with the renumbered mesh.

This function 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 Arguments

1: **nv** – Integer *Input*

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

Constraint: **nv** ≥ 3.

2: **nelt** – Integer *Input*

On entry: the number of triangles in the input mesh.

Constraint: **nelt** $\leq 2 \times$ **nv** $- 1$.

3: **nedge** – Integer *Input*

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

Constraint: **nedge** ≥ 1.

4: **nnzmax** – Integer *Input*

On entry: the maximum number of nonzero entries in the matrix based on the input mesh. It is the dimension of the arrays **irow** and **icol** as declared in the function from which nag_mesh2d_renum (d06ccc) is called.

Constraint: $4 \times \mathbf{nelt} + \mathbf{nv} \leq \mathbf{nnzmax} \leq \mathbf{nv}^2$.

5: **nnz** – Integer * *Output*

On exit: the number of nonzero entries in the matrix based on the input mesh.

6: **coor**[$2 \times \mathbf{nv}$] – double *Input/Output*

Note: the (i,j)th element of the matrix is stored in **coor**[$(j-1) \times 2 + i - 1$].

On entry: **coor**[$(i-1) \times 2$] contains the x coordinate of the ith input mesh vertex, for $i = 1, 2, \ldots, \mathbf{nv}$; while **coor**[$(i-1) \times 2 + 1$] contains the corresponding y coordinate.

On exit: **coor**[$(i-1) \times 2$] will contain the x coordinate of the ith renumbered mesh vertex, for $i = 1, 2, \ldots, \mathbf{nv}$; while **coor**[$(i-1) \times 2 + 1$] will contain the corresponding y coordinate.

7: **edge**[$3 \times \mathbf{nedge}$] – Integer *Input/Output*

Note: the (i,j)th element of the matrix is stored in **edge**[$(j-1) \times 3 + i - 1$].

On entry: the specification of the boundary or interface edges. **edge**[$(j-1) \times 3$] and **edge**[$(j-1) \times 3 + 1$] contain the vertex numbers of the two end points of the jth boundary edge. **edge**[$(j-1) \times 3 + 2$] is a user-supplied tag for the jth boundary or interface edge: **edge**[$(j-1) \times 3 + 2$] $= 0$ for an interior edge and has a nonzero tag otherwise. Note that the edge vertices are numbered from 1 to **nv**.

Constraint: $1 \leq \mathbf{edge}[(j-1) \times 3 + i - 1] \leq \mathbf{nv}$ and $\mathbf{edge}[(j-1) \times 3] \neq \mathbf{edge}[(j-1) \times 3 + 1]$, for $i = 1, 2$ and $j = 1, 2, \ldots, \mathbf{nedge}$.

On exit: the renumbered specification of the boundary or interface edges.

8: **conn**[$3 \times \mathbf{nelt}$] – Integer *Input/Output*

Note: the (i,j)th element of the matrix is stored in **conn**[$(j-1) \times 3 + i - 1$].

On entry: the connectivity of the mesh between triangles and vertices. For each triangle j, **conn**[$(j-1) \times 3 + i - 1$] gives the indices of its three vertices (in anticlockwise order), for $i = 1, 2, 3$ and $j = 1, 2, \ldots, \mathbf{nelt}$. Note that the mesh vertices are numbered from 1 to **nv**.

Constraint: $1 \leq \mathbf{conn}[(j-1) \times 3 + i - 1] \leq \mathbf{nv}$ and $\mathbf{conn}[(j-1) \times 3] \neq \mathbf{conn}[(j-1) \times 3 + 1]$ and $\mathbf{conn}[(j-1) \times 3] \neq \mathbf{conn}[(j-1) \times 3 + 2]$ and
$\mathbf{conn}[(j-1) \times 3 + 1] \neq \mathbf{conn}[(j-1) \times 3 + 2]$, for $i = 1, 2, 3$ and $j = 1, 2, \ldots, \mathbf{nelt}$.

On exit: the renumbered connectivity of the mesh between triangles and vertices.

9: **irow**[**nnzmax**] – Integer *Output*

10: **icol**[**nnzmax**] – Integer *Output*

On exit: the first **nnz** elements contain the row and column indices of the nonzero elements supplied in the finite element matrix A.

11: **itrace** – Integer *Input*

On entry: the level of trace information required from nag_mesh2d_renum (d06ccc).

itrace ≤ 0
No output is generated.

itrace $= 1$
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.

itrace > 1
The output is similar to that produced when **itrace** $= 1$ but the sparsities (for each row of the matrix, indices of nonzero entries) of the matrix before and after renumbering are also output.

12: **outfile** – const char * *Input*

On entry: the name of a file to which diagnostic output will be directed. If **outfile** is **NULL** the diagnostic output will be directed to standard output.

13: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_FAIL_SPARSITY

An error has occurred during the computation of the compact sparsity of the finite element matrix. Check the Triangle/Vertices connectivity.

NE_INT

On entry, **nedge** $= \langle value \rangle$.
Constraint: **nedge** ≥ 1.

On entry, **nv** $= \langle value \rangle$.
Constraint: **nv** ≥ 3.

