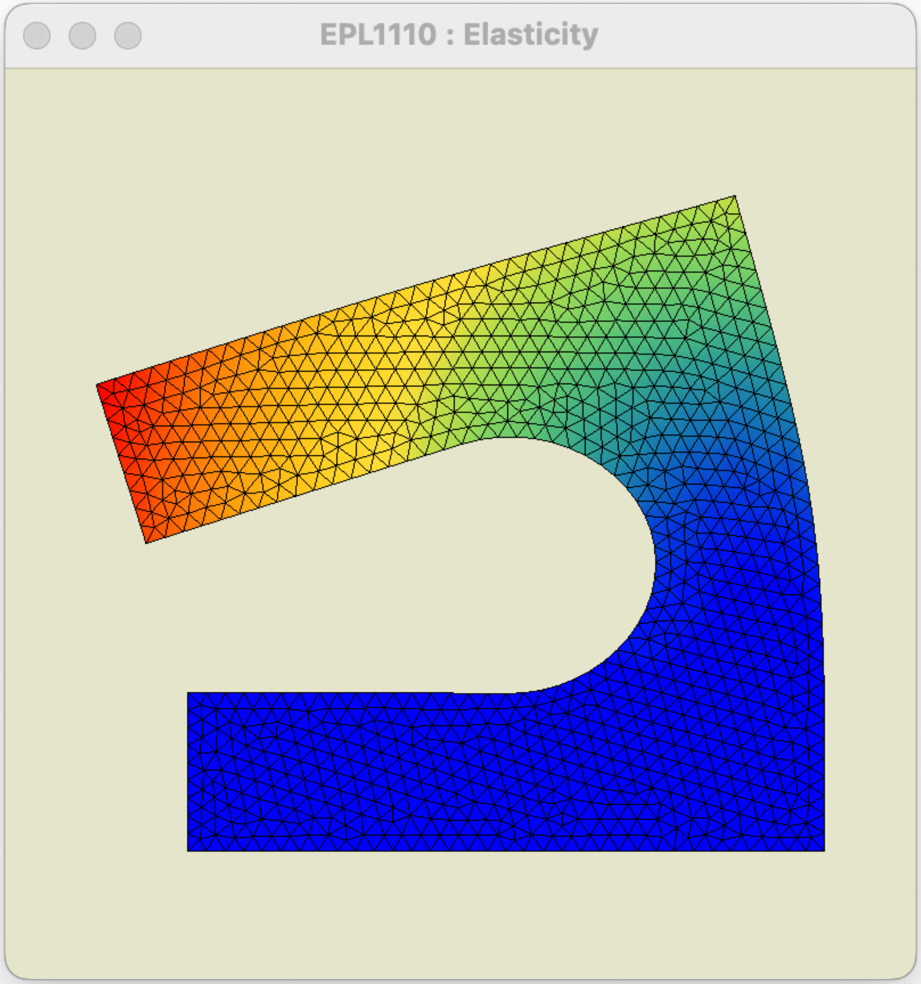
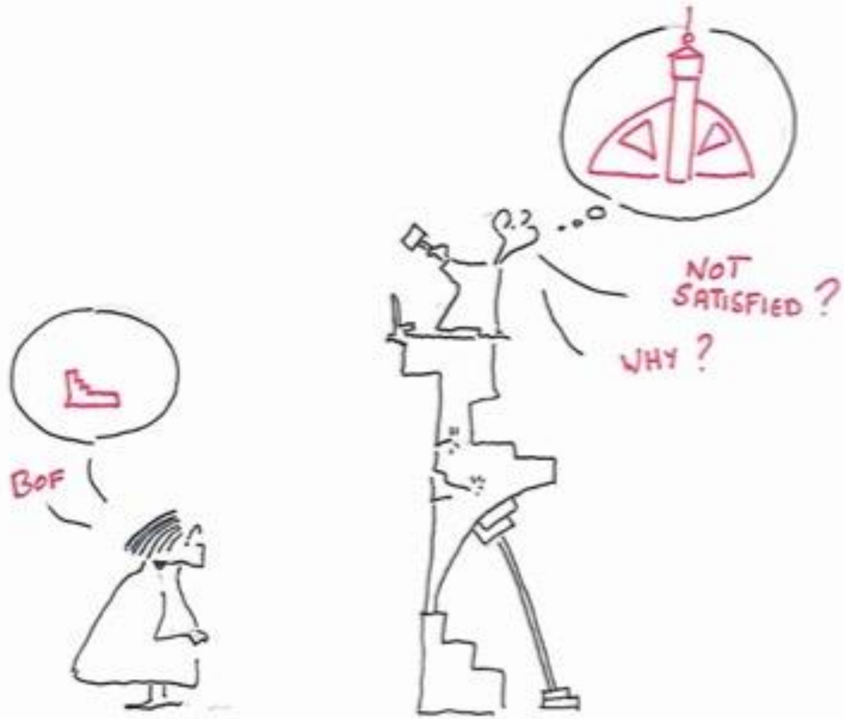
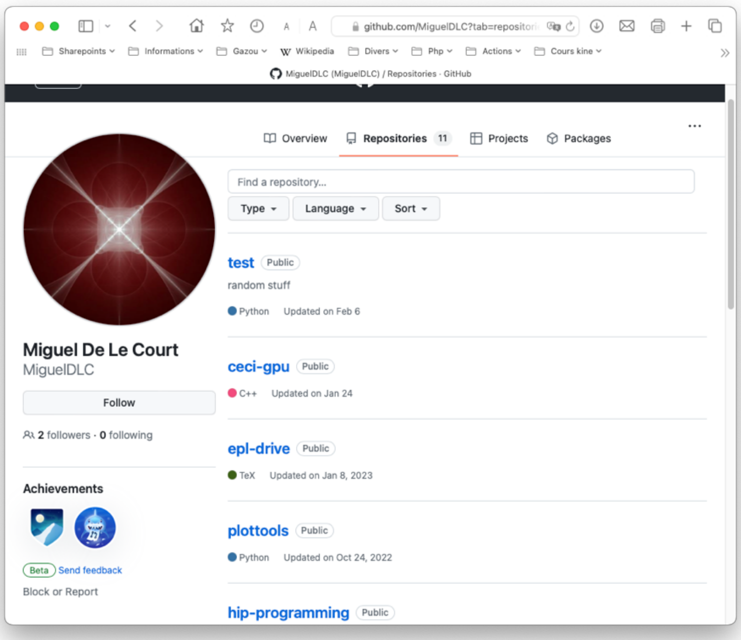
Projet 2024-25



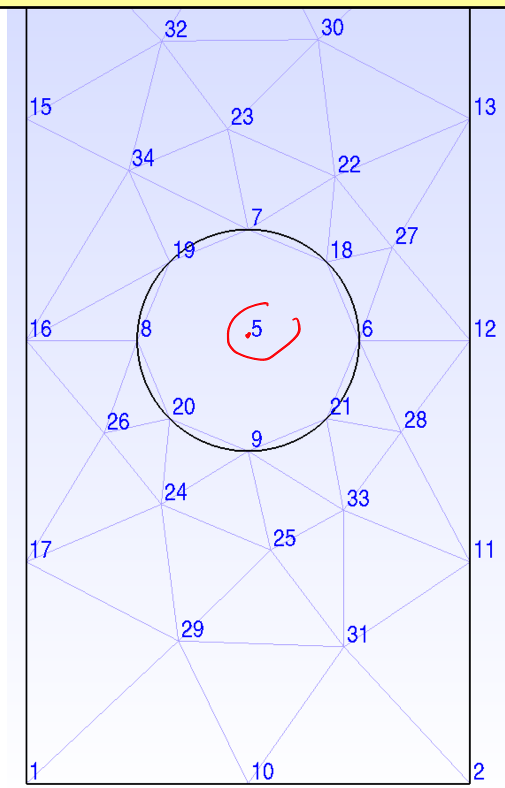


J

My mesh is bad !
La solution de Miguel



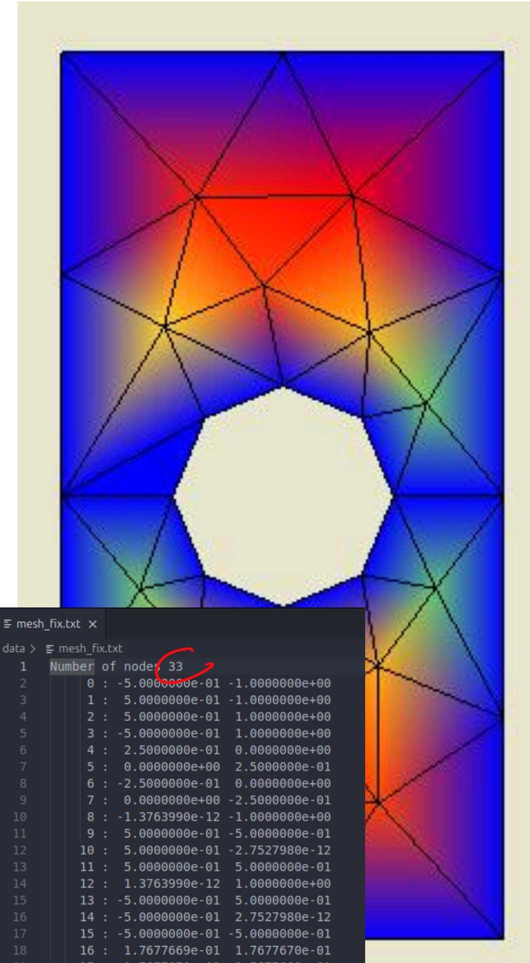
```python
class Mesh:
    nlocal: int
    nnodes: int
    nelem: int
    nedges: int
    nodes: NDArray[np.float64]
    elem: NDArray[np.int32]
    edges: NDArray[np.int32]
    domains: List[str]

    def __init__(self, path):
        self.path = path
```

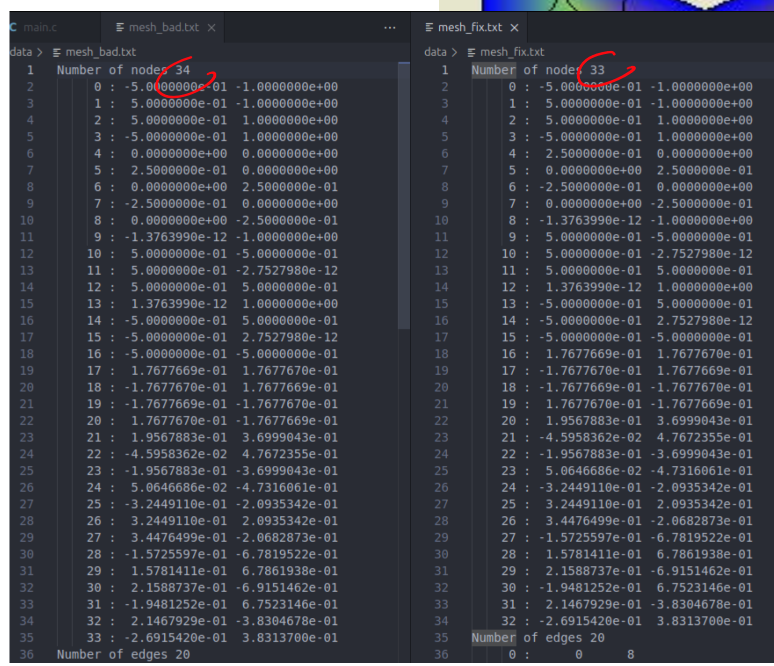


fixmesh.py

My mesh is broken !

