UNIVERSIDAD POLITECNICA DE VALENCIA
DEPARTAMENTO DE INGENIERIA MECANICA Y DE MATERIALES

ELEMENTOS FINITOS

(E.T.S.I.I.V)

IMPLEMENTACION COMPUTACIONAL DEL METODO EF

LECCION 12.- PROGRAMA COMPLETO DE EF
PROBLEMA TENSION PLANA

J. L. OLIVER
Dr. Ingeniero Industrial

Valencia, 2005

001 PASOS A SEGUIR PARA REALIZAR UN ANALISIS POR EF CURSO 2004-5

***Preprocessing*: problem definition**

***Processing*: problem solution**

***Postprocessing*: showing results**

001 ESTRUCTURA DE DATOS PARA LA DEFINICION DEL PROBLEMA CURSO 2004-5

Geometry Data Set:

NodeCoordinates

Element Data Set:

ElemTypes, ElemNodeLists, ElemMaterial, ElemFabrication

Degree of Freedom Data Set:

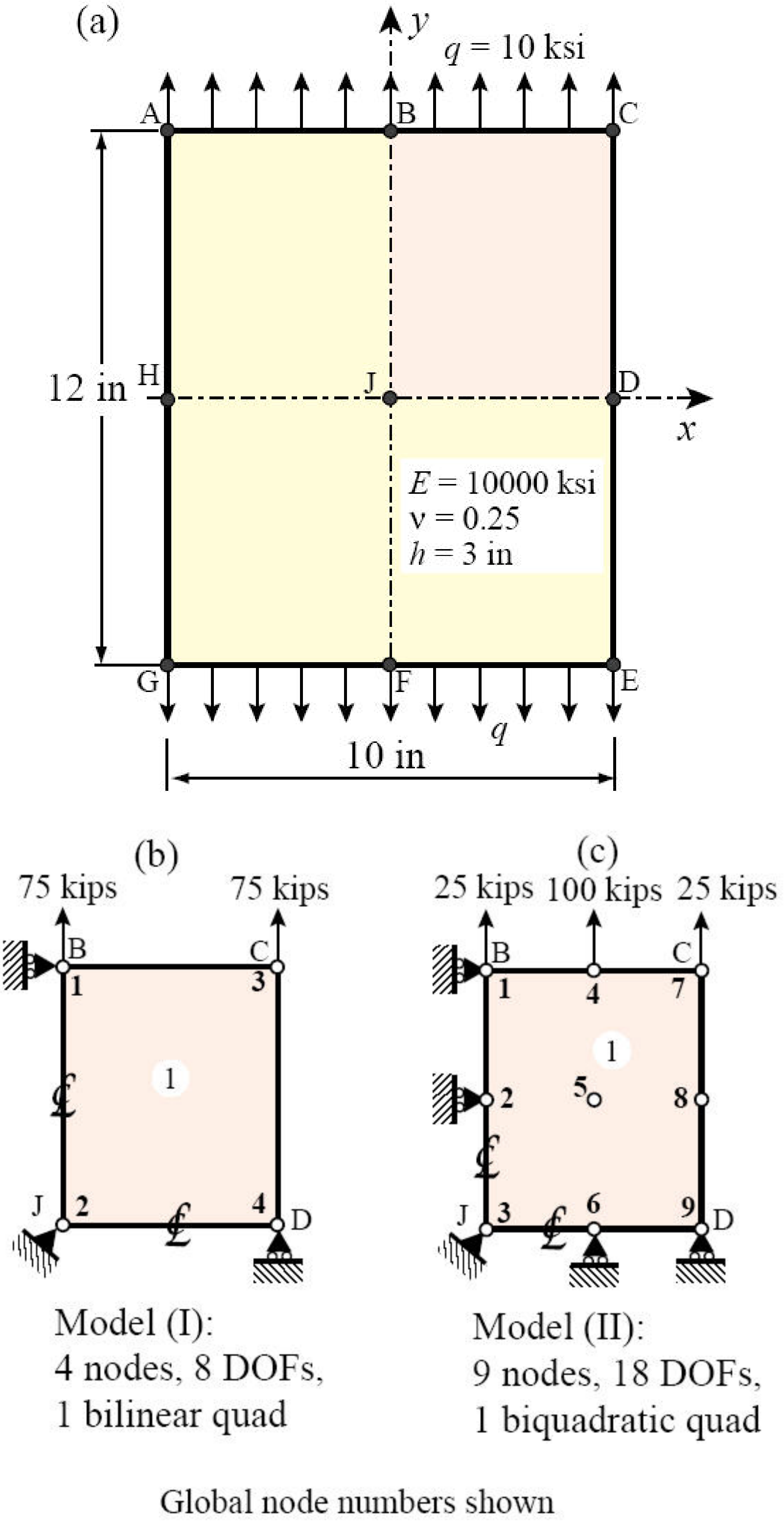
FreedomTags, FreedomValues

Miscellaneous Processing Data Set:

ProcessOptions

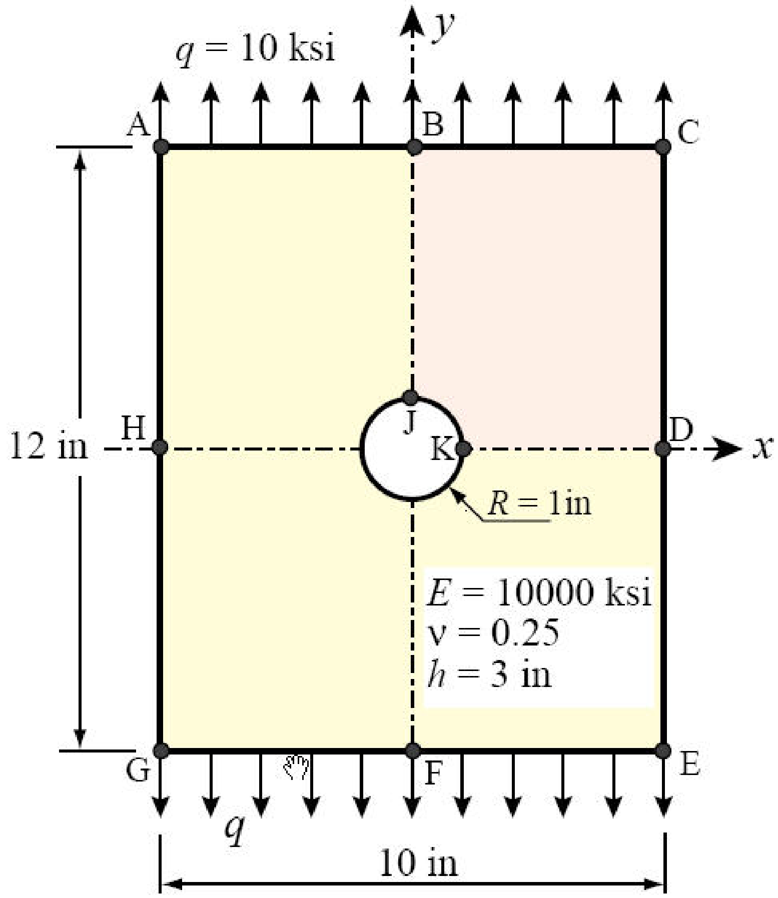
MODELO CON UN SOLO ELEMENTO

Figure 27.1 is a rectangular plate in plane stress under uniform uniaxial loading. Its analytical solution is $\sigma_{yy} = q$, $\sigma_{xx} = \sigma_{xy} = 0$, $u_y = qy/E$, $u_x = -\nu qx/E$. This problem should be solved exactly by *any* finite element mesh. In particular, the two one-element models shown on the right of that figure.