NE_INT_2

On entry, **nelt** $= \langle value \rangle$ and **nv** $= \langle value \rangle$.
Constraint: **nelt** $\leq 2 \times$ **nv** $- 1$.

On entry, the endpoints of the edge J have the same index I: $J = \langle value \rangle$ and $I = \langle value \rangle$.

On entry, vertices 1 and 2 of the triangle K have the same index I: $K = \langle value \rangle$ and $I = \langle value \rangle$.

On entry, vertices 1 and 3 of the triangle K have the same index I: $K = \langle value \rangle$ and $I = \langle value \rangle$.

On entry, vertices 2 and 3 of the triangle K have the same index I: $K = \langle value \rangle$ and $I = \langle value \rangle$.

NE_INT_3

On entry, **nnzmax** $= \langle value \rangle$, **nelt** $= \langle value \rangle$ and **nv** $= \langle value \rangle$.
Constraint: **nnzmax** $\geq (4 \times$ **nelt** $+$ **nv**$)$ and **nnzmax** $\leq$ **nv**^{2}.

NE_INT_4

On entry, $\mathbf{CONN}(I, J) = \langle value \rangle$, $I = \langle value \rangle$, $J = \langle value \rangle$ and $\mathbf{nv} = \langle value \rangle$.
Constraint: $\mathbf{CONN}(I, J) \geq 1$ and $\mathbf{CONN}(I, J) \leq \mathbf{nv}$, where $\mathbf{CONN}(I, J)$ denotes $\mathbf{conn}[(J-1) \times 3 + I - 1]$.

On entry, $\mathbf{EDGE}(I, J) = \langle value \rangle$, $I = \langle value \rangle$, $J = \langle value \rangle$ and $\mathbf{nv} = \langle value \rangle$.
Constraint: $\mathbf{EDGE}(I, J) \geq 1$ and $\mathbf{EDGE}(I, J) \leq \mathbf{nv}$, where $\mathbf{EDGE}(I, J)$ denotes $\mathbf{edge}[(J-1) \times 3 + I - 1]$.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

A serious error has occurred in an internal call to the renumbering function. Check the input mesh especially the connectivity. Seek expert help.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

NE_NOT_CLOSE_FILE

Cannot close file $\langle value \rangle$.

NE_NOT_WRITE_FILE

Cannot open file $\langle value \rangle$ for writing.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_mesh2d_renum (d06ccc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

None.

10 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 (nag_mesh2d_inc (d06aac)) and it has 250 vertices and 402 triangles (see Figure 1 in Section 10.3). The function nag_mesh2d_bound (d06bac) 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 and 3 in Section 10.3).