My mesh est ugly !





main.c | mesh_bad.txt | mesh_fix.txt

data > mesh_bad.txt

```
Number of nodes 34
         0 : -5.0000000e-01 -1.0000000e+00
         1 :  5.0000000e-01 -1.0000000e+00
         2 :  5.0000000e-01  1.0000000e+00
         3 : -5.0000000e-01  1.0000000e+00
         4 :  0.0000000e+00  0.0000000e+00
         5 :  2.5000000e-01  0.0000000e+00
         6 :  0.0000000e+00  2.5000000e-01
         7 : -2.5000000e-01  0.0000000e+00
         8 :  0.0000000e+00 -2.5000000e-01
         9 : -1.3763990e-12 -1.0000000e+00
        10 :  5.0000000e-01 -5.0000000e-01
        11 :  5.0000000e-01 -2.7527980e-12
        12 :  5.0000000e-01  5.0000000e-01
        13 :  1.3763990e-12  1.0000000e+00
        14 : -5.0000000e-01  5.0000000e-01
        15 : -5.0000000e-01  2.7527980e-12
        16 : -5.0000000e-01 -5.0000000e-01
        17 :  1.7677669e-01  1.7677670e-01
        18 : -1.7677670e-01  1.7677669e-01
        19 : -1.7677669e-01 -1.7677670e-01
        20 :  1.7677670e-01 -1.7677669e-01
        21 :  1.9567883e-01  3.6999043e-01
        22 : -4.5958362e-02  4.7672355e-01
        23 : -1.9567883e-01 -3.6999043e-01
        24 :  5.0646686e-02 -4.7316061e-01
        25 : -3.2449110e-01 -2.0935342e-01
        26 :  3.2449110e-01  2.0935342e-01
        27 :  3.4476499e-01 -2.0682873e-01
        28 : -1.5725597e-01 -6.7819522e-01
        29 :  1.5781411e-01  6.7861938e-01
        30 :  2.1588737e-01 -6.9151462e-01
        31 : -1.9481252e-01  6.7523146e-01
        32 :  2.1467929e-01 -3.8304678e-01
        33 : -2.6915420e-01  3.8313700e-01
Number of edges 20
```

data > mesh_fix.txt

```
Number of nodes 33
         0 : -5.0000000e-01 -1.0000000e+00
         1 :  5.0000000e-01 -1.0000000e+00
         2 :  5.0000000e-01  1.0000000e+00
         3 : -5.0000000e-01  1.0000000e+00
         4 :  2.5000000e-01  0.0000000e+00
         5 :  0.0000000e+00  2.5000000e-01
         6 : -2.5000000e-01  0.0000000e+00
         7 :  0.0000000e+00 -2.5000000e-01
         8 : -1.3763990e-12 -1.0000000e+00
         9 :  5.0000000e-01 -5.0000000e-01
        10 :  5.0000000e-01 -2.7527980e-12
        11 :  5.0000000e-01  5.0000000e-01
        12 :  1.3763990e-12  1.0000000e+00
        13 : -5.0000000e-01  5.0000000e-01
        14 : -5.0000000e-01  2.7527980e-12
        15 : -5.0000000e-01 -5.0000000e-01
        16 :  1.7677669e-01  1.7677670e-01
        17 : -1.7677670e-01  1.7677669e-01
        18 : -1.7677669e-01 -1.7677670e-01
        19 :  1.7677670e-01 -1.7677669e-01
        20 :  1.9567883e-01  3.6999043e-01
        21 : -4.5958362e-02  4.7672355e-01
        22 : -1.9567883e-01 -3.6999043e-01
        23 :  5.0646686e-02 -4.7316061e-01
        24 : -3.2449110e-01 -2.0935342e-01
        25 :  3.2449110e-01  2.0935342e-01
        26 :  3.4476499e-01 -2.0682873e-01
        27 : -1.5725597e-01 -6.7819522e-01
        28 :  1.5781411e-01  6.7861938e-01
        29 :  2.1588737e-01 -6.9151462e-01
        30 : -1.9481252e-01  6.7523146e-01
        31 :  2.1467929e-01 -3.8304678e-01
        32 : -2.6915420e-01  3.8313700e-01
Number of edges 20
         0 :        0      8
```

Ecrire un code informatique efficace
pour l'élasticité linéaire 2D

Tensions planes et déformations planes
Triangles linéaires ou quadratiques
Quads bilinéaires ou biquadratiques
Problèmes axisymétriques
Conditions essentielles en xy et en normale/tangentielle
Conditions naturelles en xy et en normale/tangentielle
Obtenir les tensions dans le domaine

2 parties
dans le projet 2024-25 !

Définir un problème original !
Le résoudre avec votre code !
Analyser le résultat !

Ecrire un code informatique efficace pour l'élasticité linéaire plane

Tensions planes et déformations planes
Triangles linéaires ou quadratiques
Quads bilinéaires ou biquadratiques
Problèmes axisymétriques
Conditions essentielles en xy et en normale/tangentielle
Conditions naturelles en xy et en normale/tangentielle
Obtenir les tensions dans le domaine

2 parties dans le projet 2023-24 !

Définir un problème original !
Le résoudre avec votre code !
Analyser le résultat !

Définition du problème Soumission du texte d'une page Approbation finale par l'assistant de référence	 Vendredi 21 février Vendredi 28 février
Soumission du projet Soumission du code du projet	 Vendredi 28 mars - Vendredi 4 avril
Interview sur le projet Interview avec votre assistant de référence... et démonstration de votre programme	 Lundi 7 avril - Lundi 14 avril Mardi 8 avril - Mardi 15 avril
Soumission d'un code pour le grand prix de l'élément le plus fini Soumission du code optimisé	 Lundi 5 mai

Échéances !

Questions ?

Données du problème

mesh.txt

```
Number of nodes 335
     0 :  0.0000000e+00  1.0000000e+00
     1 :  0.0000000e+00  0.0000000e+00
     2 :  1.0000000e+00  1.0000000e+00
     3 :  1.0000000e+00  7.5000000e-01
     4 :  5.0000000e-01  7.5000000e-01
     5 :  5.0000000e-01  2.5000000e-01
     6 :  1.0000000e+00  2.5000000e-01
     7 :  1.0000000e+00  0.0000000e+00
     8 :  0.0000000e+00  9.5000000e-01
     9 :  0.0000000e+00  9.0000000e-01
```

problem.txt

```
Type of problem : planar strains
Young modulus   :  2.1100000e+11
Poisson ratio   :  3.0000000e-01
Mass density    :  7.8500000e+03
Gravity         :  9.8100000e+00
```

myFem

result.txt

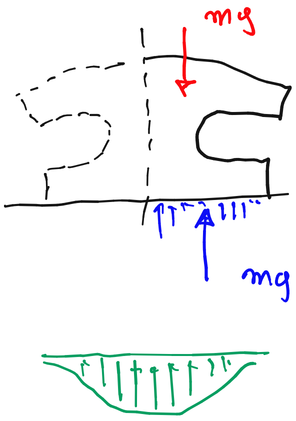
```
Elastic deformation 335
     0 :  0.0002330e-03  1.0000000e+00
     1 :  0.0000000e+00  0.0000000e+00
```

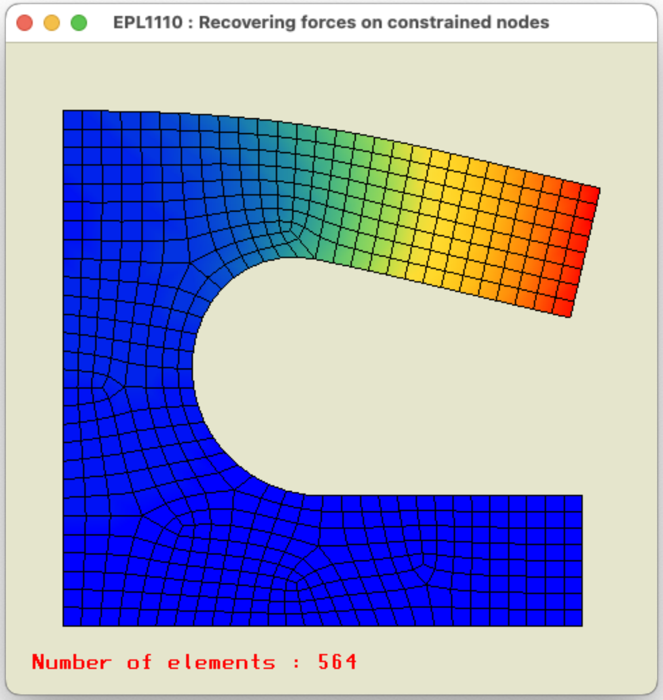
Si !

```
for (iElem = 0; iElem < theMesh->nElem; iElem++) {
    for (j=0; j < nLocal; j++) {
        map[j]  = theMesh->elem[iElem*nLocal+j];
        mapX[j] = 2*map[j];
        mapY[j] = 2*map[j] + 1;
        x[j]    = theNodes->X[map[j]];
        y[j]    = theNodes->Y[map[j]];}

    for (iInteg=0; iInteg < theRule->n; iInteg++) {
        double xsi    = theRule->xsi[iInteg];
        double eta    = theRule->eta[iInteg];
        double weight = theRule->weight[iInteg];
        femDiscretePhi2(theSpace,xsi,eta,phi);
        femDiscreteDphi2(theSpace,xsi,eta,dphidxsi,dphideta);

        double dxdxsi = 0.0;
        double dxdeta = 0.0;
        double dydxsi = 0.0;
        double dydeta = 0.0;
        for (i = 0; i < theSpace->n; i++) {
            dxdxsi += x[i]*dphidxsi[i];
            dxdeta += x[i]*dphideta[i];
            dydxsi += y[i]*dphidxsi[i];
            dydeta += y[i]*dphideta[i]; }
        double jac = fabs(dxdxsi * dydeta - dxdeta * dydxsi);

        for (i = 0; i < theSpace->n; i++) {
            dphidx[i] = (dphidxsi[i] * dydeta - dphideta[i] * dydxsi) / jac;
            dphidy[i] = (dphideta[i] * dxdxsi - dphidxsi[i] * dxdeta) / jac; }
        for (i = 0; i < theSpace->n; i++) {
            for(j = 0; j < theSpace->n; j++) {
                A[mapX[i]][mapX[j]] += (dphidx[i] * a * dphidx[j] +
                                        dphidy[i] * c * dphidy[j]) * jac * weight;
                A[mapX[i]][mapY[j]] += (dphidx[i] * b * dphidy[j] +
                                        dphidy[i] * c * dphidx[j]) * jac * weight;
                A[mapY[i]][mapX[j]] += (dphidy[i] * b * dphidx[j] +
                                        dphidx[i] * c * dphidy[j]) * jac * weight;
                A[mapY[i]][mapY[j]] += (dphidy[i] * a * dphidy[j] +
                                        dphidx[i] * c * dphidx[j]) * jac * weight; }}
         for (i = 0; i < theSpace->n; i++) {
            B[mapY[i]] -= phi[i] * g * rho * jac * weight; }}}

int *theConstrainedNodes = theProblem->constrainedNodes;
for (int i=0; i < theSystem->size; i++) {
    if (theConstrainedNodes[i] != -1) {
        double value = theProblem->conditions[theConstrainedNodes[i]]->value;
        femFullSystemConstrain(theSystem,i,value); }}

return femFullSystemEliminate(theSystem);
```

Le prochain devoir !

