Trouver $u(x)$ tel que

$$\frac{d^2u}{dx^2} + f = 0, \qquad \forall\, x \in \Omega,$$

$$u(0) = 0,$$
$$u(1) = 0,$$

Effectuons un tout petit exemple :-)

Problème de la corde à linge tendue !

Formulation forte

Trouver $u(x)$ tel que

$$\frac{d^2u}{dx^2} + f = 0, \qquad \forall\, x \in \Omega,$$

$$u(0) = 0,$$
$$u(1) = 0,$$

Plus exigeant !
Espace des solutions plus petit !
On perd des solutions réellement utiles !

Plus laxiste !!
Espace des solutions plus grand !
Les solutions en sus sont utiles !

Trouver $u(x) \in \mathcal{U}$ tel que

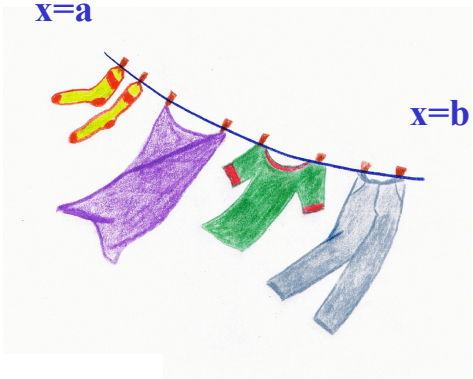
$$\underbrace{< \frac{d\widehat{u}}{dx}\frac{du}{dx} >}_{a(\widehat{u},u)} = \underbrace{< \widehat{u} f >}_{b(\widehat{u})}, \qquad \forall \widehat{u} \in \mathcal{U},$$

Trouver $u(x) \in \mathcal{U}$ tel que

$$J(u) = min_{v \in \mathcal{U}} \underbrace{\left(\tfrac{1}{2}\, a(v,v) - b(v)\right)}_{J(v)},$$

Formulation faible

La vraie formulation physique…
C'est une formulation intégrale !



$$(\frac{du}{dx})^2 \ll 1.$$

Tension constante dans la corde
Petits déplacements

$$\int_a^b f\,dx = T\frac{du}{dx}(a) - T\frac{du}{dx}(b) \qquad \forall a, b$$

$$\int_a^b f\,dx = -T\left[\frac{du}{dx}\right]_a^b \qquad \forall a, b$$

Si la fonction $\frac{du}{dx}$ est continue !

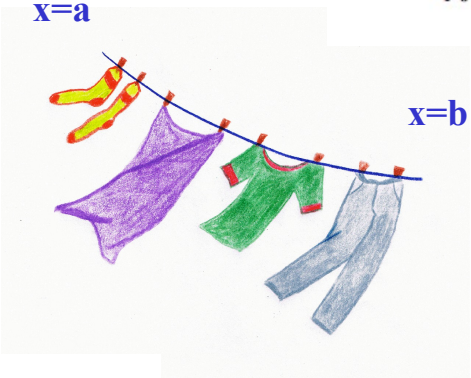
$$\int_a^b f + T\,\frac{d^2u}{dx^2}\,dx = 0 \qquad \forall a, b$$

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

Equilibre vertical des forces

La vraie formulation physique…
C'est minimiser l'énergie !

$$l = \int_0^L dl = \int_0^L \sqrt{dx^2 + du^2} = \int_0^L \sqrt{1 + \left(\frac{du}{dx}\right)^2}\, dx \approx \int_0^L \left(1 + \tfrac{1}{2}\left(\frac{du}{dx}\right)^2\right) dx$$



$$(\frac{du}{dx})^2 \ll 1.$$

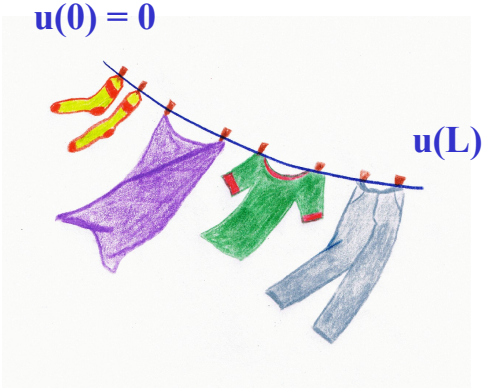
Tension constante dans la corde
Petits déplacements

$$J(u) = T(l - L) - \int_0^L f\ u\ dx$$

$$J(u) = \tfrac{1}{2}\int_0^L T\left(\frac{du}{dx}\right)^2 dx - \int_0^L f\ u\ dx$$

Minimisation de l'énergie
On minimise le travail des forces !

u(0) = 0



u(L) = 0

Problème discret

$$\sum_{j=2}^{N-1} A_{ij} U_j = B_i, \qquad i = 2, N-1.$$

$$\begin{aligned} U_1 &= 0, \\ U_N &= 0, \end{aligned}$$

N-2 équations pour les valeurs intérieures
Deux conditions aux limites

$$A_{ij} = \int_\Omega \frac{d\tau_i}{dx}(x)\ \frac{d\tau_j}{dx}(x)\ dx,$$

$$B_i = \int_\Omega f(x)\tau_i(x)\ dx,$$

Trouver $u(x)$ tel que

$$\frac{d^2u}{dx^2} + f = 0, \qquad \forall\, x \in \Omega,$$

$$u(0) = 0,$$
$$u(1) = 0,$$

Retrouvons notre petit exemple

Problème de la corde à linge tendue !