10.1 Program Text

```
/* nag_mesh2d_renum (d06ccc) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 7, 2001.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagd06.h>

#define EDGE(I, J) edge[3*((J) -1)+(I) -1]
#define CONN(I, J) conn[3*((J) -1)+(I) -1]
#define COOR(I, J) coor[2*((J) -1)+(I) -1]

int main(void)
{
  Integer  exit_status, i, itrace, nedge, nelt, nnz, nnzmax, nv, reftk;
  NagError fail;
  char     pmesh[2];
  double   *coor = 0;
  Integer  *conn = 0, *edge = 0, *icol = 0, *irow = 0;

  INIT_FAIL(fail);

  exit_status = 0;

  printf(" nag_mesh2d_renum (d06ccc) Example Program Results\n\n");
  fflush(stdout);

  /* Skip heading in data file */

#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  /* Reading of the geometry */

#ifdef _WIN32
  scanf_s("%"NAG_IFMT"", &nv);
#else
  scanf("%"NAG_IFMT"", &nv);
#endif
#ifdef _WIN32
  scanf_s("%"NAG_IFMT"", &nelt);
#else
  scanf("%"NAG_IFMT"", &nelt);
#endif
#ifdef _WIN32
  scanf_s("%"NAG_IFMT"", &nedge);
#else
  scanf("%"NAG_IFMT"", &nedge);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  nnzmax = 10*nv;

  /* Allocate memory */

  if (!(coor = NAG_ALLOC(2*nv, double)) ||
      !(conn = NAG_ALLOC(3*nelt, Integer)) ||
      !(edge = NAG_ALLOC(3*nedge, Integer)) ||
```

```
      !(irow = NAG_ALLOC(nnzmax, Integer)) ||
      !(icol = NAG_ALLOC(nnzmax, Integer)))
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

  for (i = 1; i <= nv; ++i)
    {
#ifdef _WIN32
      scanf_s("%lf", &COOR(1, i));
#else
      scanf("%lf", &COOR(1, i));
#endif
#ifdef _WIN32
      scanf_s("%lf", &COOR(2, i));
#else
      scanf("%lf", &COOR(2, i));
#endif
#ifdef _WIN32
      scanf_s("%*[^\n] ");
#else
      scanf("%*[^\n] ");
#endif
    }

  for (i = 1; i <= nelt; ++i)
    {
#ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &CONN(1, i));
#else
      scanf("%"NAG_IFMT"", &CONN(1, i));
#endif
#ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &CONN(2, i));
#else
      scanf("%"NAG_IFMT"", &CONN(2, i));
#endif
#ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &CONN(3, i));
#else
      scanf("%"NAG_IFMT"", &CONN(3, i));
#endif
#ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &reftk);
#else
      scanf("%"NAG_IFMT"", &reftk);
#endif
#ifdef _WIN32
      scanf_s("%*[^\n] ");
#else
      scanf("%*[^\n] ");
#endif
    }

  for (i = 1; i <= nedge; ++i)
    {
#ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &reftk);
#else
      scanf("%"NAG_IFMT"", &reftk);
#endif
#ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &EDGE(1, i));
#else
      scanf("%"NAG_IFMT"", &EDGE(1, i));
#endif
#ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &EDGE(2, i));
#else
```

```
      scanf("%"NAG_IFMT"", &EDGE(2, i));
#endif
#ifdef _WIN32
      scanf_s("%"NAG_IFMT"", &EDGE(3, i));
#else
      scanf("%"NAG_IFMT"", &EDGE(3, i));
#endif
#ifdef _WIN32
      scanf_s("%*[^\n] ");
#else
      scanf("%*[^\n] ");
#endif
    }

#ifdef _WIN32
  scanf_s(" ' %1s '", pmesh, _countof(pmesh));
#else
  scanf(" ' %1s '", pmesh);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  /* Compute the sparsity of the FE matrix */
  /* from the input geometry */

  /* nag_mesh2d_sparse (d06cbc).
   * Generates a sparsity pattern of a Finite Element matrix
   * associated with a given mesh
   */
  nag_mesh2d_sparse(nv, nelt, nnzmax, conn, &nnz, irow, icol, &fail);

  if (fail.code == NE_NOERROR)
    {
      if (pmesh[0] == 'N')
        {
          printf(" The Matrix Sparsity characteristics\n");
          printf(" before the renumbering\n");
          printf(" nv  =%6"NAG_IFMT"\n", nv);
          printf(" nnz =%6"NAG_IFMT"\n", nnz);
        }
      else if (pmesh[0] == 'Y')
        {
          /* Output the sparsity of the mesh to view */
          /* it using the NAG Graphics Library */

          printf(" %10"NAG_IFMT"%10"NAG_IFMT"\n", nv, nnz);

          for (i = 0; i < nnz; ++i)
            printf(" %10"NAG_IFMT"%10"NAG_IFMT"\n", irow[i], icol[i]);
        }
      else
        {
          printf("Problem with the printing option Y or N\n");
          exit_status = -1;
          goto END;
        }
    }
  else
    {
      printf("Error from nag_mesh2d_sparse (d06cbc).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }

  /* Call the renumbering routine and get the new sparsity */

  itrace = 1;
```

```
  /* nag_mesh2d_renum (d06ccc).
   * Renumbers a given mesh using Gibbs method
   */
  fflush(stdout);
  nag_mesh2d_renum(nv, nelt, nedge, nnzmax, &nnz, coor, edge, conn, irow, icol,
                   itrace, 0, &fail);
  if (fail.code == NE_NOERROR)
    {
      if (pmesh[0] == 'N')
        {
          printf("\n The Matrix Sparsity characteristics\n");
          printf(" after the renumbering\n");
          printf(" nv   =%6"NAG_IFMT"\n", nv);
          printf(" nnz  =%6"NAG_IFMT"\n", nnz);
          printf(" nelt =%6"NAG_IFMT"\n", nelt);
        }
      else if (pmesh[0] == 'Y')
        {
          /* Output the sparsity of the renumbered mesh */
          /* to view it using the NAG Graphics Library */

          printf("%10"NAG_IFMT"%10"NAG_IFMT"\n", nv, nnz);

          for (i = 0; i < nnz; ++i)
            printf(" %10"NAG_IFMT"%10"NAG_IFMT"\n", irow[i], icol[i]);

          /* Output the renumbered mesh to view */
          /* it using the NAG Graphics Library */

          printf(" %10"NAG_IFMT"%10"NAG_IFMT"\n", nv, nelt);

          for (i = 1; i <= nv; ++i)
            printf("  %15.6e  %15.6e  \n",
                    COOR(1, i), COOR(2, i));

          reftk = 0;
          for (i = 1; i <= nelt; ++i)
            printf(" %10"NAG_IFMT"%10"NAG_IFMT"%10"NAG_IFMT"%10"NAG_IFMT"\n",
                    CONN(1, i), CONN(2, i), CONN(3, i), reftk);
        }
    }
  else
    {
      printf("Error from nag_mesh2d_renum (d06ccc).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }

 END:
  NAG_FREE(coor);
  NAG_FREE(conn);
  NAG_FREE(edge);
  NAG_FREE(irow);
  NAG_FREE(icol);

  return exit_status;
}
```