Appliquer des conditions de Neumann !
Retrouver les forces de réactions !
Intégrer la densité de force de gravité !
Vérifier la seconde et la troisième loi de Newton !



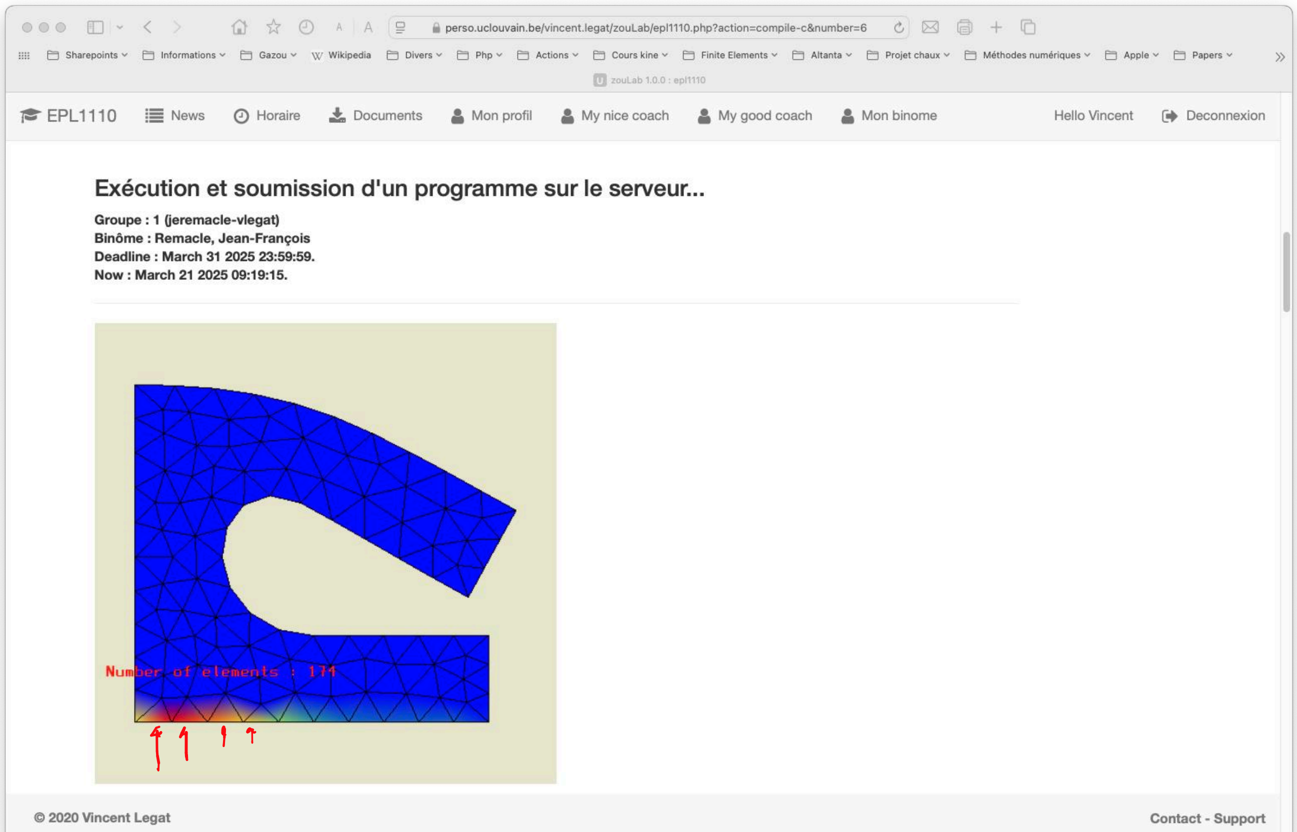



```
void femElasticityAssembleElements(femProblem *theProblem)
void femElasticityAssembleEdges(femProblem *theProblem)
void femElasticitySolve(femProblem *theProblem)
void femElasticityForces(femProblem *theProblem)
void femElasticityIntegrate(femProblem *theProblem, double (*f)(double,double))
```

Le prochain devoir !

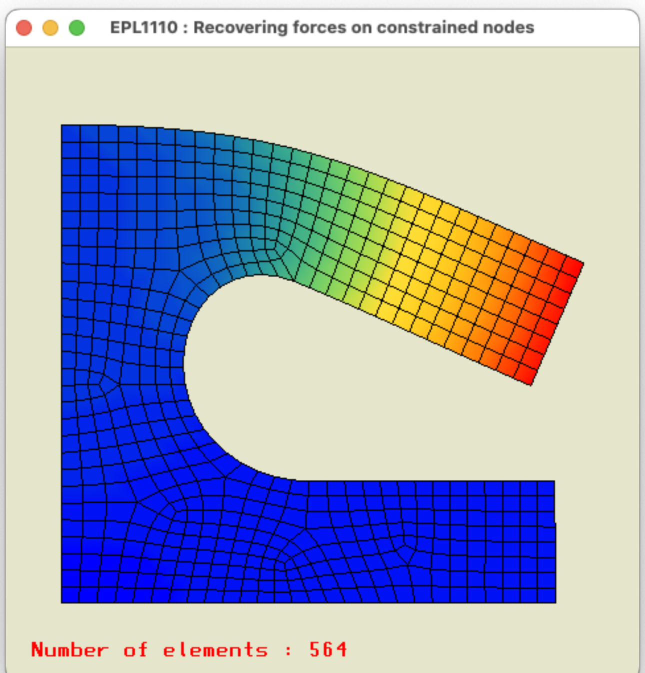
New deadline : 31 mars 2025 !

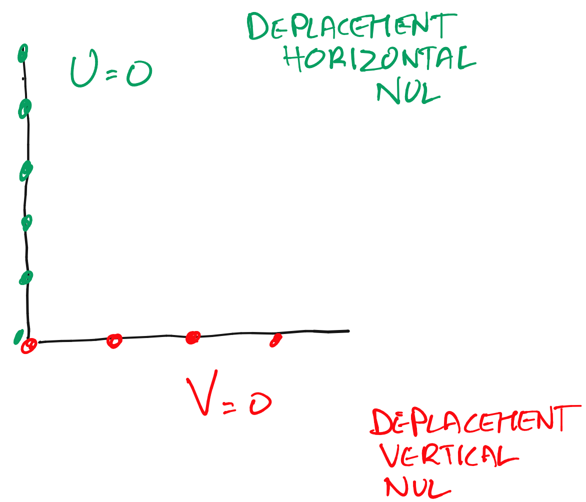
Ce devoir n'est pas nécessaire -stricto sensu- pour le projet !



Conditions aux limites

```
Linear elasticity problem
   Young modulus    E   =  2.1100000e+11 [N/m2]
   Poisson's ratio nu   =  3.0000000e-01 [-]
   Density         rho  =  7.8500000e+03 [kg/m3]
   Gravity          g   =  9.8100000e+00 [m/s2]
   Planar strains formulation
   Boundary conditions :
              Symmetry : imposing  0.00e+00 as the horizontal displacement
                Bottom : imposing  0.00e+00 as the vertical displacement
```



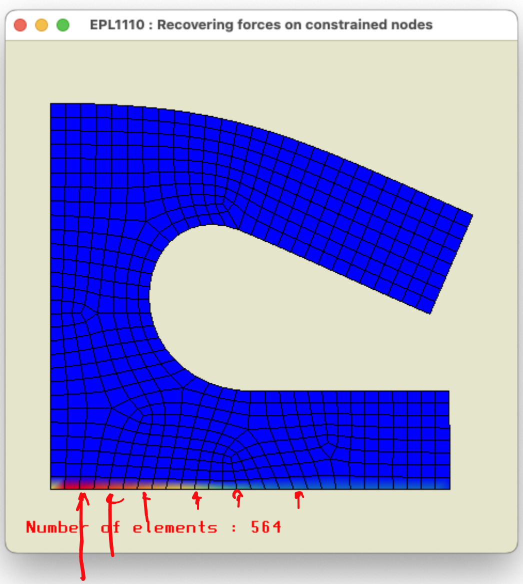


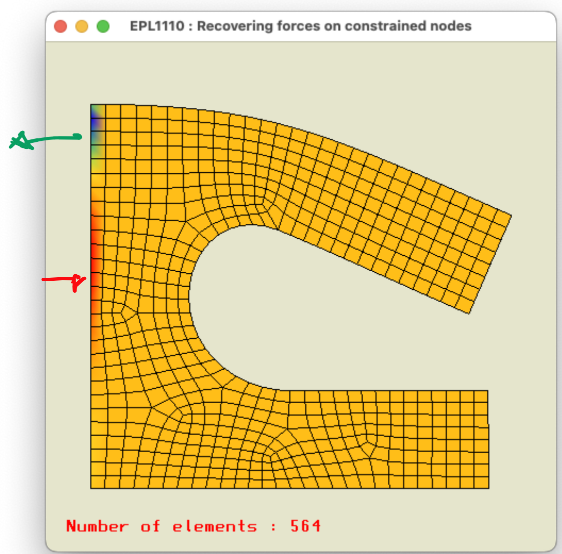
Forces appliquées sur les nœuds contraints

```
Linear elasticity problem
Boundary conditions :
              Symmetry : imposing  0.00e+00 as the horizontal displacement
                Bottom : imposing  0.00e+00 as the vertical displacement
                   Top : imposing -1.00e+04 as the vertical force density
```

```
==== Minimum displacement           :  0.0000000e+00 [m]
==== Maximum displacement           :  1.6227532e-06 [m]
==== Global horizontal force        :  2.0317257e-10 [N]
==== Global vertical force          :  1.0000000e+04 [N]
==== Weight                         :  0.0000000e+00 [N]
```