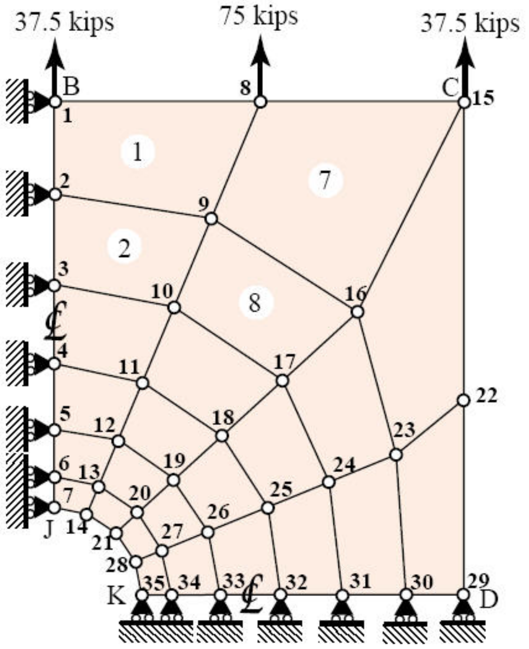


001 EJEMPLO PREPARACION DATOS CURSO 2004-5

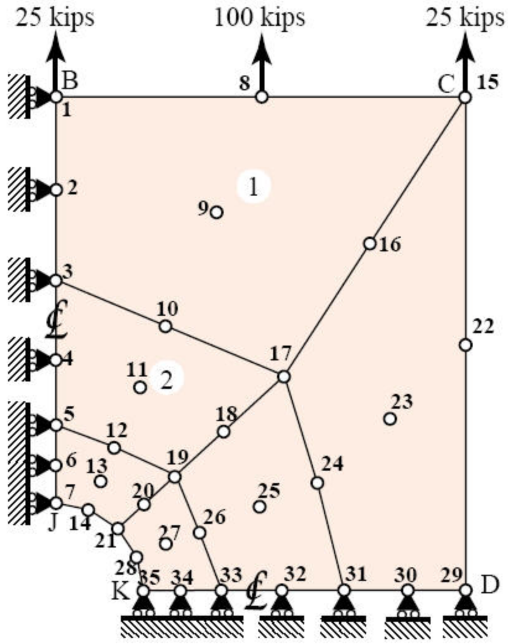
MODELO CON MALLA GENERADA



Node 8 is exactly midway between 1 and 15



Model (I): 35 nodes, 70 DOFs, 24 bilinear quads



Model (II): 35 nodes, 70 DOFs, 6 biquadratic quads

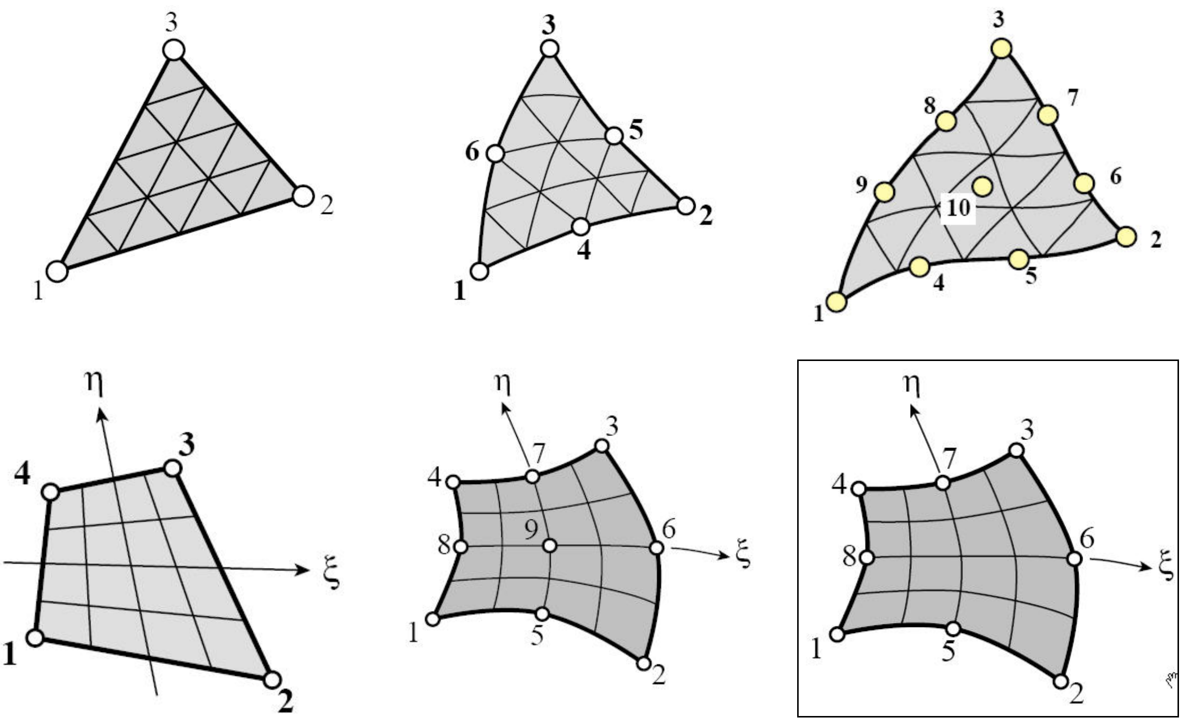
COORDENADAS NODALES

```
NodeCoordinates = {{x1, y1},{x2, y2}, ... {xN, yN}}
```

TIPO DE ELEMENTO

$$\texttt{ElemTypes} = \{\{\texttt{etyp}^{(1)}\},\ \{\texttt{etyp}^{(2)}\},\ \ldots\ \{\texttt{etyp}^{(N_e)}\}\}$$

Identifier	Plane Stress Model
"Trig3"	3-node linear triangle
"Trig6"	6-node quadratic triangle
"Trig10"	10-node cubic triangle
"Quad4"	4-node blinear quad
"Quad9"	9-node biquadratic quad



CONECTIVIDAD ELEMENTOS

$$\texttt{ElemNodeLists} = \{\{\texttt{eNL}^{(1)}\},\ \{\texttt{eNL}^{(2)}\},\ \ldots\ \{\texttt{enL}^{(N_e)}\}\}$$

Element boundaries must be traversed counterclockwise but you can start at any corner. Numbering elements with midnodes requires more care: first list corners counterclockwise, followed by midpoints (first midpoint is the one that follows first corner when going counterclockwise). When elements have an interior node, as in the 9-node biquadratic quadrilateral, that node goes last.

