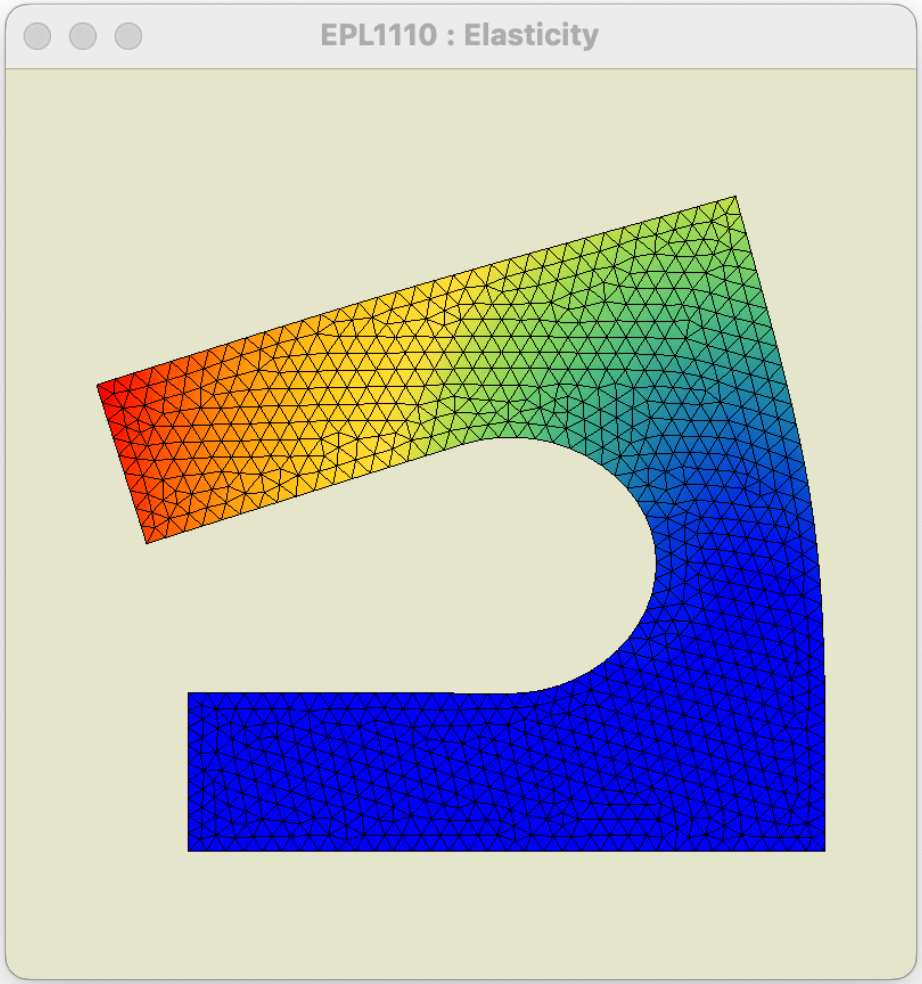
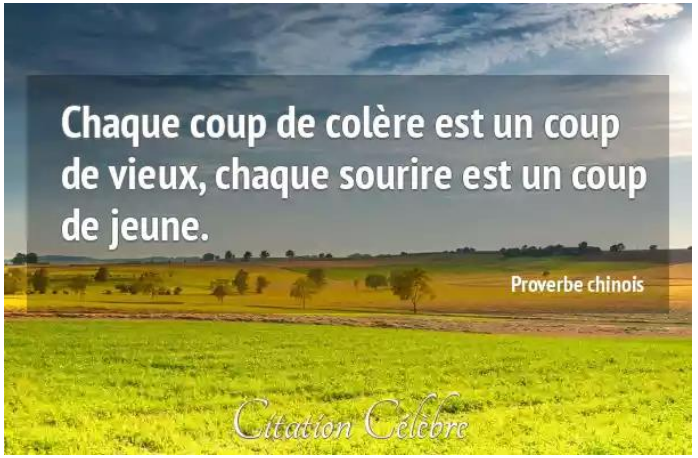
Projet
2024-25