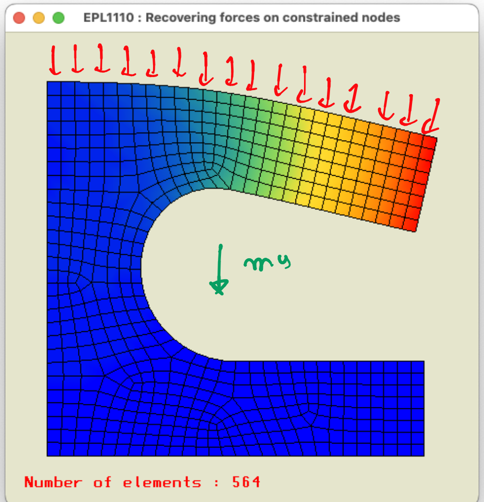
RESIDUS
EQUILIBRE EN Y



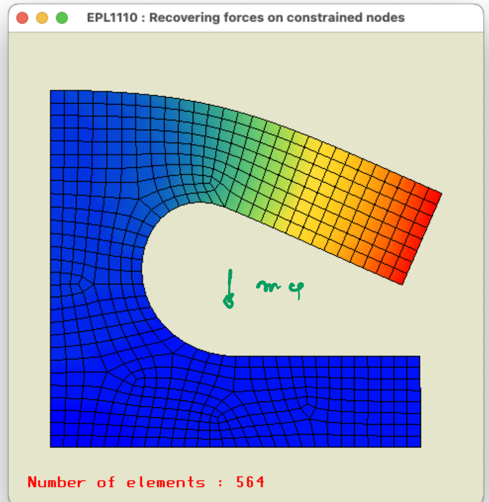


RESIDUS
DES EQUILIBRES EN X

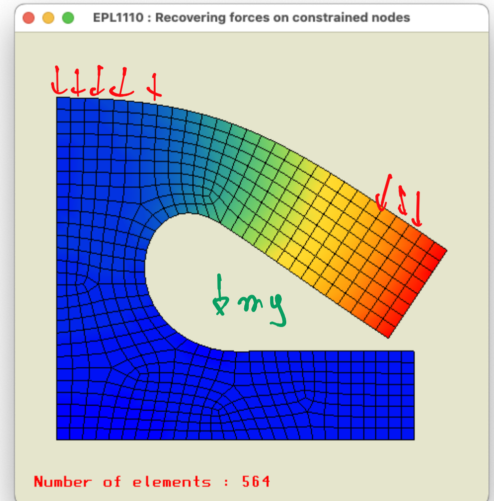
Modifions le problème



Déformation sous la charge



Déformation sous son poids propre

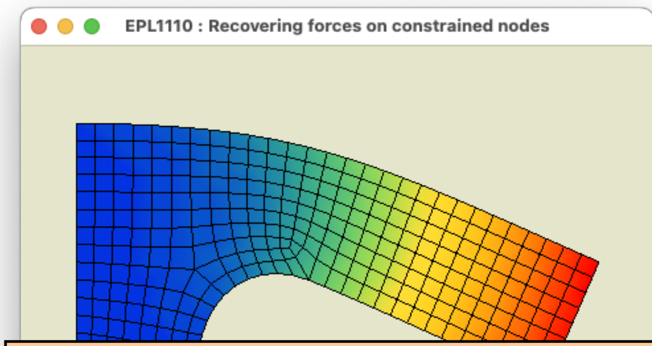


Déformation sous la charge et son poids propre

```
double E   = 211.e9;
    double nu  = 0.3;
    double rho = 7.85e3;
    double g   = 0.00;
    femProblem* theProblem = femElasticityCreate(theGeometry,E,nu,rho,g,PLANAR_STRAIN);
    femElasticityAddBoundaryCondition(theProblem,"Symmetry",DIRICHLET_X,0.0);
    femElasticityAddBoundaryCondition(theProblem,"Bottom",DIRICHLET_Y,0.0);
    femElasticityAddBoundaryCondition(theProblem,"Top",NEUMANN_Y,-1e4);
    femElasticityPrint(theProblem);
```

Modifions le problème

```
Linear elasticity problem
   Young modulus    E   =  2.1100000e+11 [N/m2]
   Poisson's ratio nu   =  3.0000000e-01 [-]
   Density         rho  =  7.8500000e+03 [kg/m3]
   Gravity          g   =  9.8100000e+00 [m/s2]
   Planar strains formulation
   Boundary conditions :
              Symmetry : imposing  0.00e+00 as the horizontal displacement
                Bottom : imposing  0.00e+00 as the vertical displacement
```



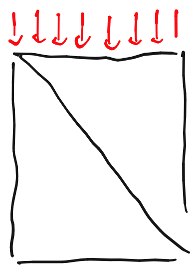

```
double E   = 211.e9;
    double nu  = 0.3;
    double rho = 7.85e3;
    double g   = 0.00;
    femProblem* theProblem = femElasticityCreate(theGeometry,E,nu,rho,g,PLANAR_STRAIN);
    femElasticityAddBoundaryCondition(theProblem,"Symmetry",DIRICHLET_X,0.0);
    femElasticityAddBoundaryCondition(theProblem,"Bottom",DIRICHLET_Y,0.0);
    femElasticityAddBoundaryCondition(theProblem,"Top",NEUMANN_Y,-1e4);
    femElasticityPrint(theProblem);
```

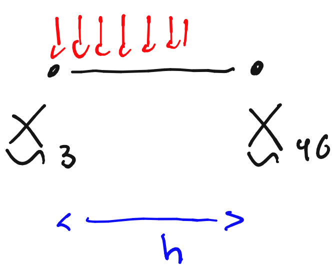
Conditions de Neumann !

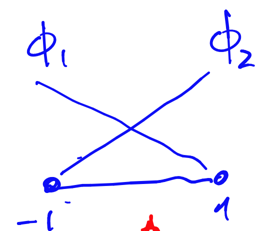
$$g = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} -10^4 \\ 0 \end{bmatrix}$$

$$\nabla^2 u = 0$$

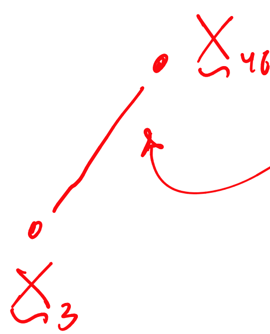
$$\sum_j \langle \nabla \tau_j \cdot \nabla \tau_i \rangle U_j = \langle\langle g \, \tau_i \rangle\rangle_N$$





