NAG Toolbox

nag_mesh_2d_sparsity (d06cb)

1 Purpose

nag_mesh_2d_sparsity (d06cb) generates the sparsity pattern of a finite element matrix associated with a given mesh.

2 Syntax

```
[nnz, irow, icol, ifail] = nag_mesh_2d_sparsity(nv, nnzmax, conn, 'nelt', nelt)
[nnz, irow, icol, ifail] = d06cb(nv, nnzmax, conn, 'nelt', nelt)
```

3 Description

nag_mesh_2d_sparsity (d06cb) generates the sparsity pattern of a finite element matrix associated with a given mesh. The sparsity pattern is returned in a coordinate storage format consistent with the sparse linear algebra functions in Chapter F11. More precisely nag_mesh_2d_sparsity (d06cb) returns the number of nonzero elements in the associated sparse matrix, and their row and column indices. This is designed to assist you in applying finite element discretization to meshes from the D06 Chapter Introduction and in solving the resulting sparse linear system using functions from Chapter F11.

The output sparsity pattern is based on the fact that finite element matrix A has elements a_{ij} satisfying:

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

4 References

None.

5 Parameters

5.1 Compulsory Input Parameters

1: **nv** – INTEGER

The total number of vertices in the input mesh.

Constraint: $\mathbf{nv} \geq 3$.

2: **nnzmax** – INTEGER

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_mesh_2d_sparsity (d06cb) is called.

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

3: $\mathbf{conn}(\mathbf{3}, \mathbf{nelt})$ – INTEGER array

The connectivity of the mesh between triangles and vertices. For each triangle j, $\mathbf{conn}(i,j)$ gives the indices of its three vertices (in anticlockwise order), for $i = 1,2,3$ and $j = 1,2,\dots,\mathbf{nelt}$.

Constraint: $1 \leq \mathbf{conn}(i,j) \leq \mathbf{nv}$ and $\mathbf{conn}(1,j) \neq \mathbf{conn}(2,j)$ and $\mathbf{conn}(1,j) \neq \mathbf{conn}(3,j)$ and $\mathbf{conn}(2,j) \neq \mathbf{conn}(3,j)$, for $i = 1,2,3$ and $j = 1,2,\dots,\mathbf{nelt}$.

5.2 Optional Input Parameters

1: **nelt** – INTEGER

Default: the dimension of the array **conn**.

The number of triangles in the input mesh.

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

5.3 Output Parameters

1: **nz** – INTEGER

The number of nonzero entries in the matrix associated with the input mesh.

2: **irow**(**nnzmax**) – INTEGER array
3: **icol**(**nnzmax**) – INTEGER array

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

4: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **nv** < 3,
or **nelt** $> 2 \times$ **nv** $- 1$,
or **nnzmax** $< 4 \times$ **nelt** $+$ **nv** or **nnzmax** $>$ **nv**^{2}
or **conn**$(i, j) < 1$ or **conn**$(i, j) >$ **nv** for some $i = 1, 3$ and j, $1 \leq j \leq$ **nelt**,
or **conn**$(1, j) =$ **conn**$(2, j)$ or **conn**$(1, j) =$ **conn**$(3, j)$ or
conn$(2, j) =$ **conn**$(3, j)$ for some $j = 1, 2, \ldots,$ **nelt**.

ifail $= 2$

A serious error has occurred in an internal call to an auxiliary function. Check the input mesh, especially the connectivity between triangles and vertices (the argument **conn**). Array dimensions should be checked as well. If the problem persists, contact NAG.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

See Section 10 in nag_mesh_2d_renumber (d06cc).