PROPIEDADES DEL MATERIAL

$$\texttt{ElemMaterial = }\{\{E^{(1)}, \nu^{(1)}\},\ \{E^{(2)}, \nu^{(2)}\},\ \ldots\ \{E^{(N_e)}, \nu^{(N_e)}\}\}$$

In the common case in which all elements have the same E and ν, this list can be easily generated by a `Table` instruction.

```
ElemMaterial=Table[{Em,nu},{numele}]
numele=Length[ElemeNodeLists]
```

PROPIEDADES GEOMETRICAS

$$\texttt{ElemFabrication = }\{\{h(1)\},\ \{h^{(2)}\},\ \ldots\ \{h^{(N_e)}\}\}$$

INDICADORES DE LIBERTAD – DISPLAZAMIENTO/FUERZA

`FreedomTags` labels each node degree of freedom as to whether the load or the displacement is specified. The configuration of this list is similar to that of `NodeCoordinates`:

$$\texttt{FreedomTags=}\{\{\texttt{tag}_{x1}, \texttt{tag}_{y1}\},\{\texttt{tag}_{x2}, \texttt{tag}_{y2}\},\ \ldots\ \{\texttt{tag}_{xN}, \texttt{tag}_{yN}\}\}$$

The tag value is 0 if the force is specified and 1 if the displacement is specified. When there are a lot of nodes, the quickest way to specify this list is to start from all zeros, and then insert the boundary conditions appropriately.

VALORES DE LIBERTAD – DISPLAZAMIENTO/FUERZA

`FreedomValues` has the same node by node configuration as `FreedomTags`. It lists the specified values of the applied node force component if the corresponding tag is zero, and of the prescribed displacement component if the tag is one. Typically most of the list entries are zero.

OPCIONES DE PROCESADO

```
ProcessingOptions={True};
```

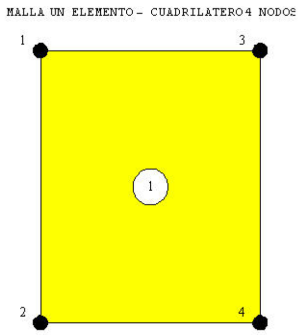
This specifies floating point numerical computations.

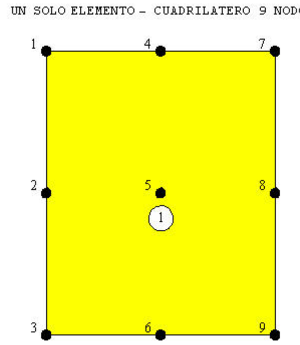
VISUALIZACION DE LA DISCRETIZACION (MALLA)

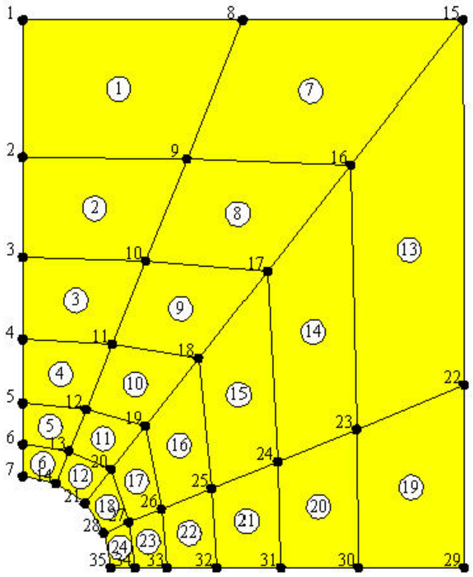
```
aspect=6/5;
Plot2DElementsAndNodes[NodeCoordinates,ElemNodeLists,aspect,
  "Plate with circular hole - 4-node quad model",True,True];
```

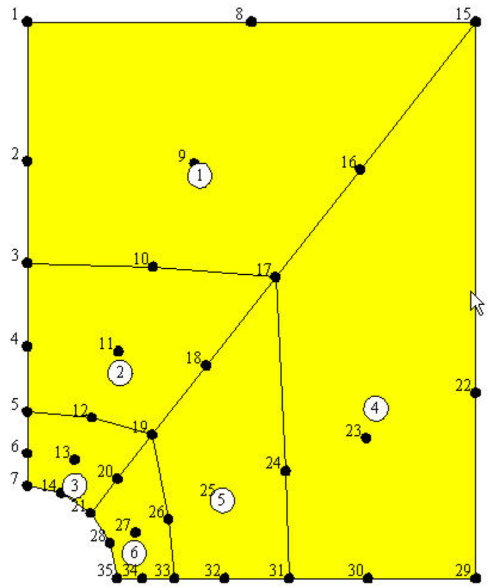
Here `aspect` is the plot frame aspect ratio (y dimension over x dimension), and the last two `True` argument values specify that node labels and element labels, respectively, be shown.

001 VISUALIZACION MALLAS EJEMPLOS CURSO 2004-5









001 PROCESADO DEL MODELO CURSO 2004-5

```
{u,f,sig}=MembraneSolution[NodeCoordinates,ElemTypes,
          ElemNodeLists,ElemMaterial,ElemFabrication,
          ProcessingOptions,FreedomTags,FreedomValues];
```

```
MembraneSolution[nodcoor_,eletyp_,elenod_,elemat_,
  elefab_,eleopt_,doftag_,dofval_]:= Module[{K,Kmod,u,f,sig,j,n,ns,
  supdof,supval,numnod=Length[nodcoor],numele=Length[elenod]},
  u=f=sig={};
  K=MembraneMasterStiffness[nodcoor,
          eletyp,elenod,elemat,elefab,eleopt];  K=N[K];
  ns=0; Do [Do [If[doftag[[n,j]]>0,ns++],{j,1,2}],{n,1,numnod}];
  supdof=supval=Table[0,{ns}];
  k=0; Do [Do [If[doftag[[n,j]]>0,k++;supdof[[k]]=2*(n-1)+j;
         supval[[k]]=dofval[[n,j]]]],{j,1,2}],{n,1,numnod}];
  f=ModifiedNodeForces[supdof,supval,K,Flatten[dofval]];
  Kmod=ModifiedMasterStiffness[supdof,K];
  u=Simplify[Inverse[Kmod].f]; u=Chop[u];
  f=Simplify[K.u]; f=Chop[f];
  sig=MembraneNodalStresses[nodcoor,
        eletyp,elenod,elemat,elefab,eleopt,u];
  sig=Chop[sig];
  Return[{u,f,sig}];
];
```

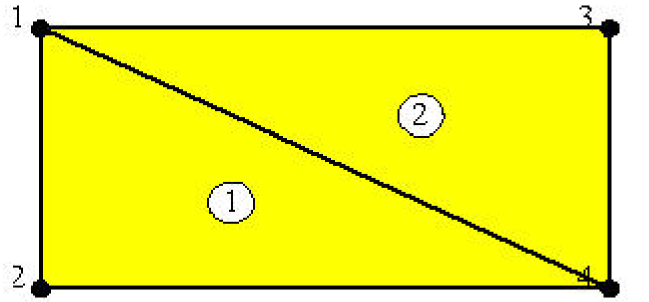
1 - ENSAMBLADO DE LA MATRIZ RIGIDEZ GLOBAL