10.2 Program Data

```
nag_mesh2d_renum (d06ccc) Example Program Data
       250       402       100       :NV NELT NEDGE
  0.100000E+01  0.000000E+00
  0.987688E+00  0.156434E+00
  0.951057E+00  0.309017E+00
  0.891007E+00  0.453990E+00
  0.809017E+00  0.587785E+00
  0.707107E+00  0.707107E+00
```

```
 0.587785E+00  0.809017E+00
 0.453990E+00  0.891007E+00
 0.309017E+00  0.951057E+00
 0.156434E+00  0.987688E+00
 0.612303E-16  0.100000E+01
 -.156434E+00  0.987688E+00
 -.309017E+00  0.951057E+00
 -.453990E+00  0.891007E+00
 -.587785E+00  0.809017E+00
 -.707107E+00  0.707107E+00
 -.809017E+00  0.587785E+00
 -.891007E+00  0.453990E+00
 -.951057E+00  0.309017E+00
 -.987688E+00  0.156434E+00
 -.100000E+01  0.122461E-15
 -.987688E+00  -.156434E+00
 -.951057E+00  -.309017E+00
 -.891007E+00  -.453990E+00
 -.809017E+00  -.587785E+00
 -.707107E+00  -.707107E+00
 -.587785E+00  -.809017E+00
 -.453990E+00  -.891007E+00
 -.309017E+00  -.951057E+00
 -.156434E+00  -.987688E+00
 -.183691E-15  -.100000E+01
 0.156434E+00  -.987688E+00
 0.309017E+00  -.951057E+00
 0.453990E+00  -.891007E+00
 0.587785E+00  -.809017E+00
 0.707107E+00  -.707107E+00
 0.809017E+00  -.587785E+00
 0.891007E+00  -.453990E+00
 0.951057E+00  -.309017E+00
 0.987688E+00  -.156434E+00
 -.100000E-01  0.000000E+00
 -.207077E-01  -.101877E+00
 -.523627E-01  -.199301E+00
 -.103582E+00  -.288015E+00
 -.172126E+00  -.364141E+00
 -.255000E+00  -.424352E+00
 -.348582E+00  -.466018E+00
 -.448781E+00  -.487316E+00
 -.551219E+00  -.487316E+00
 -.651418E+00  -.466018E+00
 -.745000E+00  -.424352E+00
 -.827874E+00  -.364141E+00
 -.896418E+00  -.288015E+00
 -.947637E+00  -.199301E+00
 -.979292E+00  -.101877E+00
 -.990000E+00  -.600057E-16
 -.979292E+00  0.101877E+00
 -.947637E+00  0.199301E+00
 -.896418E+00  0.288015E+00
 -.827874E+00  0.364141E+00
 -.745000E+00  0.424352E+00
 -.651418E+00  0.466018E+00
 -.551219E+00  0.487316E+00
 -.448781E+00  0.487316E+00
 -.348582E+00  0.466018E+00
 -.255000E+00  0.424352E+00
 -.172126E+00  0.364141E+00
 -.103582E+00  0.288015E+00
 -.523627E-01  0.199301E+00
 -.207077E-01  0.101877E+00
 -.350000E+00  0.650000E+00
 -.353278E+00  0.618813E+00
 -.362968E+00  0.588990E+00
 -.378647E+00  0.561832E+00
 -.399630E+00  0.538528E+00
 -.425000E+00  0.520096E+00
 -.453647E+00  0.507342E+00
```

```
 -.484321E+00  0.500822E+00
 -.515679E+00  0.500822E+00
 -.546353E+00  0.507342E+00
 -.575000E+00  0.520096E+00
 -.600370E+00  0.538528E+00
 -.621353E+00  0.561832E+00
 -.637032E+00  0.588990E+00
 -.646722E+00  0.618813E+00
 -.650000E+00  0.650000E+00
 -.646722E+00  0.681187E+00
 -.637032E+00  0.711010E+00
 -.621353E+00  0.738168E+00
 -.600370E+00  0.761472E+00
 -.575000E+00  0.779904E+00
 -.546353E+00  0.792658E+00
 -.515679E+00  0.799178E+00
 -.484321E+00  0.799178E+00
 -.453647E+00  0.792658E+00
 -.425000E+00  0.779904E+00
 -.399630E+00  0.761472E+00
 -.378647E+00  0.738168E+00
 -.362968E+00  0.711010E+00
 -.353278E+00  0.681187E+00
 -.956089E+00  -.230370E+00
 -.937770E+00  0.256699E+00
 -.883606E+00  -.345647E+00
 -.887487E+00  0.341109E+00
 -.956121E+00  0.230243E+00
 -.822777E+00  -.412480E+00
 -.817996E+00  0.416854E+00
 -.855587E+00  -.384073E+00
 -.892227E+00  0.373670E+00
 -.773267E+00  -.488008E+00
 -.726484E+00  0.502542E+00
 -.725231E+00  -.575236E+00
 -.602781E+00  0.507127E+00
 -.726703E+00  -.474435E+00
 -.811237E+00  0.458137E+00
 -.642004E+00  -.515653E+00
 -.657296E+00  0.554235E+00
 -.683312E+00  0.585866E+00
 -.684897E+00  0.549035E+00
 -.532743E+00  -.604318E+00
 -.679026E+00  0.643799E+00
 -.629779E+00  -.554290E+00
 -.709261E+00  0.615629E+00
 -.457237E+00  -.642375E+00
 -.405478E+00  0.503111E+00
 -.326542E+00  0.503173E+00
 -.278515E+00  -.827042E+00
 -.394319E+00  -.758942E+00
 -.264390E+00  0.496138E+00
 -.298359E+00  0.472495E+00
 -.320709E+00  0.544904E+00
 -.428352E+00  -.549588E+00
 -.329737E+00  -.863799E+00
 -.321314E+00  -.611999E+00
0.241817E-01  0.516892E+00
 -.494322E+00  -.591405E+00
 -.742355E+00  0.573809E+00
 -.630648E+00  0.745293E+00
 -.596882E+00  -.556800E+00
 -.695894E+00  -.626810E+00
 -.190165E+00  0.636670E+00
 -.735093E+00  0.626685E+00
 -.117140E+00  0.546818E+00
 -.240672E+00  -.500057E+00
 -.527531E+00  -.525526E+00
0.432372E-01  0.269690E+00
 -.683440E+00  0.515823E+00
 -.713830E+00  0.540385E+00
```

```
-.348979E+00  0.766001E+00