Formulation forte

Trouver $u(x)$ tel que

$$\frac{d^2u}{dx^2} + f = 0, \qquad \forall\, x \in \Omega,$$

$$u(0) = 0,$$
$$u(1) = 0,$$

Plus exigeant !
Espace des solutions plus petit !
On perd des solutions réellement utiles !

Plus laxiste !!
Espace des solutions plus grand !
Les solutions en sus sont utiles !

Trouver $u(x) \in \mathcal{U}$ tel que

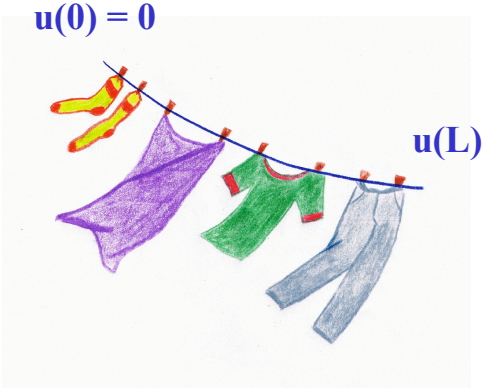
$$\underbrace{< \frac{d\widehat{u}}{dx}\frac{du}{dx} >}_{a(\widehat{u},u)} = \underbrace{< \widehat{u} f >}_{b(\widehat{u})}, \qquad \forall \widehat{u} \in \mathcal{U},$$

Trouver $u(x) \in \mathcal{U}$ tel que

$$J(u) = min_{v \in \mathcal{U}} \underbrace{\left(\tfrac{1}{2}\, a(v,v) - b(v)\right)}_{J(v)},$$

Formulation faible

u(0) = 0



u(L) = 0

Problème discret

$$\sum_{j=2}^{N-1} A_{ij} U_j = B_i, \qquad i = 2, N-1.$$

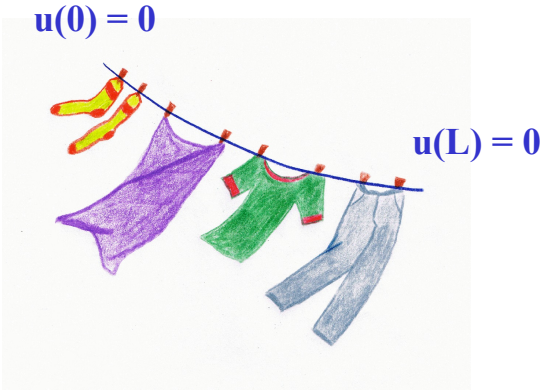
$$\begin{aligned} U_1 &= 0, \\ U_N &= 0, \end{aligned}$$

N-2 équations pour les valeurs intérieures
Deux conditions aux limites

$$A_{ij} = \int_\Omega \frac{d\tau_i}{dx}(x)\ \frac{d\tau_j}{dx}(x)\ dx,$$

$$B_i = \int_\Omega f(x)\tau_i(x)\ dx,$$

Problème discret



La version pour toi qui n’a pas compris !

Construisons
le système
linéaire

Construisons le système linéaire

$$A_{i\ i-1} = \int_{\Omega_e} \phi^e_{2,x}(x)\ \phi^e_{1,x}(x)\ dx,$$

$$A_{ii} = \int_{\Omega_e} \phi^e_{2,x}(x)\ \phi^e_{2,x}(x)\ dx + \int_{\Omega_{e+1}} \phi^{e+1}_{1,x}(x)\ \phi^{e+1}_{1,x}(x)\ dx,$$

$$A_{i\ i+1} = \int_{\Omega_{e+1}} \phi^{e+1}_{1,x}(x)\ \phi^{e+1}_{2,x}(x)\ dx,$$

$$B_i = \int_{\Omega_e} \phi^e_2(x)\ f(x)\ dx + \int_{\Omega_{e+1}} \phi^{e+1}_1(x)\ f(x)\ dx.$$

On peut obtenir aisément le système global en assemblant les systèmes locaux !

Matrices de raideur locales

$$A^e_{ij} = \int_{\Omega_e} \phi^e_{i,x}(x)\ \phi^e_{j,x}(x)\ dx,$$

$$B^e_i = \int_{\Omega_e} f(x)\phi^e_i(x)\ dx.$$

Vecteurs des forces locales

C'est comme construire un système multicorps !

$$
\begin{aligned}
A_{i\ i-1} &= A^{e}_{21}, \\
A_{ii} &= A^{e}_{22} + A^{e+1}_{11}, \\
A_{i\ i+1} &= A^{e+1}_{12}, \\
B_i &= B^{e}_{2} + B^{e+1}_{1}.
\end{aligned}
$$

$$
\begin{array}{lcll}
B_1 & = & 0.5 & \\
 & & 0.5 & \\
 & & \cdots & \\
 & & 1.0 & \\
B_i & = & 1.0 & \boxed{+\,3.5} \\
 & & 0.0 & \\
 & & \cdots & \\
 & & 1.0 & \\
B_j & & 0.5 & \boxed{+\,4.5} \\
 & & 0.5 & \\
 & & \cdots & \\
 & & 1.0 & \\
B_N & = & 4.0 &
\end{array}
$$

$$
\begin{aligned}
B^{e}_{1} &= 3.5 \\
B^{e}_{2} &= 4.5
\end{aligned}
$$

Eléments	Noeuds
...	
e	i j
...	