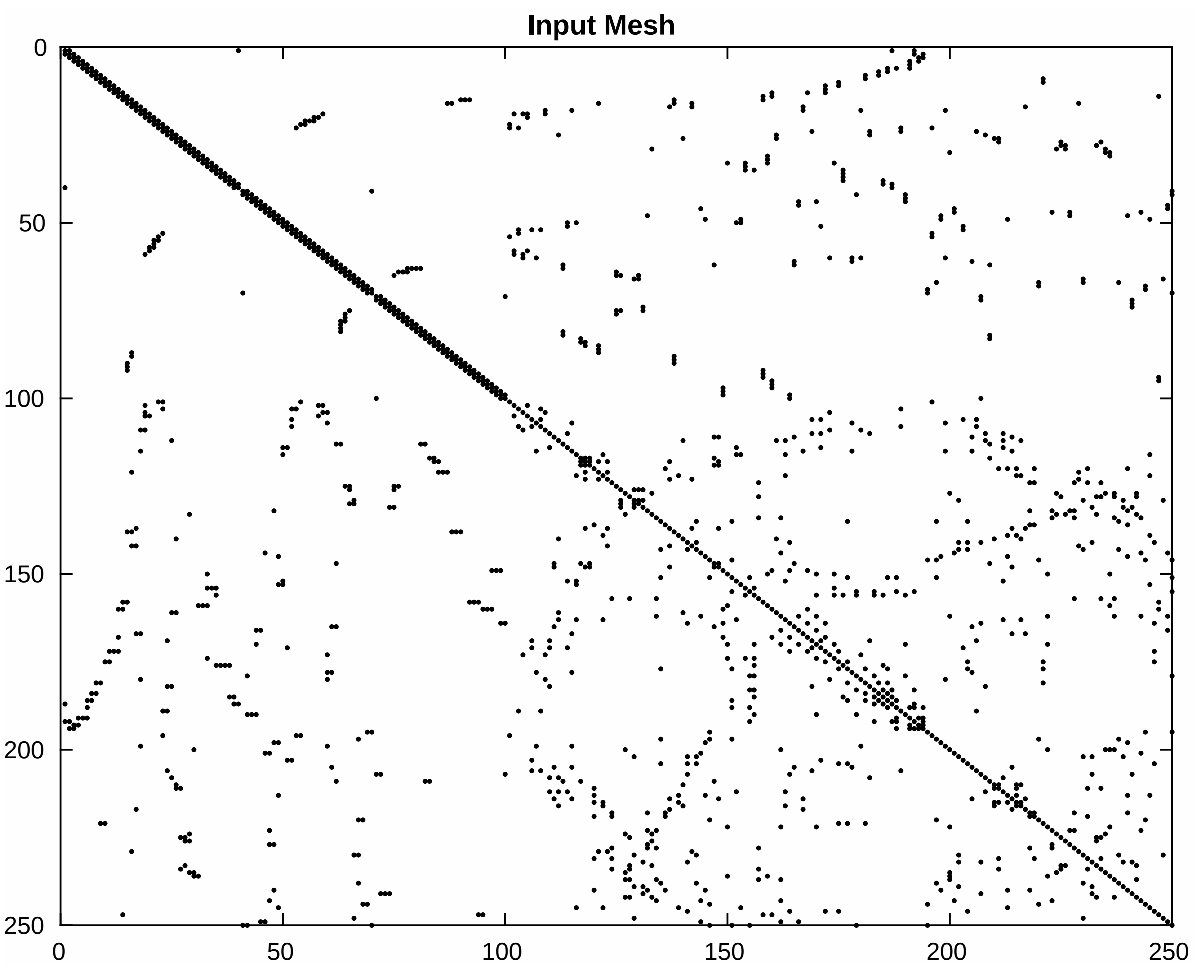
9.1 Program Text

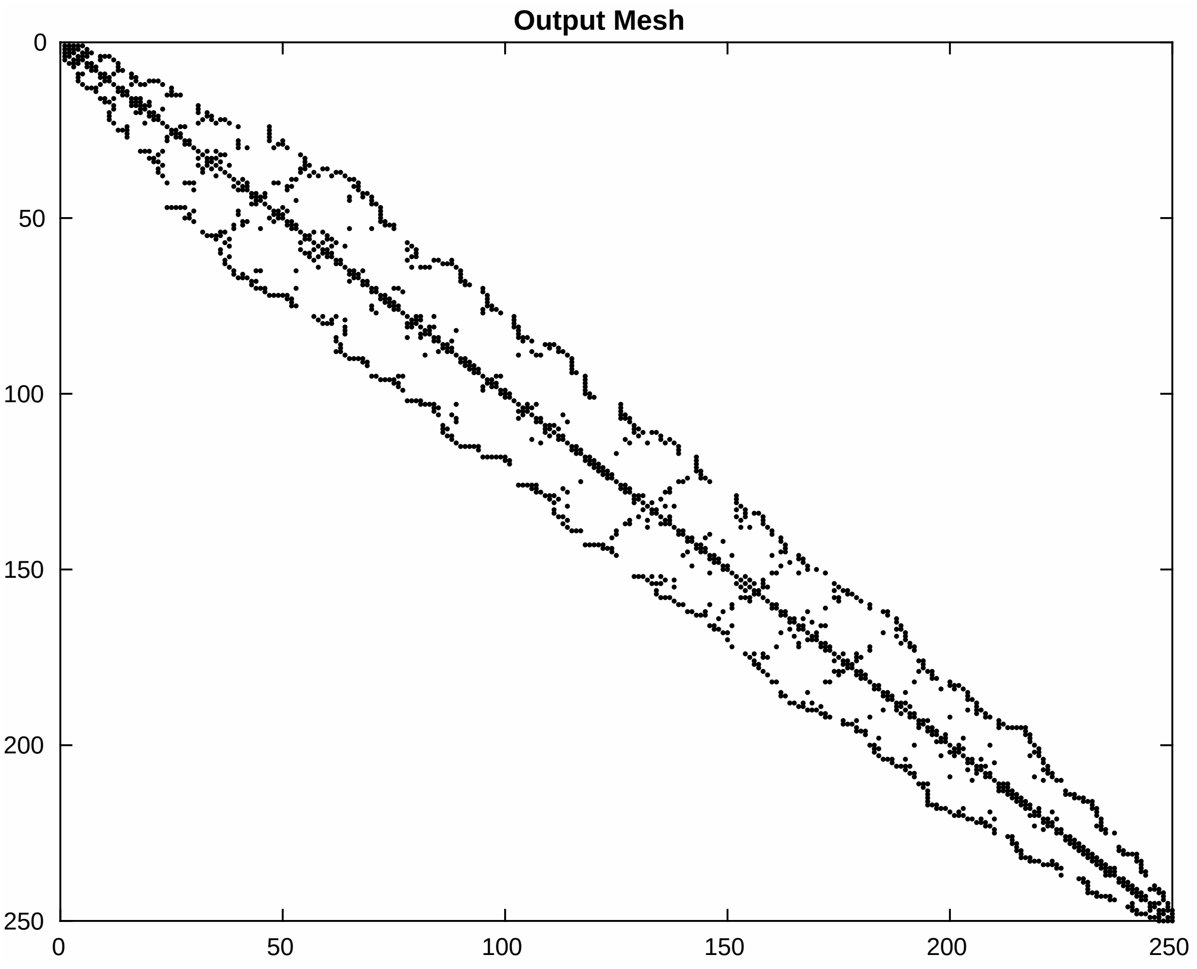
```
     function d06cb_example

fprintf('d06cb example results\n\n');

edge = zeros(3, 100, nag_int_name);
coor = zeros(2, 250);

% Define boundaries
ncirc     = 3; % 3 circles
nvertices = [40, 30, 30];
radii     = [1, 0.49, 0.15];
centres   = [0, 0; -0.5, 0; -0.5, 0.65];

% First circle is outer circle
csign = 1;
i1 = 0;
nvb = 0;
for icirc = 1:ncirc
   for i = 0:nvertices(icirc)-1
      i1 = i1+1;
      theta = 2*pi*i/nvertices(icirc);
      coor(1,i1) = radii(icirc)*cos(theta) + centres(icirc, 1);
      coor(2,i1) = csign*radii(icirc)*sin(theta) +  centres(icirc, 2);
      edge(1,i1) = i1;
      edge(2,i1) = i1 + 1;
      edge(3,i1) = 1;
   end
   edge(2,i1) = nvb + 1;
   nvb = nvb + nvertices(icirc);
   % Subsequent circles are inner circles
   csign = -1;
end
nedge = nvb;

% Initialise mesh control parameters
bspace = zeros(1, 100);
bspace(1:nvb) = 0.05;
smooth = true;
itrace = nag_int(0);

nnzmax = nag_int(3000);

% Mesh geometry
[nv, nelt, coor, conn, ifail] = d06aa(edge, coor, bspace, smooth, itrace);

% Compute the sparsity of the FE matrix from the input geometry
[nz, irow, icol, ifail] = d06cb(nv, nnzmax, conn, 'nelt', nelt);

fprintf('\nNumber of non-zero entries in input mesh:  %d\n', nz);

% Plot sparsity of input mesh
fig1 = figure;
plot(irow(1:double(nz)), icol(1:double(nz)), '.');
title ('Input Mesh');
set(gca,'YDir','reverse');

% Call the renumbering routine and get the new sparsity
[nz, coor, edge, conn, irow, icol, ifail] = ...
    d06cc(nnzmax, coor, edge, conn, itrace, 'nelt', nelt);
```

```
fprintf('Number of non-zero entries in output mesh: %d\n', nz);
% Plot smoothed mesh
fig2 = figure;
plot(irow(1:double(nz)), icol(1:double(nz)), '.');
title ('Output Mesh');
set(gca,'YDir','reverse');
```

9.2 Program Results

```
     d06cb example results

Number of non-zero entries in input mesh:  1556
Number of non-zero entries in output mesh: 1556
```




Output Mesh
0
50
100
150
200
250
0
50
100
150
200
250