The function begins by assembling the free-free master stiffness matrix K by calling the module MembraneMasterStiffness. this module is listed in Figure 27.5. As a study of its code reveals, it can handle the five element types described in the previous section. The modules that compute the element stiffness matrices have been studied in previous Chapters and need not be listed here.

```
MembraneMasterStiffness[nodcoor_,eletyp_,elenod_,
  elemat_,elefab_,eleopt_]:=
  Module[{numele=Length[elenod],numnod=Length[nodcoor],numer,
   ne,eNL,eftab,neldof,i,n,Em,ν,Emat,th,ncoor,Ke,K},
  K=Table[0,{2*numnod},{2*numnod}];  numer=eleopt[[1]];
  For [ne=1,ne<=numele,ne++,
       {type}=eletyp[[ne]];
       If [type!="Trig3"&&type!="Trig6"&&type!="Trig10"&&
           type!="Quad4"&&type!="Quad9",
           Print["Illegal element type,ne=",ne]; Return[Null]];
       eNL=elenod[[ne]]; n=Length[eNL];
       eftab=Flatten[Table[{2*eNL[[i]]-1,2*eNL[[i]]},{i,n}]];
       ncoor=Table[nodcoor[[eNL[[i]]]],{i,n}];
       {Em,ν}=elemat[[ne]];
       Emat=Em/(1-ν^2)*{{1,ν,0},{ν,1,0},{0,0,(1-ν)/2}};
       {th}=elefab[[ne]];
       If [type=="Trig3", Ke=Trig3IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer}] ];
       If [type=="Quad4", Ke=Quad4IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer,2}] ];
       If [type=="Trig6", Ke=Trig6IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer,3}] ];
       If [type=="Quad9", Ke=Quad9IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer,3}] ];
       If [type=="Trig10",Ke=Trig10IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer,3}] ];
       neldof=Length[Ke];
       For [i=1,i<=neldof,i++,ii=eftab[[i]];
         For [j=i,j<=neldof,j++,jj=eftab[[j]];
             K[[jj,ii]]=K[[ii,jj]]+=Ke[[i,j]]
         ];
       ];
     ];
  Return[K];
];
```

EJEMPLO ENSAMBLADO DE LAS MATRICES DE DOS TRAINGULOS

MALLA DOS ELEMENTOS - TRAINGULOS 3 NODOS



Out[37]//MatrixForm=

	1	2	3	4	7	8
1	$\frac{24}{5}$	0	$-\frac{24}{5}$	$-\frac{12}{5}$	0	$\frac{12}{5}$
2	0	$\frac{64}{5}$	$-\frac{8}{5}$	$-\frac{64}{5}$	$\frac{8}{5}$	0
3	$-\frac{24}{5}$	$-\frac{8}{5}$	8	4	$-\frac{16}{5}$	$-\frac{12}{5}$
4	$-\frac{12}{5}$	$-\frac{64}{5}$	4	14	$-\frac{8}{5}$	$-\frac{6}{5}$
7	0	$\frac{8}{5}$	$-\frac{16}{5}$	$-\frac{8}{5}$	$\frac{16}{5}$	0
8	$\frac{12}{5}$	0	$-\frac{12}{5}$	$-\frac{6}{5}$	0	$\frac{6}{5}$

Out[36]//MatrixForm=

	1	2	7	8	5	6
1	2	0	0	2	-2	-2
2	0	$\frac{1}{2}$	1	0	-1	$-\frac{1}{2}$
7	0	1	2	0	-2	-1
8	2	0	0	8	-2	-8
5	-2	-1	-2	-2	4	3
6	-2	$-\frac{1}{2}$	-1	-8	3	$\frac{17}{2}$

	1	2	3	4	5	6	7	8
1	$\frac{34}{5}$	0	$-\frac{24}{5}$	$-\frac{12}{5}$	-2	-2	0	$\frac{22}{5}$
2	0	$\frac{133}{10}$	$-\frac{8}{5}$	$-\frac{64}{5}$	-1	$-\frac{1}{2}$	$\frac{13}{5}$	0
3	$-\frac{24}{5}$	$-\frac{8}{5}$	8	4	0	0	$-\frac{16}{5}$	$-\frac{12}{5}$
4	$-\frac{12}{5}$	$-\frac{64}{5}$	4	14	0	0	$-\frac{8}{5}$	$-\frac{6}{5}$
5	-2	-1	0	0	4	3	-2	-2
6	-2	$-\frac{1}{2}$	0	0	3	$\frac{17}{2}$	-1	-8
7	0	$\frac{13}{5}$	$-\frac{16}{5}$	$-\frac{8}{5}$	-2	-1	$\frac{26}{5}$	0
8	$\frac{22}{5}$	0	$-\frac{12}{5}$	$-\frac{6}{5}$	-2	-8	0	$\frac{46}{5}$

MATRIZ RIGIDEZ GLOBAL

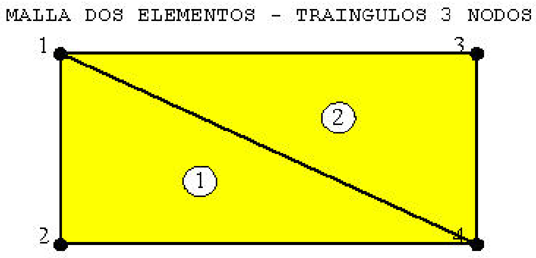
2 - APLICACIÓN CONDICIONES DE CONTORNO Y CARGAS

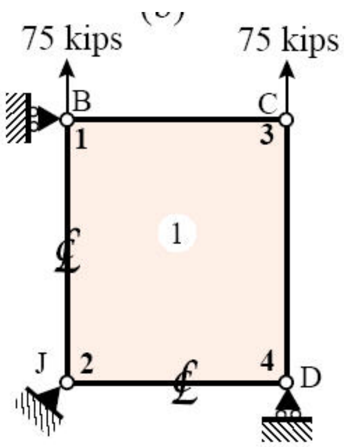
The displacement boundary conditions are applied by modules `ModifiedmasterStiffness` and `ModifiedNodeForces`, which return the modified stiffness matrix $\hat{\mathbf{K}}$ and the modified force vector $\hat{\mathbf{f}}$ in `Khat` and `fnat`, respectively.

```
ModifiedMasterStiffness[pdof_, Km_] := Module[{i, j, k, K}, K = Km;
   For[k = 1, k ≤ Length[pdof], k++, i = pdof[[k]];
    For[j = 1, j ≤ Length[K], j++, K[[i, j]] = K[[j, i]] = 0]; K[[i, i]] = 1;]; Return[K];];
```

```
ModifiedNodeForces[pdof_, pdofv_, Km_, nfv_] :=
  Module[{i, j, k, l, d, kk = Length[pdof], n = Length[Km], fixed, rhs}, rhs = nfv;
   d = pdofv; fixed = Table[False, {n}];
   Do[i = pdof[[k]]; fixed[[i]] = True, {k, 1, kk}];
   For[k = 1, k ≤ kk, k++, i = pdof[[k]];
    For[j = 1, j ≤ n, j++, If[fixed[[j]], Continue[]];
     rhs[[j]] = rhs[[j]] - Km[[i, j]] * d[[k]];]; rhs[[i]] = d[[k]];];
   Return[rhs];];
```