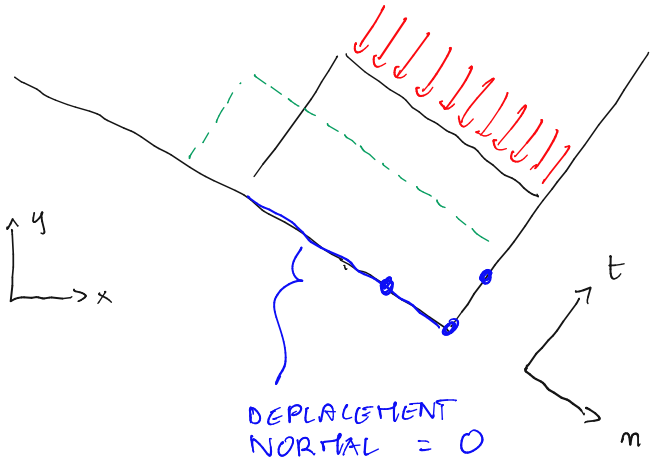


Mes assistant et tuteur
ne me répondent pas…



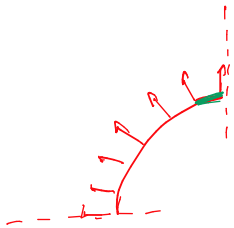
… et c'est uniquement maintenant
qu'on voit apparaître certains groupes !

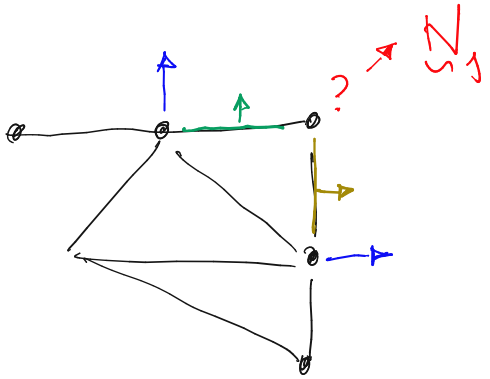
Exprimer les conditions aux limites en termes de normale et tangente !



$$\underline{u} = [u_x \quad u_y]$$

$$\underline{u}_n = [u_n \quad u_t]$$





$$\sum_j \langle \tau_i \tau_j \rangle N_j = \langle \tau_i \underset{\sim}{n} \rangle$$

$$\underset{\sim}{dx} = \begin{bmatrix} dx \\ dy \end{bmatrix}$$

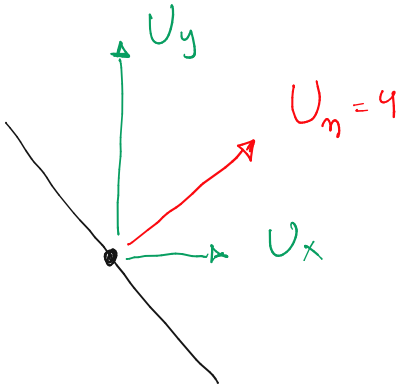
$$\underset{\sim}{t} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \frac{\underset{\sim}{dx}}{|\underset{\sim}{dx}|}$$

$$\underset{\sim}{n} \cdot \underset{\sim}{t} = 0$$

$$\underset{\sim}{n} \sim \begin{bmatrix} -t_y \\ t_x \end{bmatrix} \qquad \underset{\sim}{t} \sim \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Mais au fond,
en réfléchissant
c'est quoi la normale en un nœud ?

Ajouter une contrainte linéaire entre deux variable…



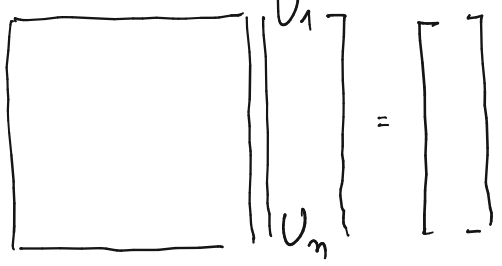
$$U_n = n_x U_x + n_y U_y = 4$$

CECI EST LA CONDITION ESSENTIELLE QUE JE VEUX IMPOSER !

$$U_x = 4 - \frac{n_y}{n_x} U_y$$

… est-ce compliqué ?