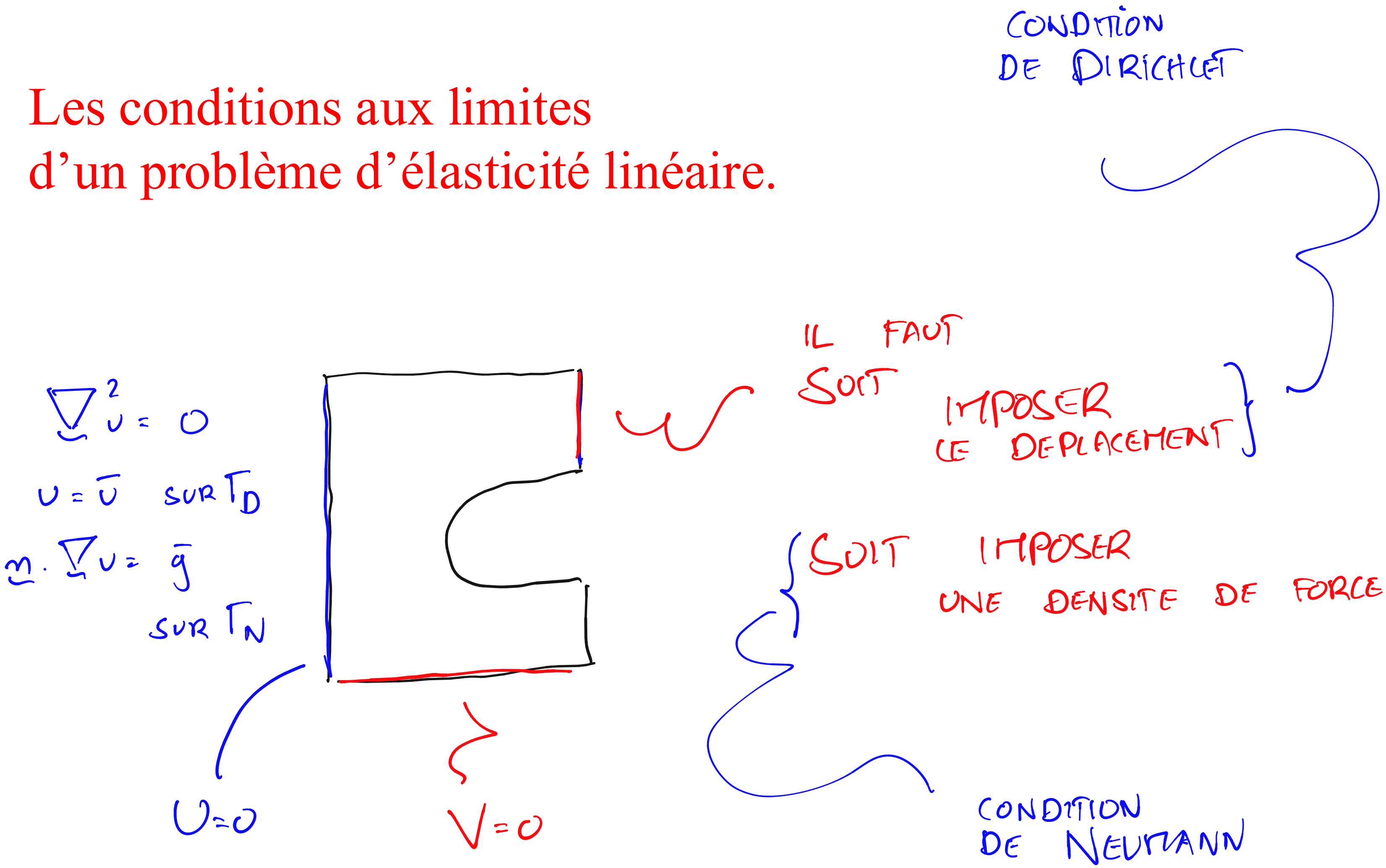


$$\phi_1 = (1-\xi)/2$$
$$\phi_2 = (1+\xi)/2$$
$$J = \frac{h}{2}$$

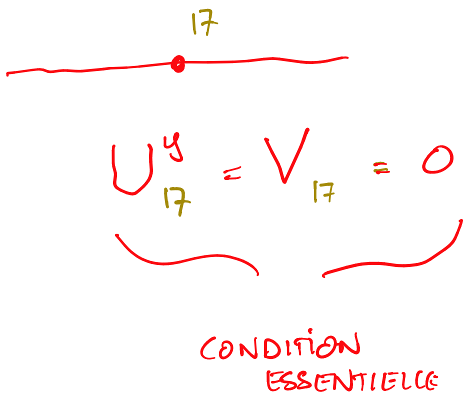


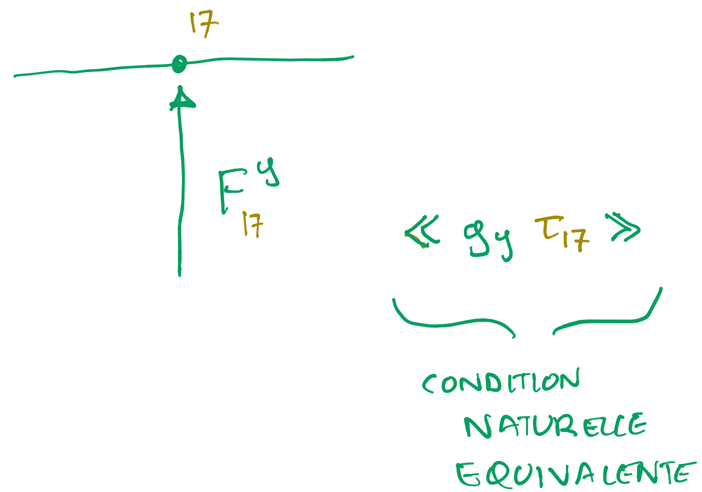
$$h = \sqrt{(X_{46}-X_3)^2 + (Y_{46}-Y_3)^2}$$

Les conditions aux limites d'un problème d'élasticité linéaire.

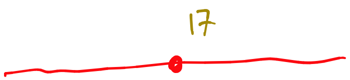


On peut appliquer une condition de Dirichlet ou…
une condition de Neumann équivalente.



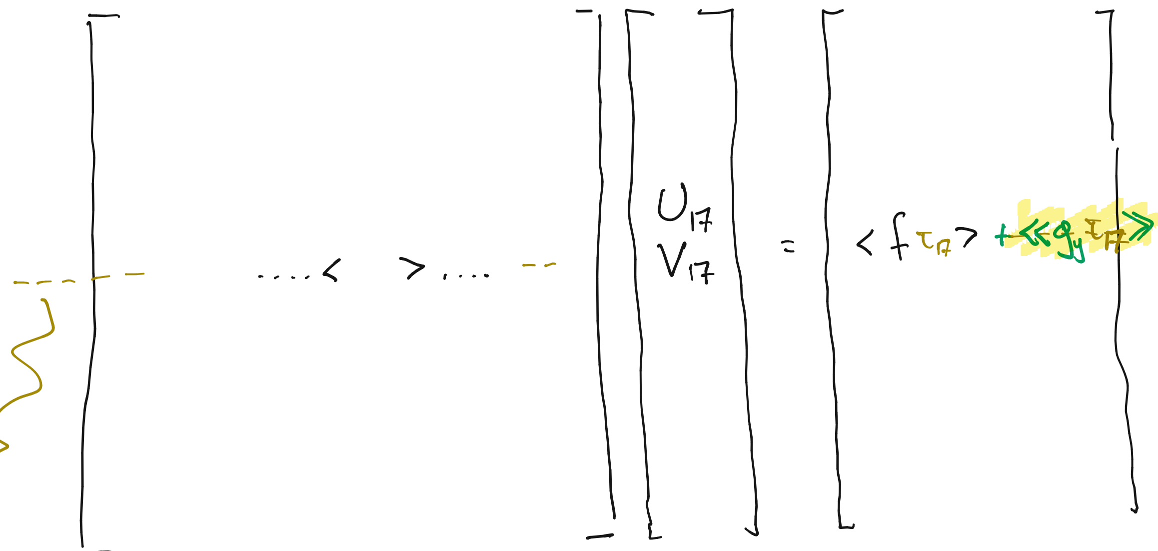


Retrouver les forces de contact sans rien calculer a posteriori !



$$U^y_{17} = V_{17} = 0$$

CONDITION ESSENTIELLE



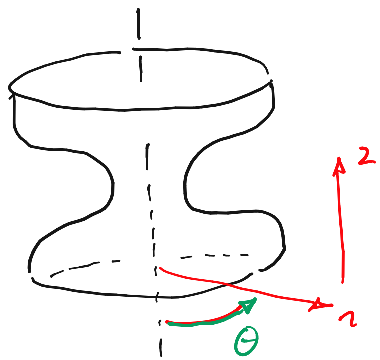
$17 \times 2 + 1 = 35$

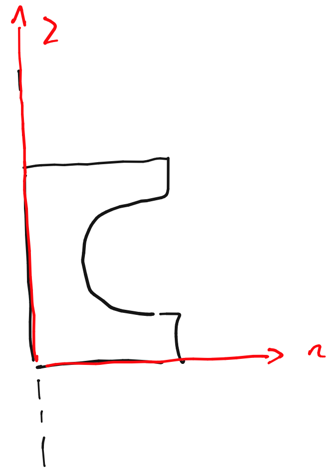
ON SUPPRIME LA LIGNE

RESIDU

$$\sum A_{17j} U_j - B_{17} = R_{17} \neq 0$$

Et l'axisymétrique ?





$$\underset{\sim}{u} = \begin{bmatrix} u_x(x,y) \\ u_y(x,y) \end{bmatrix}$$

EXPRESSION EN COORDONNEES CARTESIENNES

$$\nabla \underset{\sim}{u}$$

EXPRESSION EN COORDONNEES CYLINDRIQUES

$$\underset{\sim}{u} = \begin{bmatrix} u_r \\ u_\theta \\ u_z \end{bmatrix} = \begin{bmatrix} u_r(r,z) \\ 0 \\ u_z(r,z) \end{bmatrix}$$

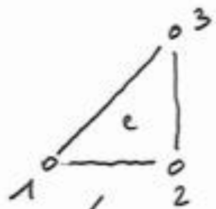
$$u_\theta = 0$$

$$\frac{\partial}{\partial \theta} = 0$$

COMPUTING A LOCAL ELASTICITY MATRIX

$$
\begin{bmatrix}
A\langle \tau_{i,x}\,\tau_{j,x}\rangle + C\langle \tau_{i,y}\,\tau_{j,y}\rangle & B\langle \tau_{i,x}\,\tau_{j,y}\rangle + C\langle \tau_{i,y}\,\tau_{j,x}\rangle \\
B\langle \tau_{i,y}\,\tau_{j,x}\rangle + C\langle \tau_{i,x}\,\tau_{j,y}\rangle & A\langle \tau_{i,y}\,\tau_{j,y}\rangle + C\langle \tau_{i,x}\,\tau_{j,x}\rangle
\end{bmatrix}
$$

$\underline{\underline{A}}^e_{11}$: $A_{xx11} = A$, $A_{yy11} = C$, $A_{xy11} = 0$, $A_{yx11} = 0$



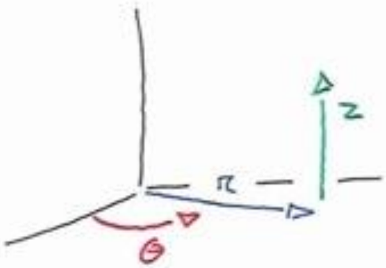
	$\tau_{i,x}$	$\tau_{i,y}$
1	-1	0
2	1	-1
3	0	1

$$
\underline{\underline{A}}^e = \begin{bmatrix}
A & 0 & -A & B & 0 & -B \\
0 & C & C & -C & -C & 0 \\
 & & A+C & -B-C & -C & B \\
 & & -B-C & A+C & C & -A \\
 & & & & C & 0 \\
 & & & & 0 & A
\end{bmatrix} \frac{1}{2}
$$

AXISYMMETRIC PROBLEMS

AXISYMMETRY !

$$u_\theta = 0, \qquad \frac{\partial}{\partial \theta} = 0$$



$$\underline{u} = \begin{bmatrix} u_r \\ u_\theta \\ u_z \end{bmatrix} = \begin{bmatrix} u(r,z) \\ 0 \\ v(r,z) \end{bmatrix}$$

$$\underline{\underline{\varepsilon}} = \begin{bmatrix} u_{r,r} & 0 & (u_{r,z} + u_{z,r})/2 \\ & u_r/r & 0 \\ & & u_{z,z} \end{bmatrix}$$

A, B, C OF PLANAR DEFORMATIONS !

$$\underline{\underline{\sigma}} = \begin{bmatrix} A\varepsilon_{rr} + B(\varepsilon_{\theta\theta} + \varepsilon_{zz}) & 0 & 2C\varepsilon_{rz} \\ & A\varepsilon_{\theta\theta} + B(\varepsilon_{rr} + \varepsilon_{zz}) & 0 \\ & & A\varepsilon_{zz} + B(\varepsilon_{rr} + \varepsilon_{\theta\theta}) \end{bmatrix}$$

$$\underline{\underline{\varepsilon}}\left(\begin{bmatrix}\tau_i & 0 & 0\end{bmatrix}\right) = \begin{bmatrix}\tau_{i,r} & 0 & \tau_{i,z}/2 \\ & \tau_i/r & 0 \\ & & 0\end{bmatrix}$$

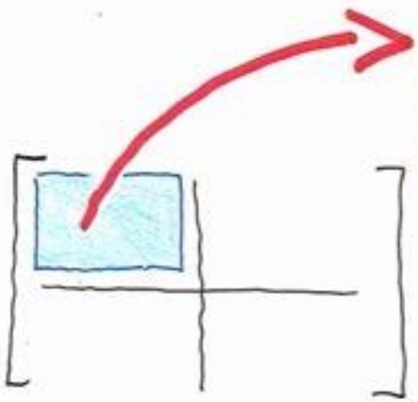
$$\underline{\underline{\varepsilon}}\left(\begin{bmatrix}0 & 0 & \tau_i\end{bmatrix}\right) = \begin{bmatrix}0 & 0 & \tau_{i,r}/2 \\ & 0 & 0 \\ & & \tau_{i,z}\end{bmatrix}$$

$$2\pi \left[\begin{array}{c|c} \left\langle r\, \underline{\underline{\varepsilon}}\begin{pmatrix}\tau_i\\0\\0\end{pmatrix} : \underline{\underline{\sigma}}\begin{pmatrix}\tau_j\\0\\0\end{pmatrix}\right\rangle & \left\langle r\, \underline{\underline{\varepsilon}}\begin{pmatrix}\tau_i\\0\\0\end{pmatrix} : \underline{\underline{\sigma}}\begin{pmatrix}0\\0\\\tau_j\end{pmatrix}\right\rangle \\ \hline \left\langle r\, \underline{\underline{\varepsilon}}\begin{pmatrix}0\\0\\\tau_i\end{pmatrix} : \underline{\underline{\sigma}}\begin{pmatrix}\tau_j\\0\\0\end{pmatrix}\right\rangle & \left\langle r\, \underline{\underline{\varepsilon}}\begin{pmatrix}0\\0\\\tau_i\end{pmatrix} : \underline{\underline{\sigma}}\begin{pmatrix}0\\0\\\tau_j\end{pmatrix}\right\rangle \end{array}\right]$$

$$\underline{\underline{A}}_{ij}$$

$$\underline{\underline{\varepsilon}}\begin{pmatrix}\tau_i\\0\\0\end{pmatrix} = \begin{bmatrix}\tau_{i,r} & 0 & \tau_{i,z}/2\\ & \tau_i/r & 0\\ & & 0\end{bmatrix}$$

$$\underline{\underline{\sigma}}\begin{pmatrix}\tau_j\\0\\0\end{pmatrix} = \begin{bmatrix}A\tau_{j,r}+B\tau_j/r & 0 & C\tau_{j,z}\\ & A\tau_j/r + B\tau_{j,r} & 0\\ & & B\tau_j/r + B\tau_{j,r}\end{bmatrix}$$