EJEMPLO DOS TRIANNGULOS





$$
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{133}{10} & 0 & 0 & -1 & -\frac{1}{2} & \frac{13}{5} & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 4 & 3 & -2 & 0 \\
0 & -\frac{1}{2} & 0 & 0 & 3 & \frac{17}{2} & -1 & 0 \\
0 & \frac{13}{5} & 0 & 0 & -2 & -1 & \frac{26}{5} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
$$

MATRIZ RIGIDEZ GLOBAL MODIFICADA CON CONDICIONES CONTORNO

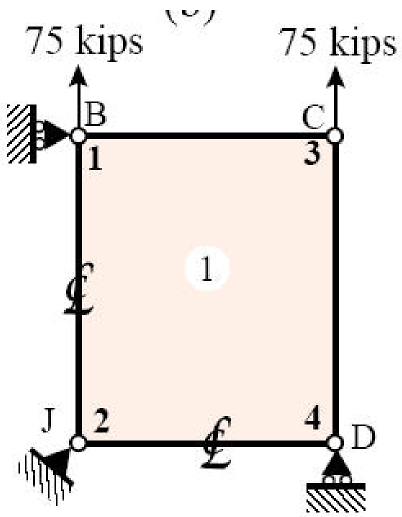
3 - OBTENCION DESPLAZAMIENTOS NODALES Y FUERZAS DE REACCION

The unknown node displacements u are then obtained through the built in `LinearSolve` function, as `u=LinearSolve[Khat,fhat]`. This solution strategy is of course restricted to very small systems, but it has the advantages of simplicity.

The function returns arrays u, `f` and p, which are lists of length 12, 12 and 13, respectively. Array u contains the computed node displacements ordered $u_{x1} < u_{y1}, u_{x2}, \ldots u_{y8}$. Array `f` contains the node forces recovered from $\mathbf{f} = \mathbf{Ku}$; this includes the reactions f_{x1}, f_{y1} and f_{y8}.

```
MembraneSolution[nodcoor_, eletyp_, elenod_, elemat_, elefab_, eleopt_,
   doftag_, dofval_] :=
  Module[{K, Kmod, u, f, sig, j, n, ns, supdof, supval, numnod = Length[nodcoor],
    numele = Length[elenod]}, u = f = sig = {};
   K = MembraneMasterStiffness[nodcoor, eletyp, elenod, elemat, elefab, eleopt];
   Print["Master Stiff K"]; Print[MatrixForm[K]]; K = N[K];
 Print["eigs of K=", Chop[Eigenvalues[K]]]; ns = 0;
   Do[Do[If[doftag[[n, j]] > 0, ns++], {j, 1, 2}], {n, 1, numnod}];
   Print["doftag=", doftag]; Print["ns=", ns]; supdof = supval = Table[0, {ns}];
k = 0; Do[Do[If[doftag[[n, j]] > 0, k++; supdof[[k]] = 2 * (n - 1) + j;
      supval[[k]] = dofval[[n, j]]]], {j, 1, 2}], {n, 1, numnod}]; Print["supdof=", supdof];
   f = ModifiedNodeForces[supdof, supval, K, Flatten[dofval]];
   Print["Vector Fuerzas f"]; Print[MatrixForm[f]];
Kmod = ModifiedMasterStiffness[supdof, K]; Print["Matriz Rigidez Modificada Kmod"];
   Print[MatrixForm[Kmod]];
 Print["eigs of Kmod=", Eigenvalues[Kmod]]; u = Simplify[Inverse[Kmod].f];
   u = Chop[u]; Print["Desplazamientos Nodales u=", MatrixForm[u]];
   f = Simplify[K.u]; f = Chop[f]; Print["Fuerzas de Reaccion f=", MatrixForm[f]];
   sig = MembraneNodalStresses[nodcoor, eletyp, elenod, elemat, elefab, eleopt, u];
sig = Chop[sig];
Return[{u, f, sig}];
];
```

EJEMPLO PLACA CON UN ELEMENTO CUADRILATERO 4 NODOS




```
{u, f, sig} = MembraneSolution[NodeCoordinates, ElemTypes, ElemNodeLists,
   ElemMaterial, ElemFabrication, ElemOptions, FreedomTags, FreedomValues];
```

Master Stiff K

16133.3	-5000.	3066.67	-1000.	-11133.3	1000.	-8066.67	5000.
-5000.	13688.9	1000.	-6488.89	-1000.	-355.556	5000.	-6844.44
3066.67	1000.	16133.3	5000.	-8066.67	-5000.	-11133.3	-1000.
-1000.	-6488.89	5000.	13688.9	-5000.	-6844.44	1000.	-355.556
-11133.3	-1000.	-8066.67	-5000.	16133.3	5000.	3066.67	1000.
1000.	-355.556	-5000.	-6844.44	5000.	13688.9	-1000.	-6488.89
-8066.67	5000.	-11133.3	1000.	3066.67	-1000.	16133.3	-5000.
5000.	-6844.44	-1000.	-355.556	1000.	-6488.89	-5000.	13688.9

eigs of K={42453.9, 24400., 22612.8, 16133.3, 13688.9, 0, 0, 0}

MATRIZ RIGIDEZ GLOBAL Y VALORES PROPIOS

doftag={{1, 0}, {1, 1}, {0, 0}, {0, 1}}

ns=4

supdof={1, 3, 4, 8}

GRADOS DE LIBERTAD RESTRINGIDOS – CONDICIONES CONTORNO

Vector Fuerzas f

$$\begin{pmatrix} 0 \\ 75 \\ 0 \\ 0 \\ 0 \\ 75 \\ 0 \\ 0 \end{pmatrix}$$

VECTOR FUERZAS NODALES APLICADAS

Matriz Rigidez Modificada Kmod

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 13688.9 & 0 & 0 & -1000. & -355.556 & 5000. & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1000. & 0 & 0 & 16133.3 & 5000. & 3066.67 & 0 \\ 0 & -355.556 & 0 & 0 & 5000. & 13688.9 & -1000. & 0 \\ 0 & 5000. & 0 & 0 & 3066.67 & -1000. & 16133.3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

eigs of Kmod={21227., 19575.4, 11306.4, 7535.67, 1., 1., 1., 1.}

MATRIZ RIGIDEZ GLOBAL MODIFICADAS CON CONDICIONES CONTORNO

$$\mathbf{Ku} = \mathbf{f}$$

PROBLEMA A RESOLVER

$$\text{Desplazamientos Nodales u=} \begin{pmatrix} 0 \\ 0.006 \\ 0 \\ 0 \\ -0.00125 \\ 0.006 \\ -0.00125 \\ 0 \end{pmatrix}$$