$$U_n = a + b\,U_1$$



$$J(v) = \frac{1}{2} \langle \nabla v \cdot \nabla v \rangle - \langle f\, v \rangle$$

$$v \simeq v^h = \sum_{i=1}^{n} U_i \tau_i$$

$$J(v^h) = \frac{1}{2} \left\langle \left(\sum U_i \nabla \tau_i\right) \cdot \left(\sum U_j \nabla \tau_j\right) \right\rangle - \left\langle f \left(\sum U_i \tau_i\right) \right\rangle$$

$$A_{ij} = \langle \nabla \tau_i \cdot \nabla \tau_j \rangle$$

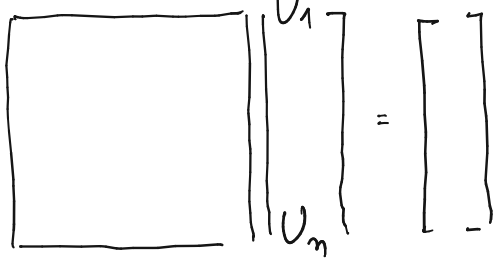
$$B_i = \langle f\, \tau_i \rangle$$

$$= \frac{1}{2} \sum_{ij} A_{ij} U_i U_j - \sum_i B_i U_i$$

$$\frac{\partial J}{\partial U_i} = 0$$

$$\boxed{\sum_{j=1}^{n} A_{ij} U_j = B_i}$$

$$U_n = a + b\,U_1$$



$$v^h = \sum_{i=1}^{n-1} U_i \tau_i + \tau_n (a + b\,U_1)$$

$$J(v^h) = \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=1}^{n-1} A_{ij} U_j U_i - \sum_{i=1}^{n-1} B_i U_i + \sum_{i=1}^{n-1} A_{ni} U_i (a + b\,U_1) + \frac{1}{2} A_{nn} (a + b\,U_1)(a + b\,U_1) - B_n (a + b\,U_1)$$

$\frac{\partial J}{\partial U_i}$

POUR L'EQUATION 1

$$b \left[\left(\sum_{j=1}^{n-1} A_{nj} U_j \right) + A_{nn} (a + b\,U_1) - B_n \right] + \sum_{j=1}^{n-1} A_{1j} U_j - B_1 = 0$$

POUR LES AUTRES EQUATIONS

$$\sum_{j=1}^{n-1} A_{ij} U_j - B_i + A_{in} (a + b\,U_1) = 0$$

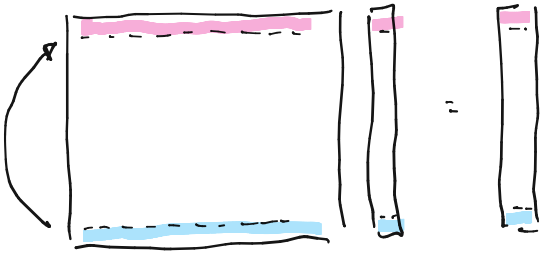
POUR L'EQUATION 1

$$b\left[\left(\sum A_{jn} U_j\right) + A_{nn}(a + bU_1) - B_n\right] + \sum A_{1j} U_j - B_1 = 0$$

Et pratiquement !

POUR LES AUTRES EQUATIONS

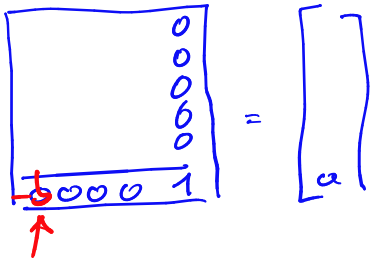
$$\sum A_{ij} U_j - B_i + A_{in}(a + bU_1) = 0$$