Chaque élément fini peut être vu comme une pièce d'un petit Mecano !

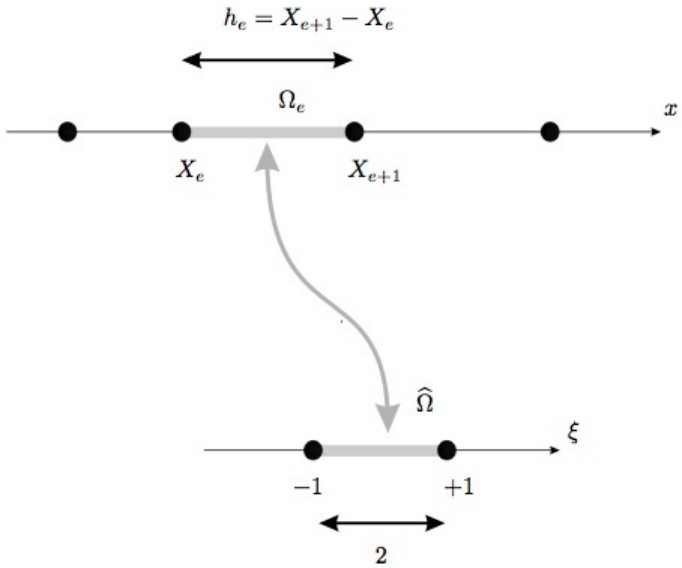
Il y a beaucoup d'intégrales !

$$x(\xi) = \xi\frac{(X_{e+1}-X_e)}{2} + \frac{(X_{e+1}+X_e)}{2},$$

$$\xi(x) = \frac{2x-(X_{e+1}+X_e)}{(X_{e+1}-X_e)}.$$

Isomorphisme linéaire entre l'élément parent et tous les autres éléments…

$$\begin{aligned}
A_{ij}^e &= \int_{\Omega_e} \phi_{i,x}^e(x)\ \phi_{j,x}^e(x)\ dx, \\
&= \int_{-1}^{1} \left(\phi_{i,\xi}(\xi)\frac{d\xi}{dx}\right)\ \left(\phi_{j,\xi}(\xi)\frac{d\xi}{dx}\right)\ \left(\frac{dx}{d\xi}\ d\xi\right), \\
&= \int_{-1}^{1} \phi_{i,\xi}(\xi)\phi_{j,\xi}(\xi)\frac{d\xi}{dx}\ d\xi, \\
&= \frac{2}{(X_{e+1}-X_e)}\int_{-1}^{1} \phi_{i,\xi}(\xi)\phi_{j,\xi}(\xi)\ d\xi, \\
B_i^e &= \int_{\Omega} \phi_i^e(x)\ f(x)\ dx, \\
&= \frac{(X_{e+1}-X_e)}{2}\int_{-1}^{1} \phi_i(\xi) f(x(\xi))\ d\xi.
\end{aligned}$$



On intègre systématiquement sur l'élément parent !

Effectuons
un tout petit
exercice
à la main :

Et zou !

Effectuons un tout petit exercice à la main :

$$\phi_1(\xi) = (1-\xi)/2, \qquad \phi_{1,\xi}(\xi) = -1/2,$$
$$\phi_2(\xi) = (1+\xi)/2, \qquad \phi_{2,\xi}(\xi) = 1/2.$$

$$B_i^e = \frac{h}{6}\begin{bmatrix} 2X_e + X_{e+1} \\ X_e + 2X_{e+1} \end{bmatrix}$$

$$A_{ij}^1 = \frac{1}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \qquad B_i^1 = \frac{h}{6}\begin{bmatrix} 1/2 \\ 1 \end{bmatrix}$$

$$A_{ij}^2 = \frac{1}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \qquad B_i^2 = \frac{h}{6}\begin{bmatrix} 2 \\ 5/2 \end{bmatrix}$$

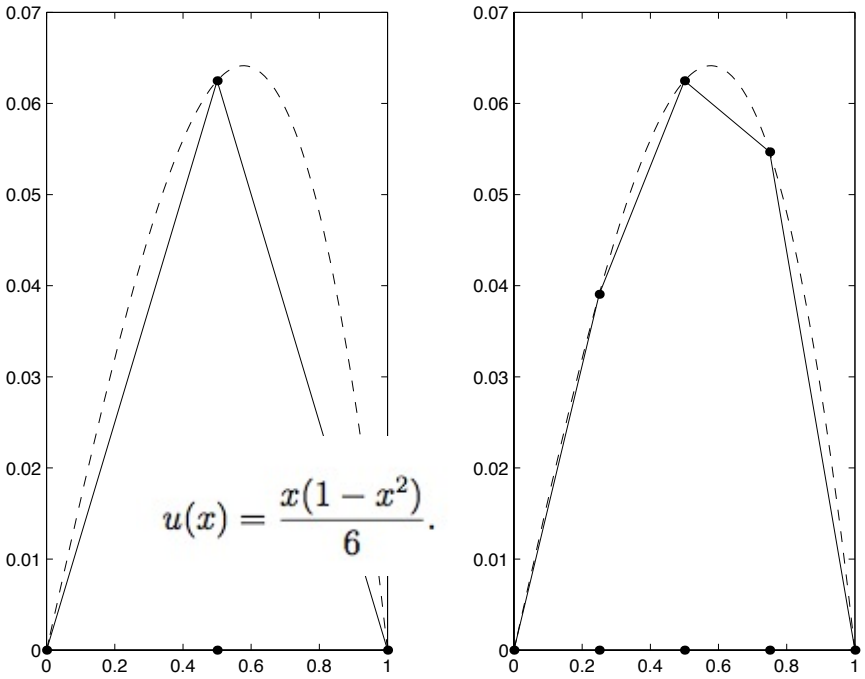
$$\frac{1}{h}\begin{bmatrix} 1 & -1 & \\ -1 & 2 & -1 \\ & -1 & 1 \end{bmatrix}\begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} = \frac{h}{6}\begin{bmatrix} 1/2 \\ 6/2 \\ 5/2 \end{bmatrix}$$