SOLUCION DESPLAZAMIENTOS NODALES

$$u_y = qy/E, u_x = -\nu qx/E$$

SOLUCION EXACTA

$$\text{Fuerzas de Reaccion f=} \begin{pmatrix} 0 \\ 75. \\ 0 \\ -75. \\ 0 \\ 75. \\ 0 \\ -75. \end{pmatrix}$$

VECTOR FUERZAS APLICADAS Y DE REACCION

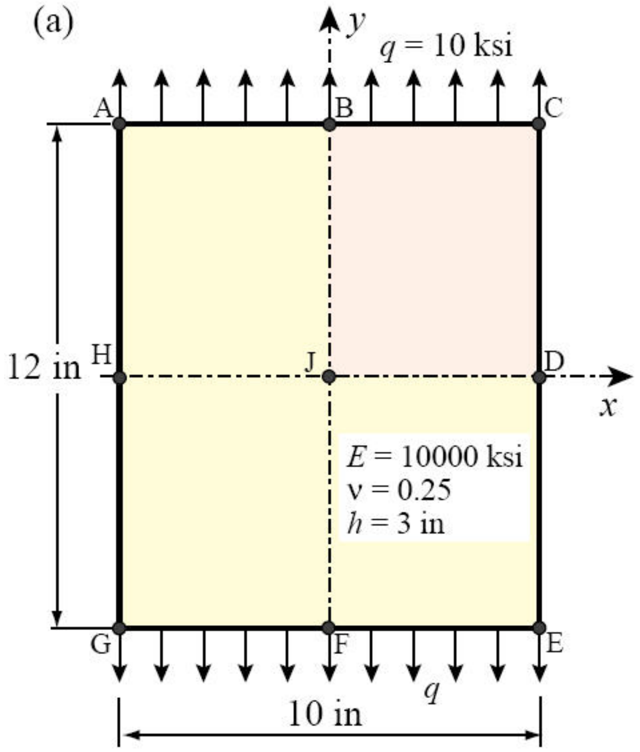
Finally, array sig contains the nodal stresses σ_{xx}, σ_{yy}, σ_{xy} at each node, recovered from the displacement solution. This computation is driven by module MembraneNodeStresses, which is

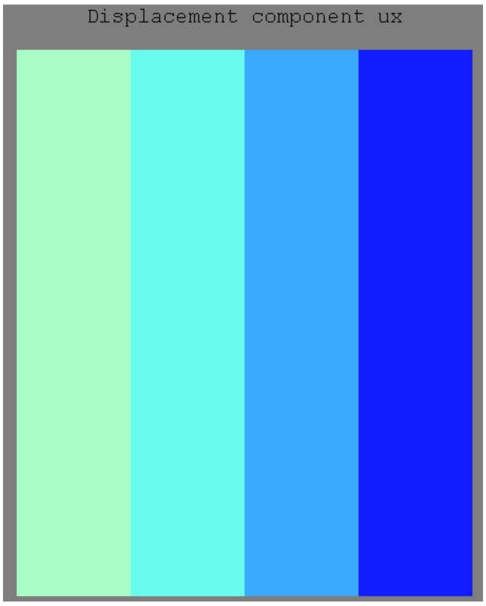
```
MembraneMasterStiffness[nodcoor_,eletyp_,elenod_,
  elemat_,elefab_,eleopt_]:=
  Module[{numele=Length[elenod],numnod=Length[nodcoor],numer,
   ne,eNL,eftab,neldof,i,n,Em,ν,Emat,th,ncoor,Ke,K},
  K=Table[0,{2*numnod},{2*numnod}];  numer=eleopt[[1]];
  For [ne=1,ne<=numele,ne++,
       {type}=eletyp[[ne]];
       If [type!="Trig3"&&type!="Trig6"&&type!="Trig10"&&
           type!="Quad4"&&type!="Quad9",
           Print["Illegal element type,ne=",ne]; Return[Null]];
       eNL=elenod[[ne]]; n=Length[eNL];
       eftab=Flatten[Table[{2*eNL[[i]]-1,2*eNL[[i]]},{i,n}]];
       ncoor=Table[nodcoor[[eNL[[i]]]],{i,n}];
       {Em,ν}=elemat[[ne]];
       Emat=Em/(1-ν^2)*{{1,ν,0},{ν,1,0},{0,0,(1-ν)/2}};
       {th}=elefab[[ne]];
       If [type=="Trig3", Ke=Trig3IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer}] ];
       If [type=="Quad4", Ke=Quad4IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer,2}] ];
       If [type=="Trig6", Ke=Trig6IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer,3}] ];
       If [type=="Quad9", Ke=Quad9IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer,3}] ];
       If [type=="Trig10",Ke=Trig10IsoPMembraneStiffness[ncoor,
          {Emat,0,0},{th},{numer,3}] ];
       neldof=Length[Ke];
       For [i=1,i<=neldof,i++,ii=eftab[[i]];
         For [j=i,j<=neldof,j++,jj=eftab[[j]];
             K[[jj,ii]]=K[[ii,jj]]+=Ke[[i,j]]
         ];
       ];
     ];
  Return[K];
];
```

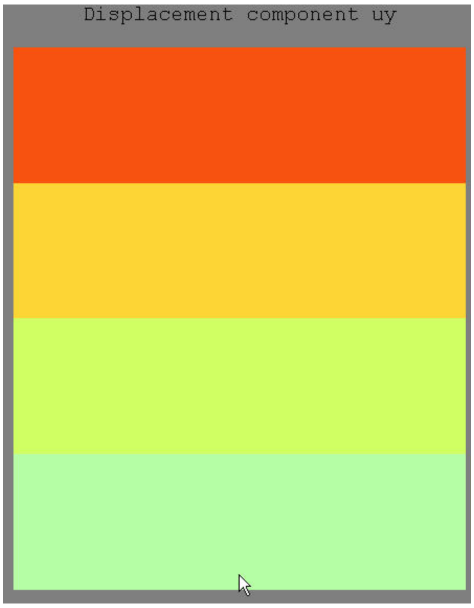
2 VISUALIZACION DESPLAZAMIENTOS

```
ux = uy = Table[0, {numnod}];
Do[ux[[n]] = u[[2*n - 1]]; uy[[n]] = u[[2*n]], {n, 1, numnod}];
uxmax = uymax = 0;
Do[uxmax = Max[Abs[ux[[n]]], uxmax]; uymax = Max[Abs[uy[[n]]], uymax], {n, 1, numnod}];
ContourPlotNodeFuncOver2DMesh[NodeCoordinates, ElemNodeLists, ux, uxmax,
  Nsub, aspect, "Displacement component ux"];
ContourPlotNodeFuncOver2DMesh[NodeCoordinates, ElemNodeLists, uy, uymax,
  Nsub, aspect, "Displacement component uy"];
```