0.108745E+00  -.781322E+00
0.224298E+00  0.387771E+00
-.685077E+00  -.503895E+00
-.610711E+00  -.499689E+00
0.400866E+00  -.767548E+00
0.414938E+00  0.115869E+00
0.401302E+00  -.477101E+00
-.392997E+00  -.676467E+00
-.507962E+00  0.819646E+00
0.109821E+00  -.917500E+00
-.393191E+00  0.823936E+00
-.734269E+00  -.624011E+00
-.181713E+00  -.593922E+00
-.678737E+00  -.543121E+00
-.267152E+00  0.734268E+00
-.713192E+00  0.474779E+00
-.120008E+00  -.449576E+00
-.811122E+00  0.512425E+00
-.296046E+00  0.829124E+00
-.822793E+00  -.453084E+00
0.502770E-01  -.536928E+00
-.780735E+00  -.444067E+00
-.160004E+00  0.866397E+00
-.867902E+00  0.368547E+00
0.253845E+00  -.703146E+00
0.310887E-01  0.824703E+00
0.685429E+00  -.565692E+00
0.176656E+00  0.667501E+00
-.792326E+00  0.425863E+00
0.259813E+00  -.132696E+00
-.864353E+00  0.395698E+00
0.337358E+00  0.799966E+00
-.813146E+00  -.503876E+00
0.560499E+00  -.149591E+00
0.455341E+00  0.786195E+00
0.719755E+00  -.349992E+00
0.439775E+00  0.649733E+00
0.826871E+00  -.144892E+00
0.589606E+00  0.389410E+00
-.887266E+00  -.382046E+00
0.895071E-01  -.288937E+00
0.778392E+00  0.470947E+00
0.742697E+00  0.953286E-01
0.859885E+00  0.384343E+00
0.818251E+00  0.301063E+00
0.103511E-01  0.184473E+00
-.937762E+00  -.256864E+00
-.392448E-02  0.384783E+00
-.505369E+00  -.512943E+00
-.842361E+00  0.417785E+00
-.175436E+00  -.801078E+00
-.283950E+00  -.480225E+00
-.217608E+00  0.546482E+00
-.794087E+00  -.411204E+00
-.334275E-01  0.656052E+00
-.758602E+00  0.466834E+00
-.855969E+00  -.417043E+00
-.302687E+00  0.641793E+00
-.769554E+00  -.534987E+00
-.636154E+00  0.523972E+00
-.650081E+00  -.638369E+00
-.581606E+00  -.674515E+00
-.726686E+00  -.519913E+00
-.557572E+00  -.549267E+00
-.763666E+00  0.514416E+00
-.601791E+00  -.595417E+00
-.663690E+00  -.588820E+00
-.781475E+00  0.547136E+00
-.461495E+00  -.591849E+00
-.490838E+00  -.616460E+00
```

```
-.539847E-01  0.312225E+00
0.201811E+00  0.845927E+00
-.452118E-01  -.716454E+00
-.365587E+00  -.545250E+00
-.289703E+00  -.882622E+00
-.462033E+00  -.801846E+00
-.371130E+00  -.884959E+00
-.398175E+00  -.511898E+00
-.404759E+00  -.602550E+00
-.707548E+00  0.648164E+00
-.193724E+00  0.471118E+00
-.509470E+00  -.650706E+00
-.271842E+00  0.583669E+00
-.391386E+00  -.833837E+00
-.483869E+00  -.716019E+00
-.241859E+00  -.889815E+00
-.264020E-01  -.866908E+00
-.297361E+00  -.724815E+00
-.928668E-01  0.456584E+00
-.268381E+00  0.542930E+00
-.494685E+00  -.551610E+00
-.331937E+00  0.590142E+00
-.338743E+00  -.801578E+00
-.290178E+00  -.533561E+00
-.317392E-01  0.250295E+00
-.598252E+00  -.526993E+00
-.123884E+00  0.743799E+00
-.458827E+00  0.825051E+00
-.237766E+00  0.463959E+00
-.194011E+00  -.466058E+00
0.112781E+00  0.103479E+00    :COOR(1:2,1:NV)
      21        55        56         1
      17       217       137         1
      54        55        22         1
     148       119       118         1
      57        21        56         1
      58        20        57         1
      22        55        21         1
     205       165       111         1
      20        21        57         1
     147       119       148         1
      70       250       195         1
      23        53       196         1
     146       151       197         1
      59        19       102         1
     245       213        49         1
      52        53       103         1
     145       240       198         1
     180        60       199         1
     101       196        54         1
     127       224       235         1
      19        20       105         1
     144        46       201         1
     129       230       202         1
      51        52       203         1
     143       135       204         1
      61       165       205         1
      24       169       206         1
     137       142        17         1
      18        19       109         1
     142        16        17         1
      72        73       241         1
     182        25       208         1
     207       100        71         1
      62       113       209         1
     211       215       210         1
     210        26       211         1
     140       112       161         1
      63        81       113         1
     114       110       212         1
     145        49       213         1
```