Système discret

$f(x) = x$

Exemple

$f(x) = x$



$$\frac{1}{h}\begin{bmatrix} 1 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & -1 & 2 & -1 \\ & & & -1 & 1 \end{bmatrix}\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{bmatrix} = \frac{h}{6}\begin{bmatrix} 1/4 \\ 6/4 \\ 12/4 \\ 18/4 \\ 11/4 \end{bmatrix}.$$

$$\frac{1}{h}\begin{bmatrix} 2 & -1 & \\ -1 & 2 & -1 \\ & -1 & 2 \end{bmatrix}\begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix} = \frac{h}{6}\begin{bmatrix} 6/4 \\ 12/4 \\ 18/4 \end{bmatrix}$$

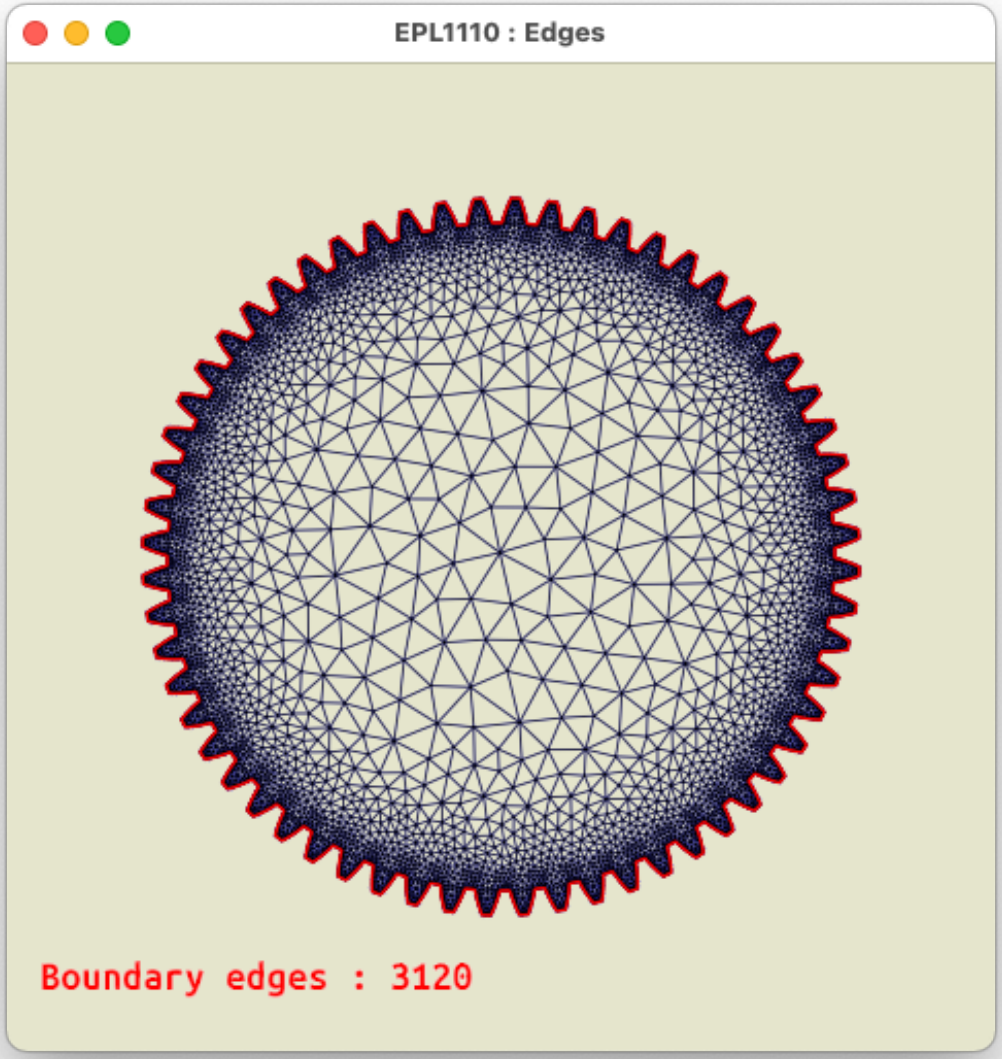
$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 5/128 \\ 8/128 \\ 7/128 \\ 0 \end{bmatrix}$$