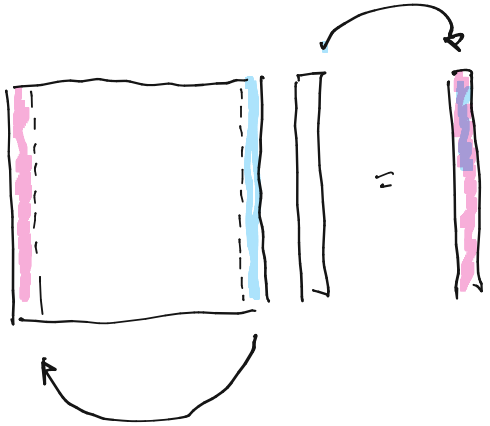


$\blacksquare = \square + b\,\square$

$L1 \leftarrow L1 + b\,Ln$

OPERATION LIGNE





$\blacksquare = \square - a\,\square$

OPERATION COLONNE

CECI EST REALISE LORSQU'ON IMPOSE $U_n = a$

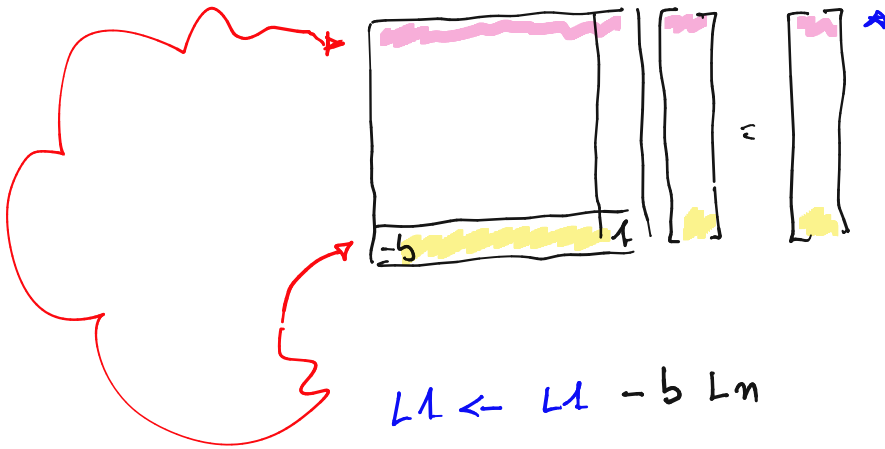
$\blacksquare = \square + b\,\square$ OPERATION COLONNE

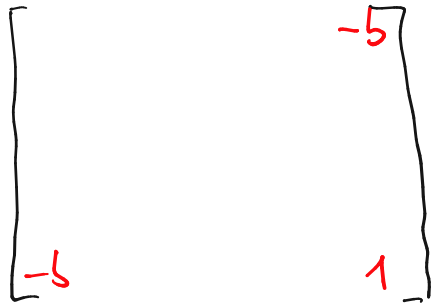
Mais ma matrice n'est plus symétrique !