$\underline{\underline{A}}_{ij}$

$$\underline{\underline{\varepsilon}}\begin{pmatrix}\tau_i\\0\\0\end{pmatrix} : \underline{\underline{\sigma}}\begin{pmatrix}\tau_j\\0\\0\end{pmatrix} = \left\langle \tau_{i,r}\left(A\tau_{j,r} + B\tau_j/r\right) r \right\rangle$$

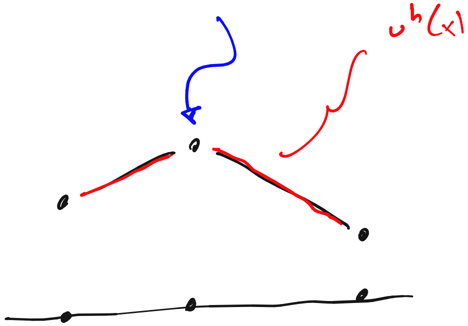
$$+ \left\langle \tau_{i,z}\, C\, \tau_{j,z}\, r \right\rangle$$

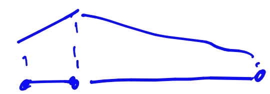
$$+ \left\langle \frac{\tau_i}{r}\left(A\frac{\tau_j}{r} + B\tau_{j,r}\right) r \right\rangle$$

IT IS ALMOST LIKE A 2D PROBLEM !

$$
\left[
\begin{array}{c|c}
\begin{aligned}
& A \langle \tau_{i,r}\, \tau_{j,r}\, r \rangle \\
& + C \langle \tau_{i,z}\, \tau_{j,z}\, r \rangle \\
& + B \langle \tau_{i,r}\, \tau_j \rangle \\
& + \langle \underbrace{\tau_i}_{\varepsilon_{\theta\theta}/r} \underbrace{\left(B\, \tau_{j,r} + A \frac{\tau_j}{r} \right.}_{\sigma_{\theta\theta}} \rangle
\end{aligned}
&
\begin{aligned}
& B \langle \tau_{i,r}\, \tau_{j,z}\, r \rangle \\
& + C \langle \tau_{i,z}\, \tau_{j,r}\, r \rangle \\
& + B \langle \tau_i\, \tau_{j,z} \rangle
\end{aligned}
\\ \hline
\begin{aligned}
& B \langle \tau_{i,z}\, \tau_{j,r}\, r \rangle \\
& + C \langle \tau_{i,r}\, \tau_{j,z}\, r \rangle \\
& + B \langle \tau_{i,z}\, \tau_j \rangle
\end{aligned}
&
\begin{aligned}
& A \langle \tau_{i,z}\, \tau_{j,z}\, r \rangle \\
& + C \langle \tau_{i,r}\, \tau_{j,r}\, r \rangle
\end{aligned}
\end{array}
\right]
$$

Et le calcul des tensions ?





$$\sigma^h = A \frac{\partial u^h}{\partial x}(x)$$

$$\underbrace{\langle \hat{\underline{\underline{\varepsilon}}}^h : \underline{\underline{\underline{\underline{C}}}} : \underline{\underline{\varepsilon}}^h \rangle}_{a(\hat{u}^h,\, u^h)} = \langle \hat{u} f \rangle$$

$J(\underline{u}^h)$ = MINIMUM OF THE POTENTIAL ENERGY

POTENTIAL ENERGY = DEFORMATION ENERGY − WORK OF EXTERNAL FORCES

→ $\underline{u}^h$ MINIMIZES THE ENERGY

DISPLACEMENTS FORMULATION

BUT $\underline{\underline{\sigma}}(\underline{u}^h)$

- DISCONTINUOUS! → SEVERAL VALUES AT EACH VERTEX!
- TYPICALLY CONTINUOUS LINEAR OR QUADRATIC

MUST BE VIEWED AS A LEAST-SQUARE FIT OF THE EXACT SOLUTION $\underline{\underline{\sigma}}(\underline{u})$.

→ ALLOWS US TO USE THE BEST STRATEGY FOR THE INTERPRETATION OF $\underline{\underline{\sigma}}(\underline{u}^h)$

$\underline{\underline{\sigma}}(\underline{u}^h)$ = SOLUTION OF A LEAST-SQUARES FIT PROBLEM

$a(\hat{\underline{u}}, \underline{u}) = b(\hat{\underline{u}})$
$\forall \hat{\underline{u}} \in \hat{\mathcal{U}}$

$a(\hat{\underline{u}}^h, \underline{u}^h) = b(\hat{\underline{u}}^h)$
$\forall \underline{u}^h \in \hat{\mathcal{U}}^h$

cfr $\hat{\mathcal{U}}^h \subset \hat{\mathcal{U}}$!

CFR LINEARITY !

? $\underline{u}^h$

$$a(\hat{\underline{u}}^h, \underline{u}^h - \underline{u}) = 0 \quad \forall \hat{\underline{u}}^h \in \hat{\mathcal{U}}^h$$

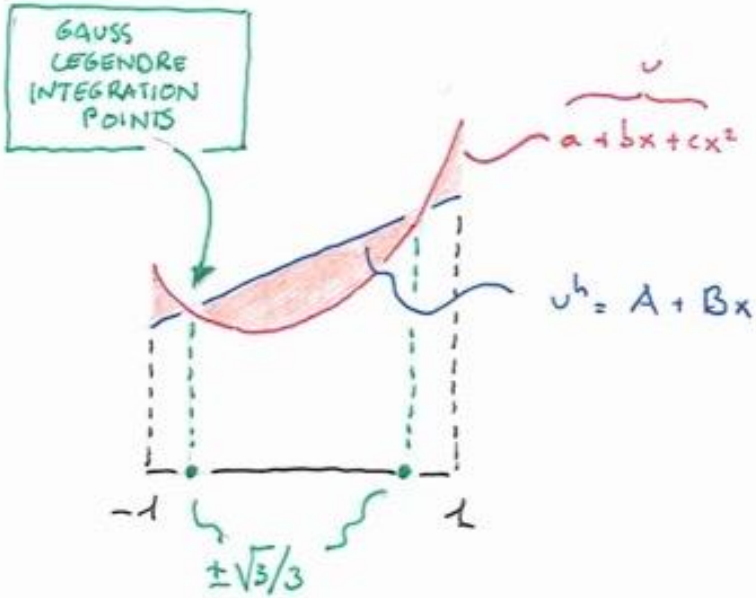
$\Updownarrow$

$\hat{b}(\hat{\underline{u}}^h) = a(\hat{\underline{u}}^h, \underline{u})$
$\hat{a}(\hat{\underline{u}}^h, \underline{u}^h) = a(\hat{\underline{u}}^h, \underline{u}^h)$

? $\underline{u}^h$

$$G(\underline{u}^h) = \min_{\underline{v}^h \in \mathcal{U}^h} \underbrace{\frac{1}{2} \left\langle \left(\underline{\underline{\varepsilon}}(\underline{v}^h) - \underline{\underline{\varepsilon}}(\underline{u})\right) : \underline{\underline{\underline{\underline{C}}}} : \left(\underline{\underline{\varepsilon}}(\underline{v}^h) - \underline{\underline{\varepsilon}}(\underline{v})\right) \right\rangle}_{G(\underline{v}^h)}$$

WHAT ARE THE "BEST VALUES" OF A LEAST-SQUARES POLYNOMIAL FIT ?



? A, B SUCH THAT

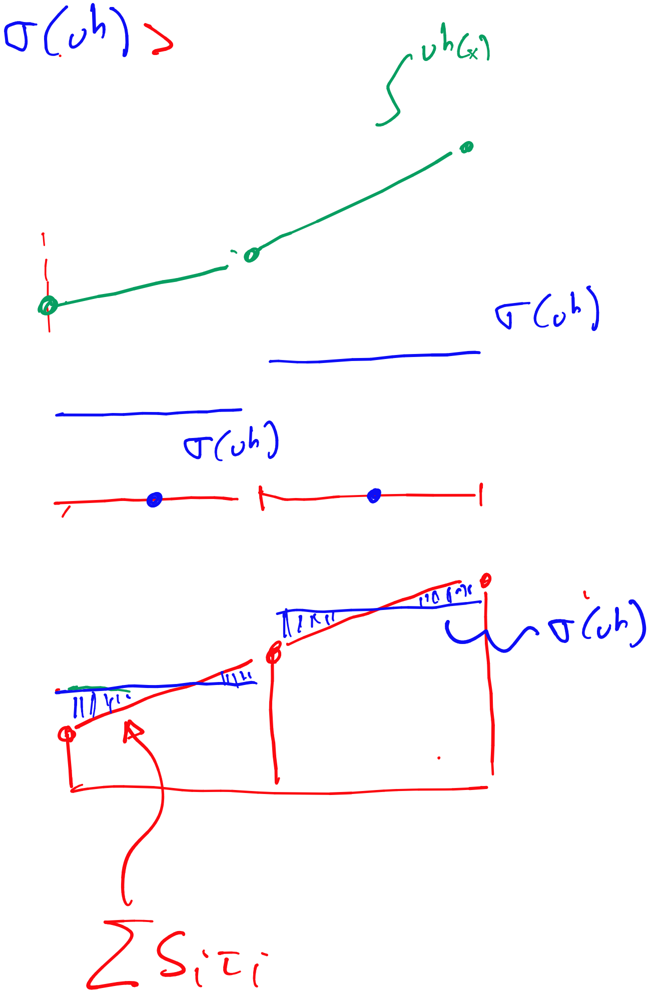
$$G(A,B) = \min_{\hat{A}\hat{B}} \int_{-1}^{1} (\hat{A} + \hat{B}x - a - bx - cx^2)^2 \, dx$$

$$\begin{cases} \int_{-1}^{1} (A + Bx - a - bx - cx^2)\, x \, dx = 0 \\ \int_{-1}^{1} (A + Bx - a - bx - cx^2) \, dx = 0 \end{cases}$$