Système discret :
3 valeurs nodales inconnues
2 conditions aux limites

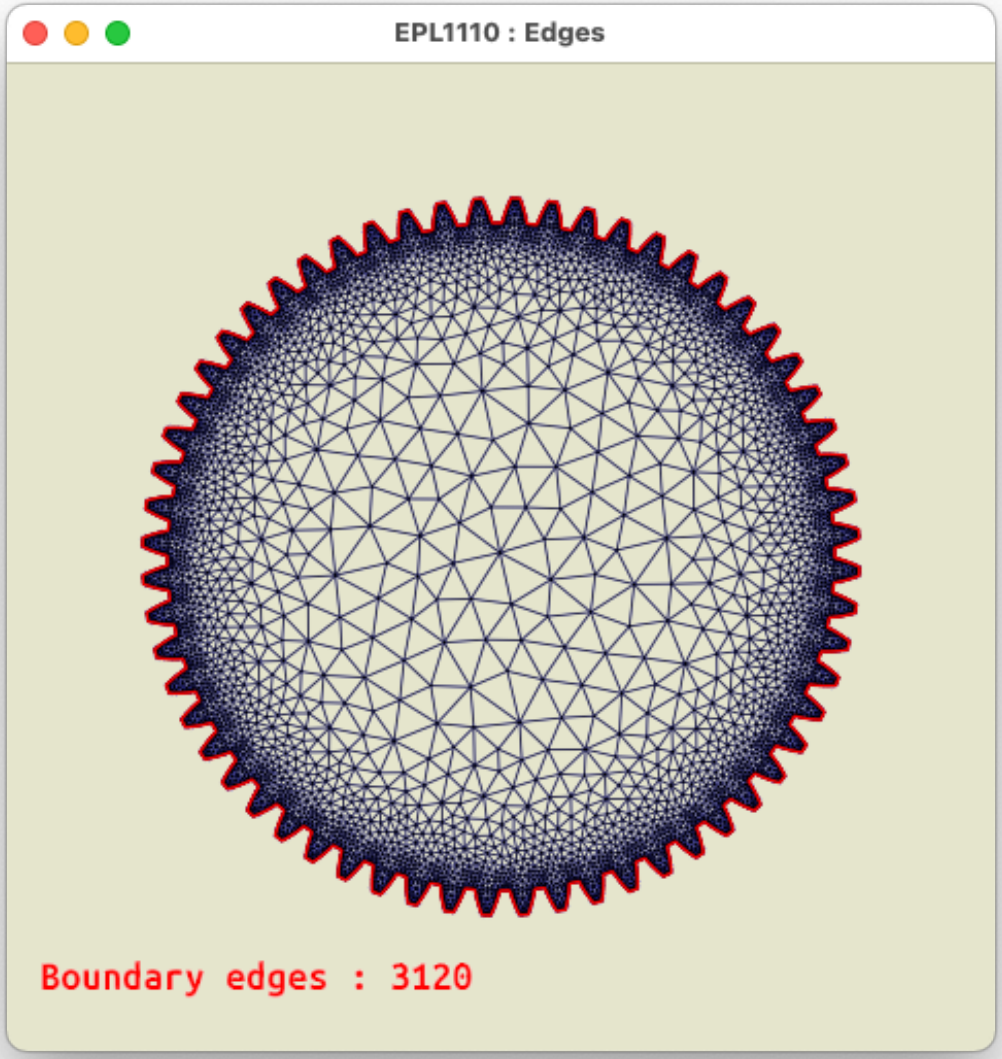
$$\frac{1}{h}\begin{bmatrix} 1 & -1 & \\ -1 & 2 & -1 \\ & -1 & 1 \end{bmatrix}\begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} = \frac{h}{6}\begin{bmatrix} 1/2 \\ 6/2 \\ 5/2 \end{bmatrix}$$

$$U_2 = h^2/4 = 1/16,$$

Système discret :
1 valeur nodale inconnues
2 conditions aux limites



Le maillage est un graphe !



Déduire la table des segments
Obtenir la frontière du maillage
Obtenir la longueur de la frontière

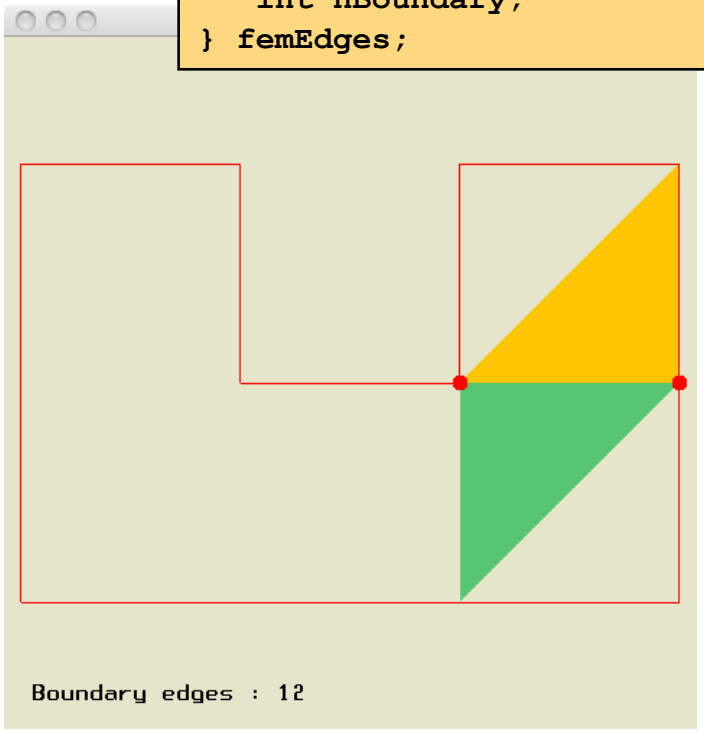
Une structure pour un segment

```
typedef struct {
   int elem[2];
   int node[2];
} femEdge;
```

```
typedef struct {
   femMesh *mesh;
   femEdge *edges;
   int nEdge;
   int nBoundary;
} femEdges;
```