EJEMPLO PLACA CON UN ELEMENTO CUADRILATERO 4 NODOS







2 VISUALIZACION TENSIONES

```
sxx = syy = sxy = Table[0, {numnod}];
Do[{sxx[[n]], syy[[n]], sxy[[n]]} = sig[[n]], {n, 1, numnod}];
sxxmax = syymax = sxymax = 0;
Do[sxxmax = Max[Abs[sxx[[n]]], sxxmax];
  syymax = Max[Abs[syy[[n]]], syymax];
  sxymax = Max[Abs[sxy[[n]]], sxymax], {n, 1, numnod}];
ContourPlotNodeFuncOver2DMesh[NodeCoordinates, ElemNodeLists, sxx, sxxmax,
  Nsub, aspect, "Nodal stress sig-xx"];
ContourPlotNodeFuncOver2DMesh[NodeCoordinates, ElemNodeLists, syy, syymax,
  Nsub, aspect, "Nodal stress sig-yy"];
ContourPlotNodeFuncOver2DMesh[NodeCoordinates, ElemNodeLists, sxy, sxymax,
  Nsub, aspect, "Nodal stress sig-xy"];
```

EJEMPLO PLACA CON UN ELEMENTO CUADRILATERO 4 NODOS

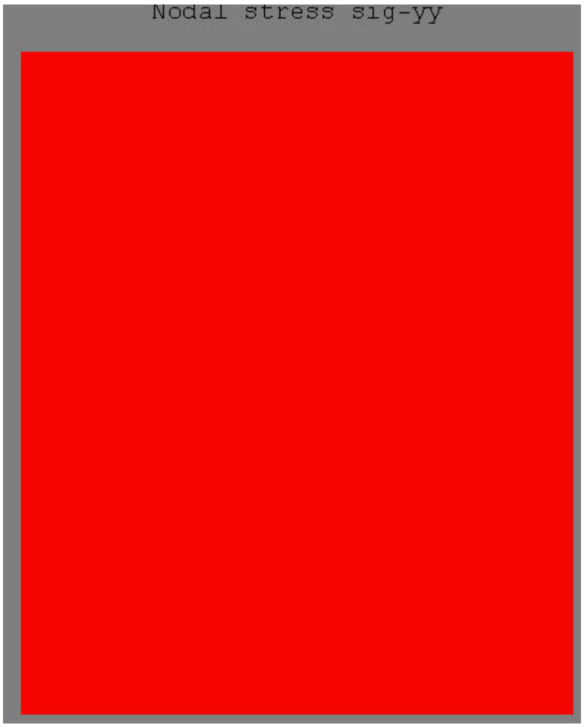
```
sig // MatrixForm
```

$$\begin{pmatrix} 0 & 10. & 0 \\ 0 & 10. & 0 \\ 0 & 10. & 0 \\ 0 & 10. & 0 \end{pmatrix}$$

SOLUCION TENSIONES NODALES PROMEDIADAS

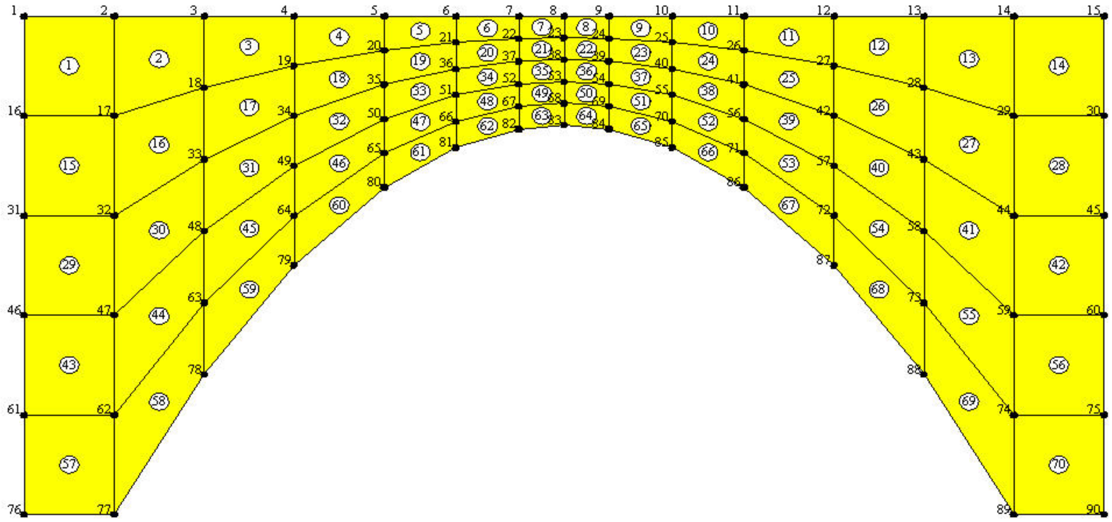
$$\sigma_{yy} = q, \sigma_{xx} = \sigma_{xy} = 0$$

SOLUCION EXACTA



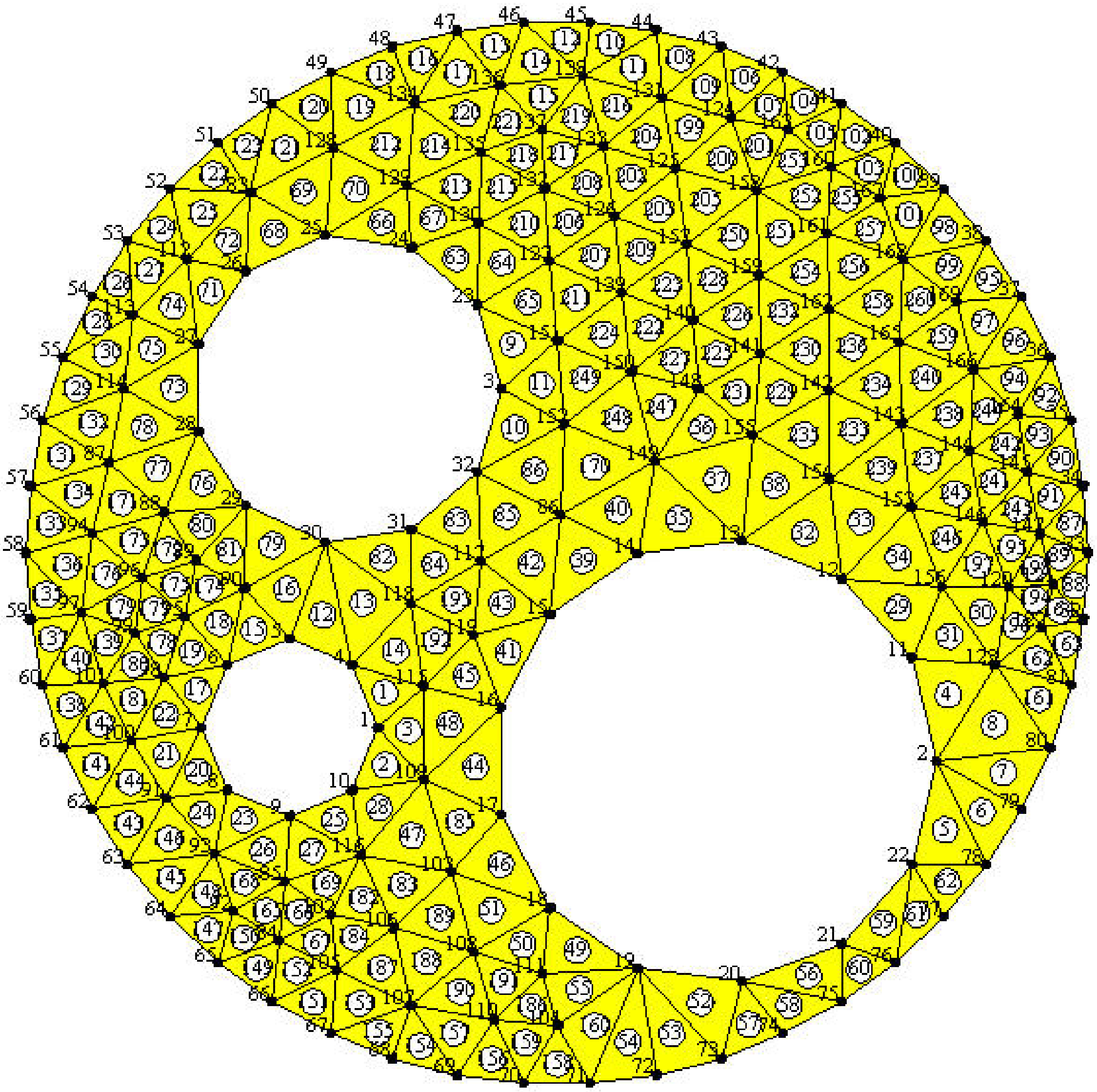
001 EJERCICIO 1 - ANIMACION CURSO 2004-5