$U_n = a + b U_1$

$-b U_1 + U_n = a$

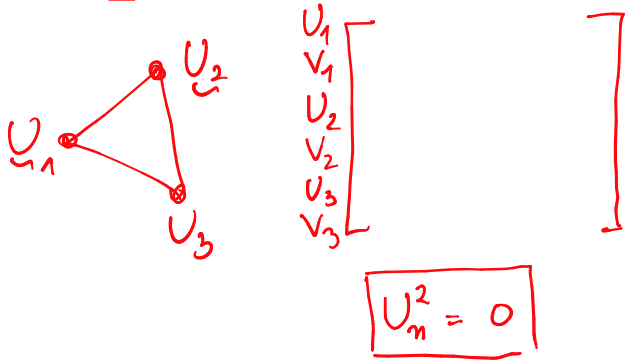
$-b(U_n - b U_1) = -ba$



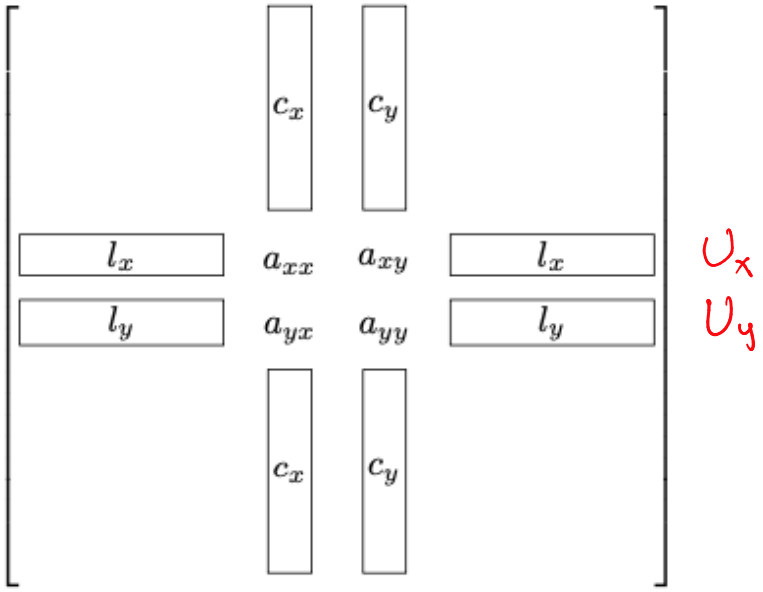


$n_x U_2 + n_y V_2 = 4$

Imposer un déplacement normal !
Est-ce vraiment compliqué ?



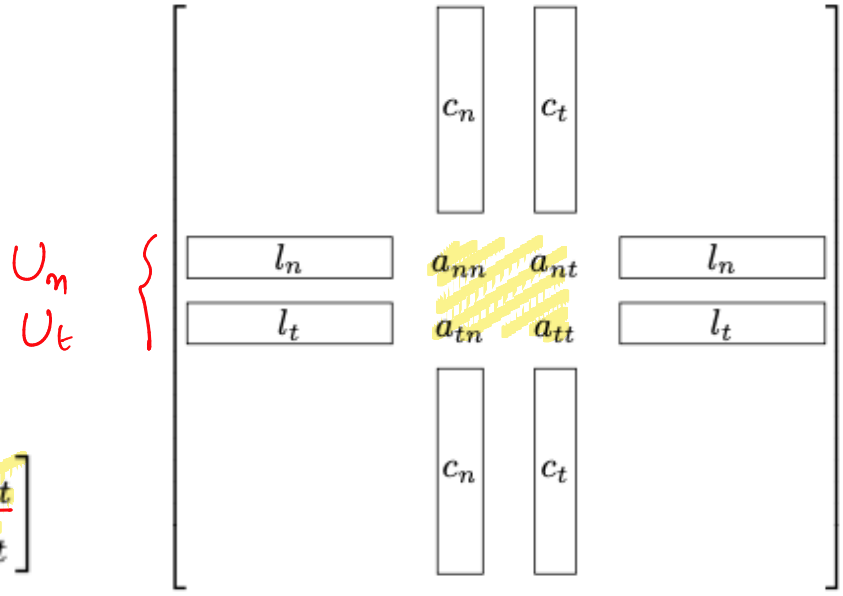
$$A = \begin{bmatrix} & & c_x & c_y & \\ l_x & & a_{xx} & a_{xy} & l_x \\ l_y & & a_{yx} & a_{yy} & l_y \\ & & c_x & c_y & \end{bmatrix} \begin{matrix} \\ U_1 \\ V_2 \\ \\ \end{matrix}$$



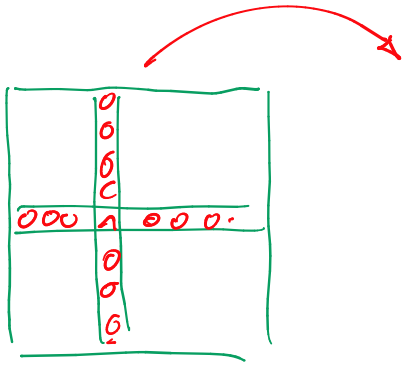
$$U_n = n_x U_x + n_y U_y$$
$$U_t = t_x U_x + n_y U_y$$

Faisons
un changement
de variables

$$\begin{bmatrix} n_x & n_y \\ t_x & t_y \end{bmatrix} \begin{bmatrix} a_{xx} & a_{xy} \\ a_{yx} & a_{yy} \end{bmatrix} \begin{bmatrix} n_x & t_x \\ n_y & t_y \end{bmatrix} = \begin{bmatrix} a_{nn} & a_{nt} \\ a_{tn} & a_{tt} \end{bmatrix}$$



$$\begin{bmatrix} & c_n & c_t & \\ l_n & a_{nn} & a_{nt} & l_n \\ l_t & a_{tn} & a_{tt} & l_t \\ & c_n & c_t & \end{bmatrix}$$