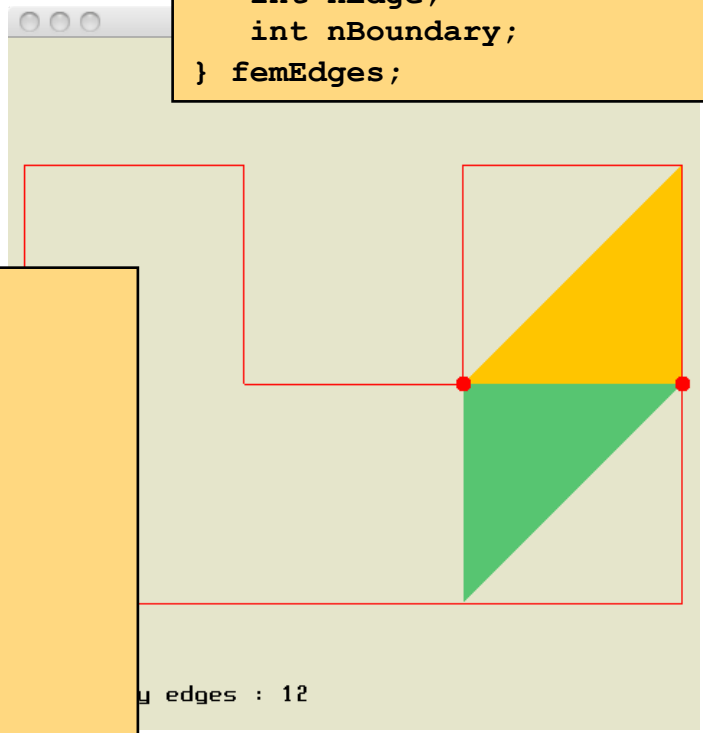
Nous allons faire les tâches suivantes :

- **Collationner tous les segments**
- **Les trier…**
- **Supprimer les doublons**
- **Identifier les segments frontières**



Une structure pour les segments

```
typedef struct {
    femMesh *mesh;
    femEdge *edges;
    int nEdge;
    int nBoundary;
} femEdges;
```



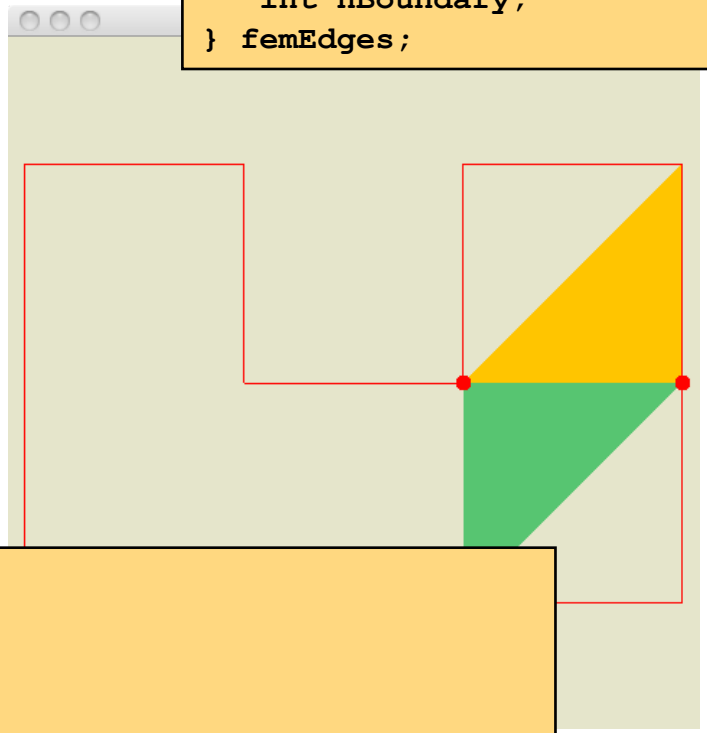

```
femEdges *femEdgesCreate(femMesh *theMesh)
{
    femEdges *theEdges = malloc(sizeof(femEdges));
    int nLoc = theMesh->nLocalNode;
    int n    = theMesh->nElem * nLoc;
    femEdge* edges = malloc(n * sizeof(femEdge));
    theEdges->mesh  = theMesh;
    theEdges->edges = edges;
    theEdges->nEdge = n;
    theEdges->nBoundary = n;
    return theEdges;
}
```

```
void femEdgesFree(femEdges *theEdges)
{
    free(theEdges->edges);
    free(theEdges);
}
```

Accéder aux données de la structure dont on a l'adresse !

```
typedef struct {
    int elem[2];
    int node[2];
} femEdge;
```