Explicar cual es el problema que se plantea y resuelve en el ejemplo “animacion.nb”. Explicar los resultados. Modelarlo con ANSYS, y comparar los resultados.



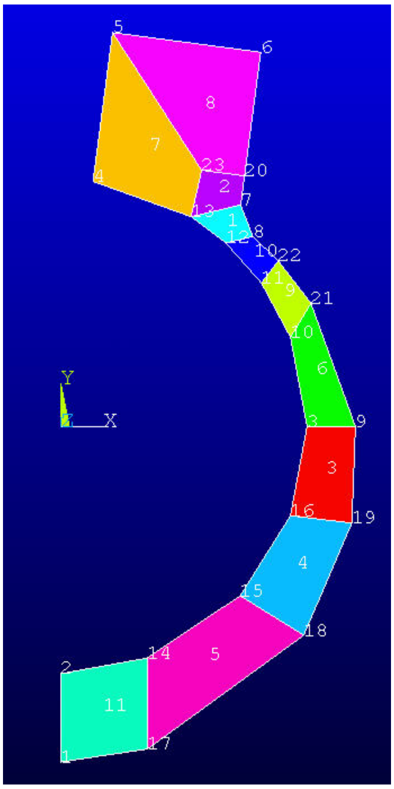
001 EJERCICIO 2 – IMPORTACION MALLA CURSO 2004-5

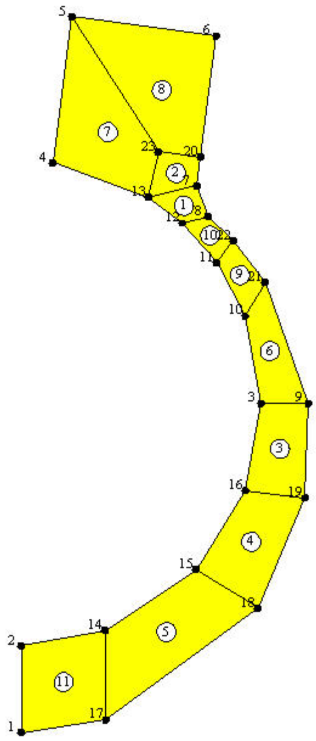
Explicar cual es el problema que se plantea y resuelve en el ejemplo “importacion.nb”. Explicar los resultados. Modelarlo con ANSYS, y comparar los resultados.

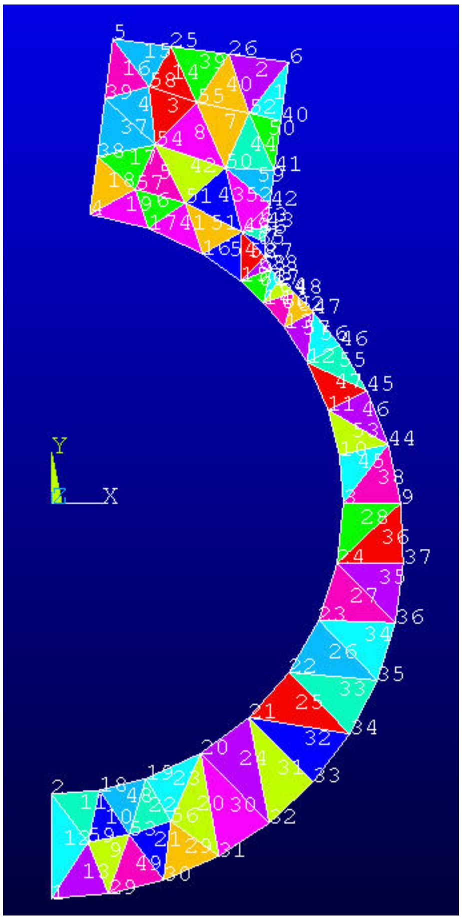


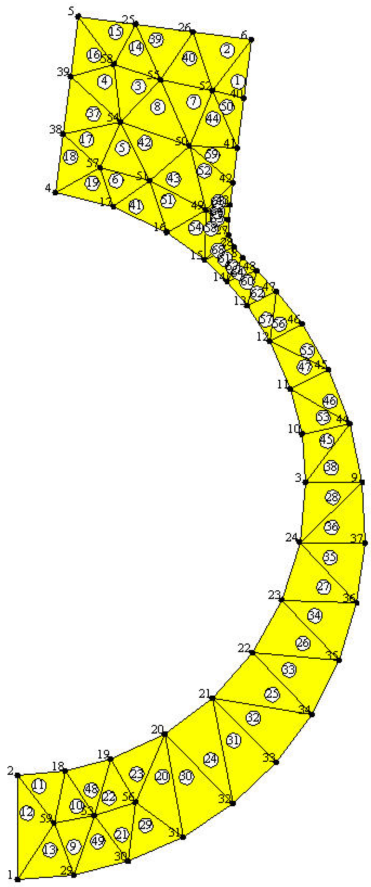
001 EJERCICIO 3 – IMPORTACION MALLA CURSO 2004-5

Utilizando el fichero “importa.nb”, importar varias discretizaciones del ejercicio de la biela bidimensional realizado en la prácticas con ANSYS. Utilizar todos los elementos que sea posible, dentro de los disponibles.









001 EJERCICIO 3 (CONT.) CURSO 2004-5