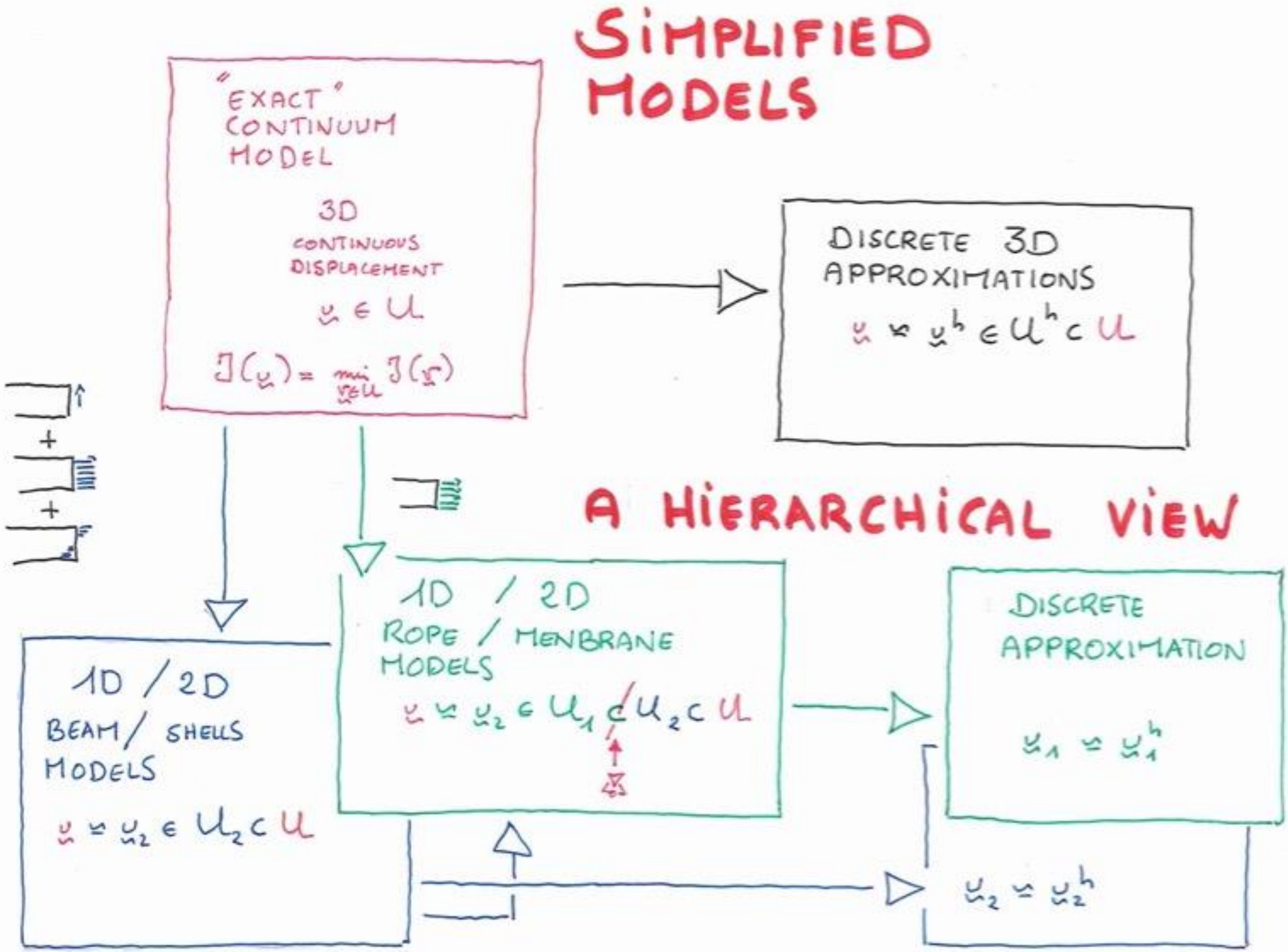
$$\begin{cases} B\frac{2}{3} - b\frac{2}{3} = 0 & \longrightarrow B = b \\ 2A - 2a - c\frac{2}{3} = 0 & \longrightarrow A = a + \frac{c}{3} \end{cases}$$

$$\frac{c}{3} - cx^2 = 0 \rightarrow x = \pm\frac{\sqrt{3}}{3}$$

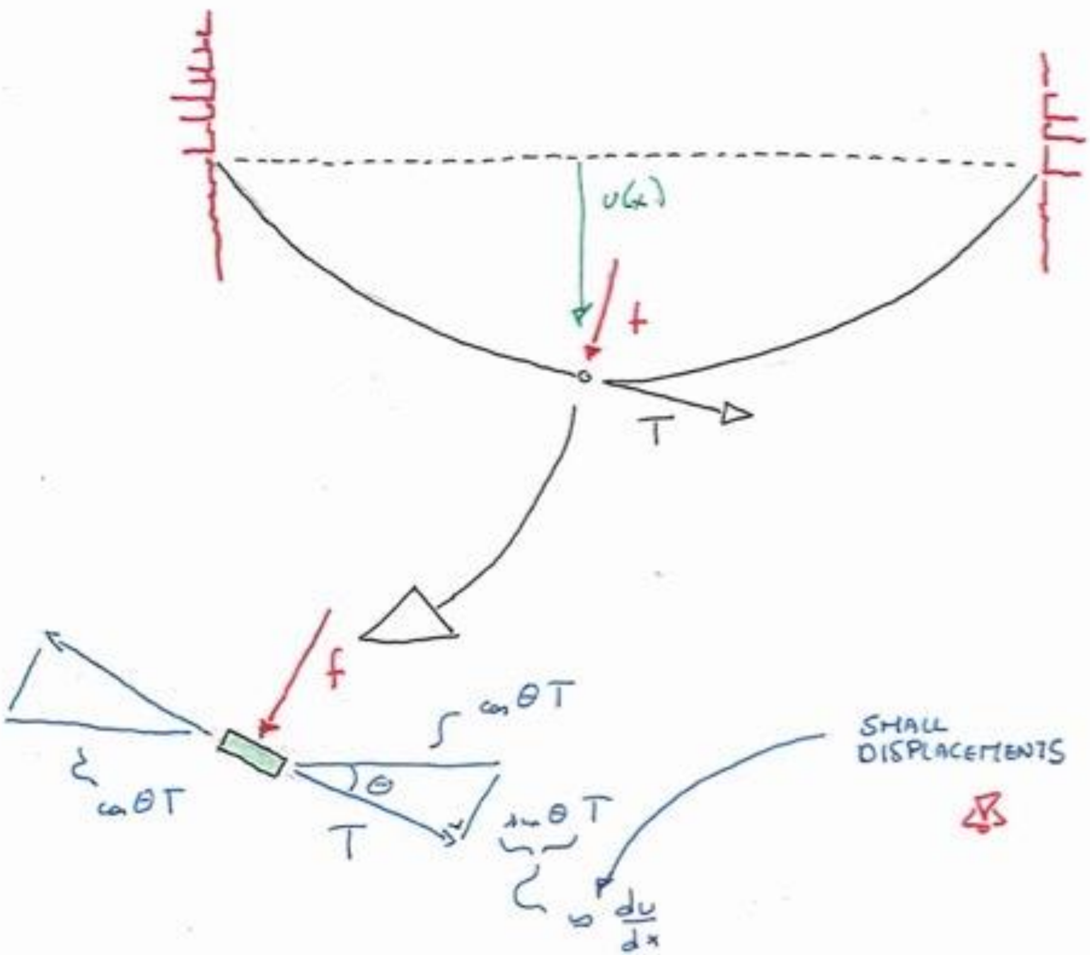
$$\sum_j \langle \tau_i \tau_j \rangle S_j = \langle \tau_i \, \sigma(u^h) \rangle$$



Pratiquement
comment obtenir
des valeurs nodales de tensions ?


SIMPLIFIED MODELS
"EXACT" CONTINUUM MODEL
3D
CONTINUOUS DISPLACEMENT
$u \in \mathcal{U}$
$J(u) = \min_{v \in \mathcal{U}} J(v)$
DISCRETE 3D APPROXIMATIONS
$u \approx u^h \in \mathcal{U}^h \subset \mathcal{U}$
A HIERARCHICAL VIEW
1D / 2D
ROPE / MENBRANE MODELS
$u \approx u_2 \in \mathcal{U}_1 \not\subset \mathcal{U}_2 \subset \mathcal{U}$
DISCRETE APPROXIMATION
$u_1 \approx u_1^h$
1D / 2D
BEAM / SHELLS MODELS
$u \approx u_2 \in \mathcal{U}_2 \subset \mathcal{U}$
$u_2 \approx u_2^h$

ROPE MODEL



VERTICAL FORCE BALANCE

$$\frac{d}{dx}\left(T\,\frac{du}{dx}\right) = -f$$

$$T\,\frac{d^2u}{dx^2} + f = 0$$

THEORY OF BEAMS

FORCE & MOMENTUM BALANCES

$$\frac{dQ}{dx} = -q$$

$$\frac{dM}{dx} = Q$$

CONSTITUTIVE LAW

- BENDING BEAM
- SMALL DEF
- ST VENANT PRINCIPLE
- NAVIER-BERNOULLI ASS.

$$M = EI \frac{1}{R}$$

CURVATURE OF BEAM

$$1/R = -\frac{d^2u}{dx^2}$$

?u

$$\frac{d^4u}{dx^4} - \frac{q}{EI} = 0$$

+ 4 BOUNDARY CONDITIONS

$$\langle EI \frac{d^4u}{dx^4} \hat{u} \rangle = \langle q \hat{u} \rangle$$

$$- \langle EI \frac{d^3u}{dx^3} \frac{d\hat{u}}{dx} \rangle = \langle q \hat{u} \rangle - \left[\overbrace{EI \frac{d^3u}{dx^3}}^{Q} \hat{u} \right]$$

$$\underbrace{\langle EI \frac{d^2u}{dx^2} \frac{d^2\hat{u}}{dx^2} \rangle}_{a(u, \hat{u})} = \underbrace{\langle q \hat{u} \rangle - \left[Q \hat{u} \right] + \left[\underbrace{EI \frac{d^2u}{dx^2}}_{M} \frac{d\hat{u}}{dx} \right]}_{b(\hat{u})}$$

IMPOSING u OR Q

$\frac{du}{dx}$ OR M

AT BOTH ENDS IS EASY !

IT IS A MINIMIZATION PROBLEM...