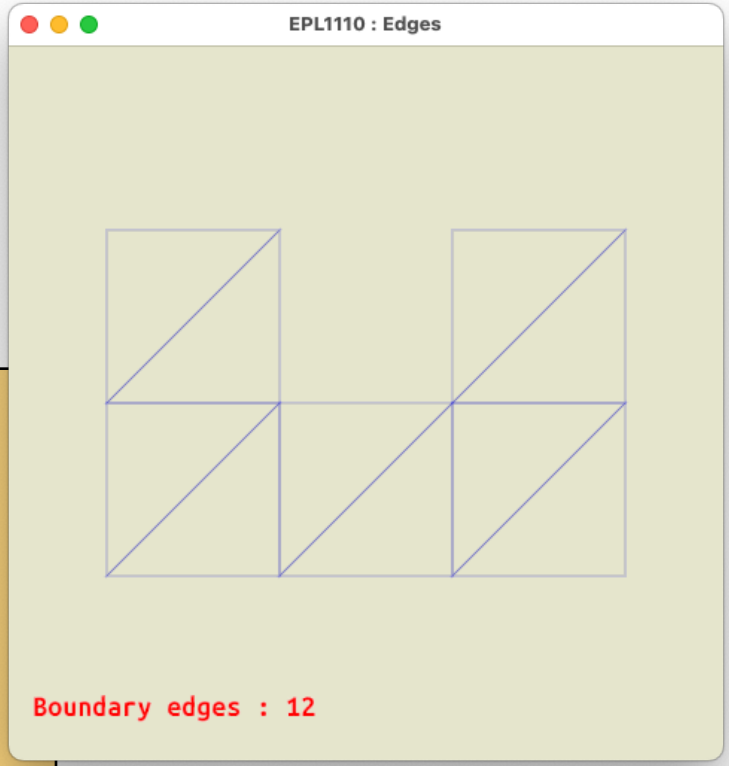
```
typedef struct {
    femMesh *mesh;
    femEdge *edges;
    int nEdge;
    int nBoundary;
} femEdges;
```



```
void femEdgesPrint(femEdges *theEdges)
{
    int i;
    for (i = 0; i < theEdges->nEdge; ++i) {
        printf("%6d : %4d %4d : %4d %4d \n",i,
               theEdges->edges[i].node[0],theEdges->edges[i].node[1],
               theEdges->edges[i].elem[0],theEdges->edges[i].elem[1]); }
}
```

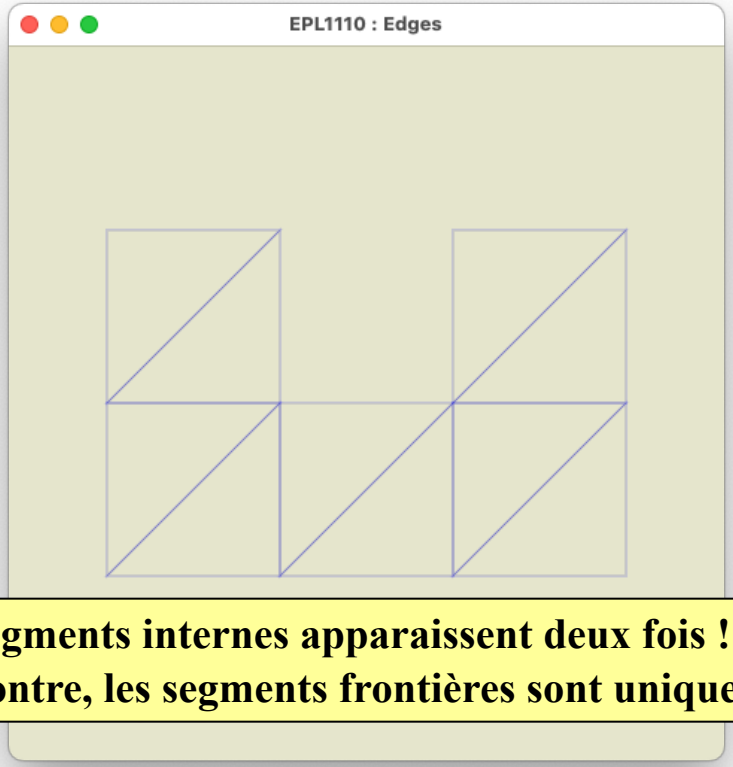
Et vraiment concrètement ?

```
Number of nodes 12
     0 :  0.0000000e+00  2.0000000e+00
     1 :  1.0000000e+00  2.0000000e+00
     2 :  0.0000000e+00  1.0000000e+00
     3 :  1.0000000e+00  1.0000000e+00
     4 :  0.0000000e+00  0.0000000e+00
     5 :  1.0000000e+00  0.0000000e+00
     6 :  2.0000000e+00  1.0000000e+00
     7 :  2.0000000e+00  0.0000000e+00
     8 :  3.0000000e+00  0.0000000e+00
     9 :  3.0000000e+00  1.0000000e+00
    10 :  2.0000000e+00  2.0000000e+00
    11 :  3.0000000e+00  2.0000000e+00
Number of triangles 10
     0 :      0      2      1
     1 :      2      3      1
     2 :      4      3      2
     3 :      5      7      6
     4 :      7      9      6
     5 :      7      8      9
     6 :      6     11     10
     7 :      9     11      6
     8 :      4      5      3
     9 :      5      6      3
```