111	148	214	1
139	122	215	1
90	138	89	1
122	163	216	1
138	16	88	1
118	137	148	1
209	83	117	1
214	137	217	1
113	82	209	1
132	240	218	1
83	84	117	1
136	120	219	1
80	81	63	1
68	244	220	1
117	84	118	1
175	177	221	1
79	80	63	1
200	30	236	1
117	118	119	1
134	243	223	1
78	79	63	1
64	77	78	1
127	133	224	1
27	28	225	1
64	78	63	1
133	127	242	1
86	87	121	1
48	132	227	1
134	223	228	1
129	131	126	1
116	163	122	1
131	241	74	1
121	16	229	1
66	130	65	1
87	16	121	1
130	126	65	1
76	77	64	1
125	75	76	1
88	16	87	1
89	138	88	1
67	197	238	1
219	120	231	1
65	75	125	1
130	129	126	1
131	129	239	1
126	75	65	1
226	133	233	1
90	15	138	1
225	128	234	1
126	131	75	1
30	200	235	1
222	200	236	1
127	200	237	1
230	67	238	1
66	129	130	1
91	15	90	1
75	131	74	1
129	202	239	1
65	125	64	1
125	76	64	1
132	48	240	1
131	232	241	1
127	237	242	1
144	201	243	1
121	123	118	1
68	69	244	1
240	48	198	1
123	142	137	1
15	16	138	1
49	153	245	1
92	15	91	1

```
 93       158        92        1
216       112       140        1
164       141       246        1
 85       121       118        1
 94        95       247        1
123       121       229        1
129        66       248        1
245       122       139        1
166        45       249        1
213       120       240        1
 70        41       250        1
 62       209       147        1
160       247        95        1
137       118       123        1
207       164       100        1
111       147       148        1
160        95        96        1
121        85        86        1
 13        14       160        1
147       117       119        1
 33        34       154        1
118        84        85        1
188         6       186        1
212       110       208        1
149       160        97        1
116       122       245        1
149        97        98        1
178       115       107        1
 34        35       154        1
115        18       199        1
183       187       192        1
171       114        51        1
154        35       156        1
228       223       132        1
114        50        51        1
 14        15       158        1
 63       113        62        1
 32        33       159        1
113        81        82        1
149       168       160        1
 26       140       161        1
150       174       170        1
112       216       163        1
141       143       204        1
 62       147       165        1
 44        45       166        1
115       205       214        1
149       164       168        1
 24        25       182        1
166       162       170        1
110       114       171        1
246       141       204        1
 60       107       199        1
154       156       174        1
109        19       104        1
135       151       177        1
 23       189       103        1
 37        38       176        1
108        52       103        1
181       186       184        1
 61       205       178        1
155       250       179        1
173        60       180        1
177       151       186        1
 25       112       208        1
185        39       187        1
106        52       108        1
181       177       186        1
102       105        58        1
 38        39       185        1
105        20        58        1
```