Imposons
la condition essentielle
sur la composante normale

$$\begin{bmatrix} & 0 & c_t & \\ 0 & 1 & 0 & 0 \\ l_t & 0 & a_{tt} & l_t \\ & 0 & c_t & \end{bmatrix}$$

$$\begin{bmatrix} & 0 & c_t & \\ 0 & 1 & 0 & 0 \\ l_t & 0 & a_{tt} & l_t \\ & 0 & c_t & \end{bmatrix}$$

nt

xy

Revenons
aux variables
originales en xy

$$\begin{bmatrix} n_x & t_x \\ n_y & t_y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & a_{tt} \end{bmatrix} \begin{bmatrix} n_x & n_y \\ t_x & t_y \end{bmatrix} = \begin{bmatrix} a'_{xx} & a'_{xy} \\ a'_{yx} & a'_{yy} \end{bmatrix}$$

$$\begin{bmatrix} & c'_x & c'_y & \\ l'_x & a'_{xx} & a'_{xy} & l'_x \\ l'_y & a'_{yx} & a'_{yy} & l'_y \\ & c'_x & c'_y & \end{bmatrix}$$

$$
\begin{aligned}
l'_x &= n_x \cdot 0 + t_x \cdot l_t = t_x(t_x l_x + t_y l_y) \\
l'_y &= n_y \cdot 0 + t_y \cdot l_t = t_y(t_x l_x + t_y l_y) \\
c'_x &= n_x \cdot 0 + t_x \cdot c_t = t_x(t_x c_x + t_y c_y) \\
c'_y &= n_y \cdot 0 + t_y \cdot c_t = t_y(t_x c_x + t_y c_y)
\end{aligned}
$$

Et pratiquement
pour les lignes et colonnes !

$$
\begin{bmatrix}
 & c'_x & c'_y & \\
l'_x & a'_{xx} & a'_{xy} & l'_x \\
l'_y & a'_{yx} & a'_{yy} & l'_y \\
 & c'_x & c'_y &
\end{bmatrix}
$$

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 !

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 !

Le dernier devoir…

```
void femElasticityAssembleNeumann(femProblem *theProblem){
    femFullSystem  *theSystem = theProblem->system;
    femIntegration *theRule = theProblem->ruleEdge;
    femDiscrete    *theSpace = theProblem->spaceEdge;
    femGeo         *theGeometry = theProblem->geometry;
    femNodes       *theNodes = theGeometry->theNodes;
    femMesh        *theEdges = theGeometry->theEdges;
    double x[2],y[2],phi[2];
    int iBnd,iElem,iInteg,iEdge,i,j,d,map[2],mapU[2];
    int nLocal = 2;
    double *B  = theSystem->B;

    for(iBnd=0; iBnd < theProblem->nBoundaryConditions; iBnd++){
        femBoundaryCondition *theCondition = theProblem->conditions[iBnd];
        femBoundaryType type = theCondition->type;
        double value = theCondition->value;
        int shift=-1;
        if (type == NEUMANN_X)  shift = 0;
        if (type == NEUMANN_Y)  shift = 1;
        if (shift == -1) continue;

        for(iEdge=0; iEdge < theCondition->domain->nElem; iEdge++){
            iElem = theCondition->domain->elem[iEdge];
            for (j=0; j < nLocal; j++) {
                map[j]  = theEdges->elem[iElem*nLocal+j];
                mapU[j] = 2*map[j] + shift;
                x[j]    = theNodes->X[map[j]];
                y[j]    = theNodes->Y[map[j]];}

            double jac = sqrt((x[1]-x[0])*(x[1]-x[0]) + (y[1]-y[0])*(y[1]-y[0]))/2.0;
            for (iInteg=0; iInteg < theRule->n; iInteg++) {
                double xsi    = theRule->xsi[iInteg];
                double weight = theRule->weight[iInteg];
                femDiscretePhi(theSpace,xsi,phi);
                for (i = 0; i < theSpace->n; i++) {
                    B[mapU[i]] += jac * weight * phi[i] * value; }}}}
}
```