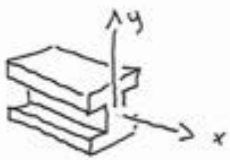
$$J(u) = \frac{1}{2} \left\langle EI \frac{d^2u}{dx^2} \frac{d^2u}{dx^2} \right\rangle - \langle q u \rangle + [Q u] - \left[M \frac{du}{dx} \right]$$

$M(u)$ MOMENTUM

$\chi(u)$ CURVATURE

DEFORMATION ENERGY

WORK OF EXTERNAL FORCES



$$\frac{1}{2} \int_0^L \int_S \sigma \, \varepsilon \, dS \, dx$$

σ: $E y \frac{d^2u}{dx^2}$

ε: $y \frac{d^2u}{dx^2}$

TIMOSHENKO BEAM THEORY

CONSTITUTIVE LAW

$$M = -EI \frac{d^2 v}{dx^2}$$

$$N = ES \frac{du}{dx}$$

CONSERVATION BALANCES

$$\frac{dN}{dx} = -\gamma$$

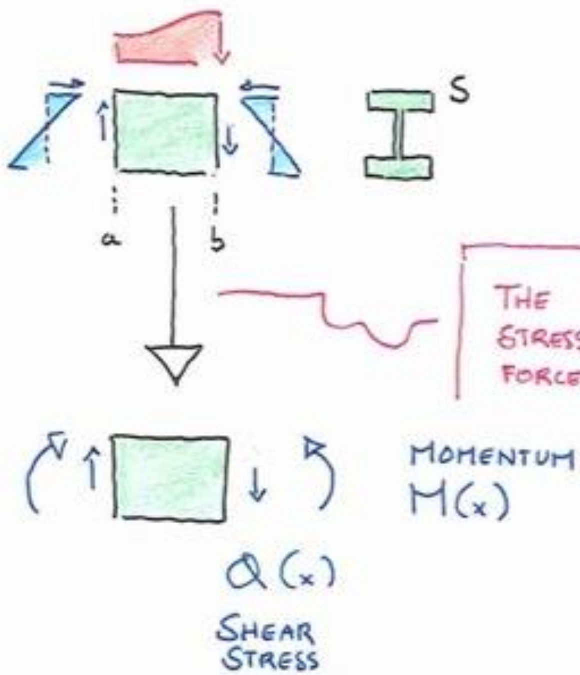
$$\frac{dQ}{dx} = -q$$

$$\frac{dM}{dx} = Q$$

$$EI \frac{d^4 v}{dx^2} = q$$

$$ES \frac{d^2 u}{dx^2} = -\gamma$$

BENDING OF A BEAM



THE PATTERN OF INTERNAL STRESS PRODUCES A TWISTING FORCE, A MOMENT ON THE MATERIAL

HORIZONTAL FORCE OBSERVED

$$\overbrace{\int_S (\sigma)}^{=0} dS = 0$$

MOMENTUM OBSERVED

$$\underbrace{\int_S (\sigma)\,\eta \, dS}_{M} \underset{\text{HOOKE}}{=} \int_S \frac{E\eta^2}{R}\, dS$$

$$= \frac{E}{R} \underbrace{\int_S \eta^2 \, dS}_{I = \frac{bh^3}{12}}$$

$$= EI \underbrace{\frac{1}{R}}_{-\frac{d^2 v}{dx^2}}$$

FORCE BALANCE

$$\int_a^b q\,dx - Q_a + Q_b = 0$$

$$\int_a^b q + \frac{dQ}{dx}\,dx = 0$$

$$\frac{dQ}{dx} = -q$$

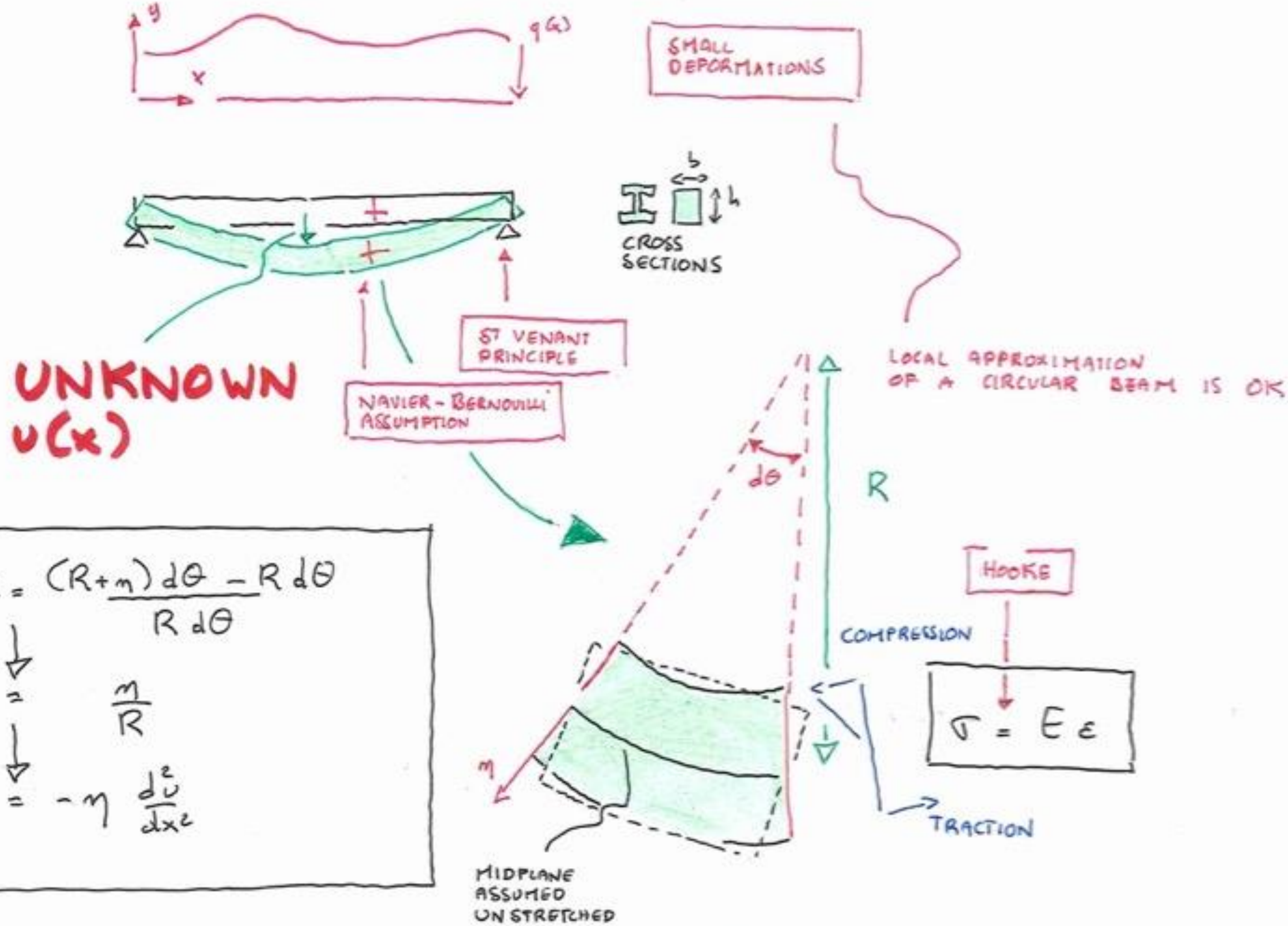
$$\frac{dM}{dx} = Q$$

MOMENTUM BALANCE

$$M_b - M_a - \int_a^b (x-a)\,q\,dx - (b-a)\,Q_b = 0$$

$$\int_a^b \left(\frac{dM}{dx} + (x-a)\,\frac{dQ}{dx}\right) dx - (b-a)\,Q_b = 0$$

$$\int_a^b \left(\frac{dM}{dx} - Q\right) dx - (b-a)\,Q_b + \Big[(x-a)\,Q\Big]_a^b = 0$$

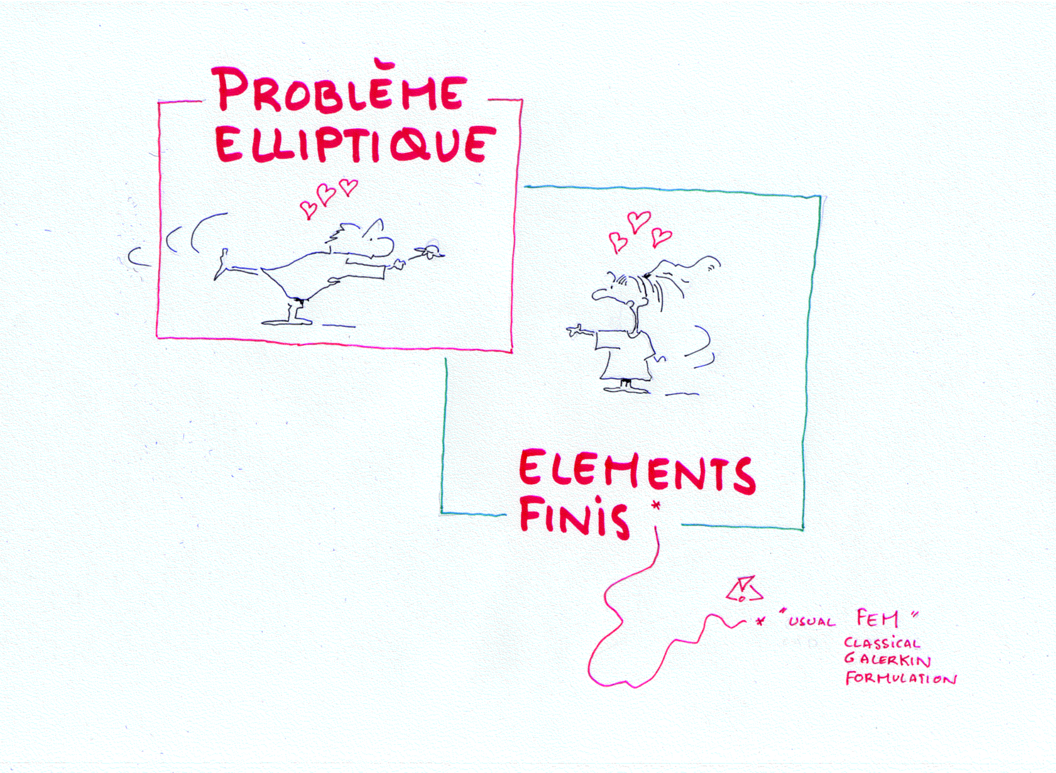


$$\varepsilon = \frac{(R+\eta)\,d\theta - R\,d\theta}{R\,d\theta}$$

$$= \frac{\eta}{R}$$

$$= -\eta \frac{d^2 v}{dx^2}$$

$$\sigma = E\varepsilon$$



Galerkin, c'est optimal
pour des équations elliptiques