151	155	188	1
60	104	59	1
1	2	192	1
104	19	59	1
155	192	188	1
24	206	189	1
179	42	190	1
103	53	23	1
188	192	194	1
59	102	58	1
155	183	192	1
102	19	105	1
191	188	194	1
23	101	22	1
193	191	194	1
101	54	22	1
149	98	99	1
100	164	99	1
162	166	249	1
33	150	159	1
151	135	197	1
195	250	146	1
152	116	50	1
114	152	50	1
153	49	50	1
116	153	50	1
176	35	36	1
33	154	174	1
250	151	146	1
42	179	250	1
176	36	37	1
44	170	190	1
157	124	234	1
242	237	128	1
158	15	92	1
94	158	93	1
236	150	222	1
32	159	31	1
160	14	247	1
97	160	96	1
161	112	25	1
26	161	25	1
162	144	243	1
157	237	134	1
163	116	152	1
114	212	152	1
168	172	13	1
99	164	149	1
165	147	111	1
62	165	61	1
249	45	46	1
222	170	162	1
167	17	18	1
115	167	18	1
168	164	172	1
160	168	13	1
169	110	171	1
108	206	106	1
190	156	179	1
166	170	44	1
171	51	203	1
169	171	106	1
172	11	12	1
13	172	12	1
173	109	104	1
60	173	104	1
174	156	170	1
33	174	150	1
175	10	11	1
172	175	11	1
40	187	39	1

```
 35       176       156         1
221       181         9         1
204       177       175         1
178       107        60         1
 61       178        60         1
190        42        43         1
176       185       156         1
180        18       109         1
173       180       109         1
181         8         9         1
 10       221         9         1
182       110       169         1
 24       182       169         1
183       155       179         1
156       183       179         1
184         7         8         1
181       184         8         1
185       183       156         1
 38       185       176         1
186         6         7         1
184       186         7         1
187       183       185         1
  1       187        40         1
188       191         6         1
151       188       186         1
189       108       103         1
 24       189        23         1
190        43        44         1
156       190       170         1
191         4         5         1
  6       191         5         1
192       187         1         1
  3       194         2         1
193         3         4         1
191       193         4         1
194       192         2         1
193       194         3         1
195       146       244         1
 70       195        69         1
196        53        54         1
 23       196       101         1
238       135       143         1
 68       220        67         1
198        48        49         1
145       198        49         1
199       107       115         1
180       199        18         1
236        30        31         1
134       237       162         1
201        46        47         1
223       243        47         1
202       143       141         1
207       232       141         1
203        52       106         1
171       203       106         1
204       135       177         1
172       246       175         1
205       111       214         1
178       205       115         1
206       169       106         1
189       206       108         1
207       141       164         1
 72       207        71         1
212       112       163         1
182       208       110         1
209        82        83         1
147       209       117         1
210       140        26         1
 27       211        26         1
234        27       225         1
215       211       120         1
```

```
212       208       112         1
152       212       163         1
213       139       215         1
136       240       120         1
214       148       137         1
115       214       167         1
215       122       216         1
213       215       120         1
216       140       210         1
215       216       210         1
217        17       167         1
214       217       167         1
218       136       219         1
157       228       124         1
231       120       211         1
218       219       124         1
220       146       197         1
 67       220       197         1
221       177       181         1
175       221        10         1
222       150       170         1
200       222       162         1
223        47       227         1
218       228       132         1
224       133        29         1
235       224        29         1
225        28       233         1
157       234       128         1
226        28        29         1
133       226        29         1
227       132       223         1
 48       227        47         1
228       218       124         1
134       228       157         1
229        16       142         1
123       229       142         1
230       143       202         1
 67       230        66         1
234       211        27         1
219       231       124         1
232       202       141         1
 72       241       207         1
233       128       225         1
226       233        28         1
234       124       231         1
211       234       231         1
235       200       127         1
 30       235        29         1
236        31       159         1
150       236       159         1
237       200       162         1
128       237       157         1
238       197       135         1
230       238       143         1
239       202       232         1
131       239       232         1
240       145       213         1
218       240       136         1
241       232       207         1
 74       241        73         1
242       128       233         1
133       242       233         1
243       201        47         1
162       243       134         1
244        69       195         1
220       244       146         1
245       153       116         1
213       245       139         1
246       204       175         1
164       246       172         1
247        14       158         1
```

```
        94        247        158          1
       248         66        230          1
       129        248        230          1
       249         46        144          1
       162        249        144          1
       250         41         42          1
       151        250        155          1   :(CONN(:,K), REFT, K=1,...,NELT)
  1    1   2    1
  2    2   3    1
  3    3   4    1
  4    4   5    1
  5    5   6    1
  6    6   7    1
  7    7   8    1
  8    8   9    1
  9    9  10    1
 10   10  11    1
 11   11  12    1
 12   12  13    1
 13   13  14    1
 14   14  15    1
 15   15  16    1
 16   16  17    1
 17   17  18    1
 18   18  19    1
 19   19  20    1
 20   20  21    1
 21   21  22    1
 22   22  23    1
 23   23  24    1
 24   24  25    1
 25   25  26    1
 26   26  27    1
 27   27  28    1
 28   28  29    1
 29   29  30    1
 30   30  31    1
 31   31  32    1
 32   32  33    1
 33   33  34    1
 34   34  35    1
 35   35  36    1
 36   36  37    1
 37   37  38    1
 38   38  39    1
 39   39  40    1
 40   40   1    1
 41   41  42    1
 42   42  43    1
 43   43  44    1
 44   44  45    1
 45   45  46    1
 46   46  47    1
 47   47  48    1
 48   48  49    1
 49   49  50    1
 50   50  51    1
 51   51  52    1
 52   52  53    1
 53   53  54    1
 54   54  55    1
 55   55  56    1
 56   56  57    1
 57   57  58    1
 58   58  59    1
 59   59  60    1
 60   60  61    1
 61   61  62    1
 62   62  63    1
 63   63  64    1
 64   64  65    1
```

```
  65  65  66   1
  66  66  67   1
  67  67  68   1
  68  68  69   1
  69  69  70   1
  70  70  41   1
  71  71  72   1
  72  72  73   1
  73  73  74   1
  74  74  75   1
  75  75  76   1
  76  76  77   1
  77  77  78   1
  78  78  79   1
  79  79  80   1
  80  80  81   1
  81  81  82   1
  82  82  83   1
  83  83  84   1
  84  84  85   1
  85  85  86   1
  86  86  87   1
  87  87  88   1
  88  88  89   1
  89  89  90   1
  90  90  91   1
  91  91  92   1
  92  92  93   1
  93  93  94   1
  94  94  95   1
  95  95  96   1
  96  96  97   1
  97  97  98   1
  98  98  99   1
  99  99 100   1
 100 100  71   1   :(I1, EDGE(:,I), I=1,NEDGE)
'N'                :Printing option 'Y' or 'N'
```

10.3 Program Results

```
 nag_mesh2d_renum (d06ccc) Example Program Results

 The Matrix Sparsity characteristics
 before the renumbering
 nv  =   250
 nnz =  1556

 Initial half-bandwidth:   234    Initial profile: 18233
   Final half-bandwidth:    28      Final profile:  4038

 The Matrix Sparsity characteristics
 after the renumbering
 nv   =   250
 nnz  =  1556
 nelt =   402
```

Example Program

Figure 1: Mesh of the Geometry

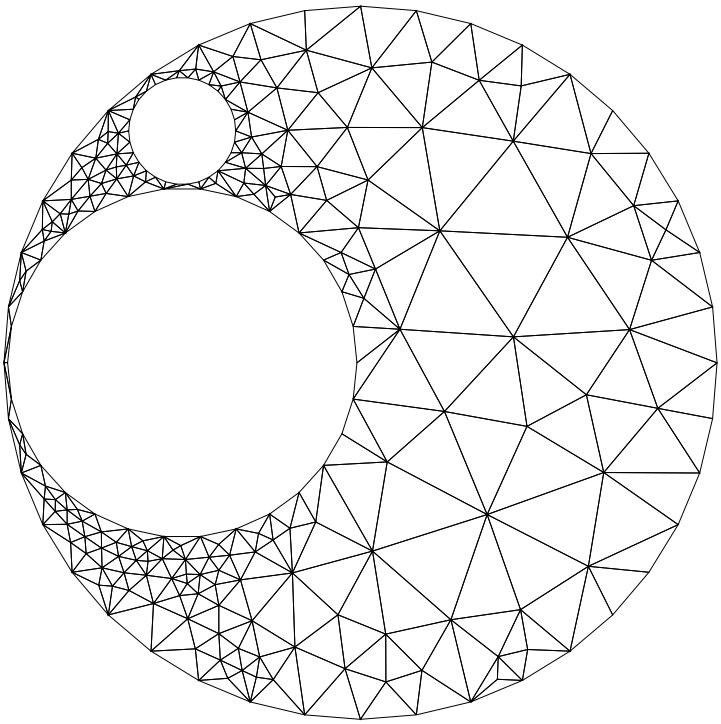

Figure 2: Sparsity of the FE Matrix Before Renumbering

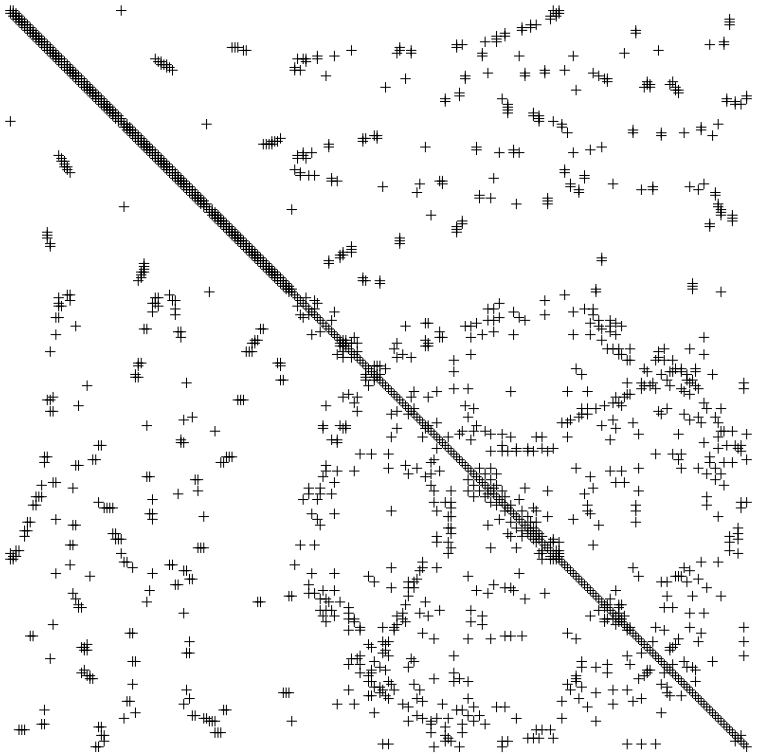

Figure 3: Sparsity of the FE Matrix After Renumbering