Collationner tous les segments

10 éléments
30 segments

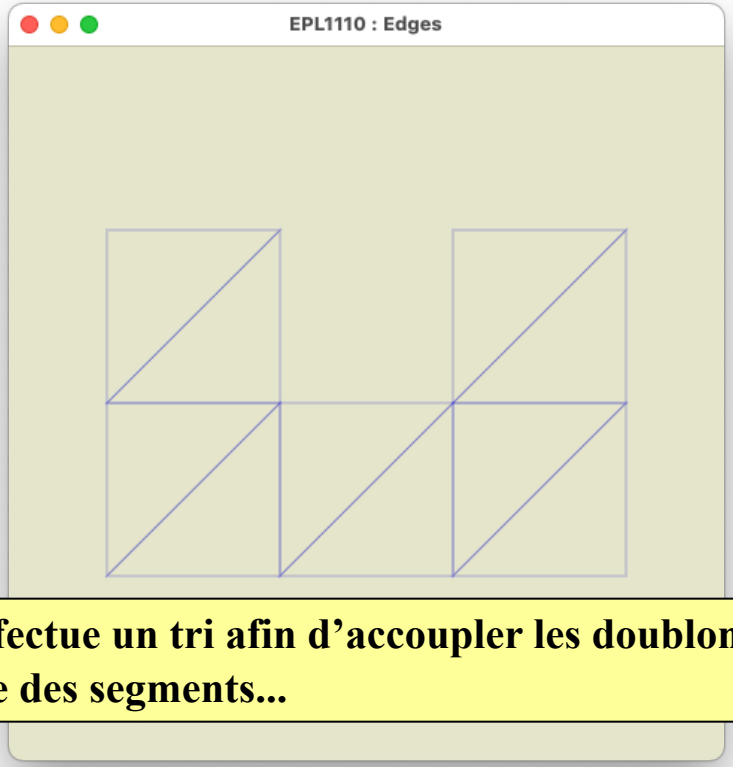


Les segments internes apparaissent deux fois !
Par contre, les segments frontières sont uniques

```
 0 :  0   2 :  0  -1
 1 :  2   1 :  0  -1
 2 :  1   0 :  0  -1
 3 :  2   3 :  1  -1
 4 :  3   1 :  1  -1
 5 :  1   2 :  1  -1
 6 :  4   3 :  2  -1
 7 :  3   2 :  2  -1
 8 :  2   4 :  2  -1
 9 :  5   7 :  3  -1
10 :  7   6 :  3  -1
11 :  6   5 :  3  -1
12 :  7   9 :  4  -1
13 :  9   6 :  4  -1
14 :  6   7 :  4  -1
15 :  7   8 :  5  -1
16 :  8   9 :  5  -1
17 :  9   7 :  5  -1
18 :  6  11 :  6  -1
19 : 11  10 :  6  -1
20 : 10   6 :  6  -1
21 :  9  11 :  7  -1
22 : 11   6 :  7  -1
23 :  6   9 :  7  -1
24 :  4   5 :  8  -1
25 :  5   3 :  8  -1
26 :  3   4 :  8  -1
27 :  5   6 :  9  -1
28 :  6   3 :  9  -1
29 :  3   5 :  9  -1
```

Trier les segments

10 éléments
30 segments

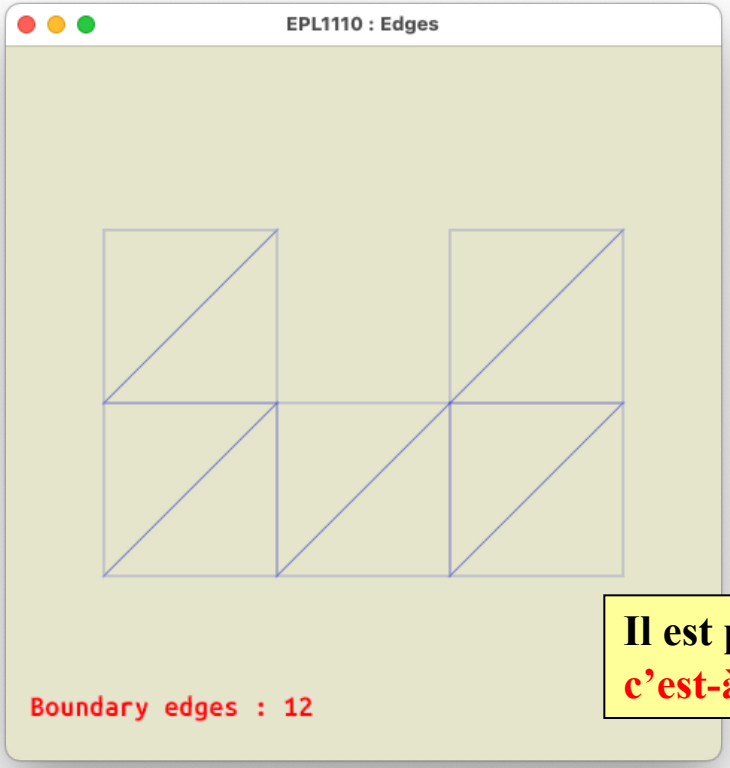


On effectue un tri afin d'accoupler les doublons dans la liste des segments...

```
 0 :  11  10 :  6  -1
 1 :   9  11 :  7  -1
 2 :   8   9 :  5  -1
 3 :   7   9 :  4  -1
 4 :   9   7 :  5  -1
 5 :   7   8 :  5  -1
 6 :   6  11 :  6  -1
 7 :  11   6 :  7  -1
 8 :  10   6 :  6  -1
 9 :   9   6 :  4  -1
10 :   6   9 :  7  -1
11 :   6   7 :  4  -1
12 :   7   6 :  3  -1
13 :   5   7 :  3  -1
14 :   6   5 :  3  -1
15 :   5   6 :  9  -1
16 :   4   5 :  8  -1
17 :   6   3 :  9  -1
18 :   3   5 :  9  -1
19 :   5   3 :  8  -1
20 :   3   4 :  8  -1
21 :   4   3 :  2  -1
22 :   2   4 :  2  -1
23 :   3   2 :  2  -1
24 :   2   3 :  1  -1
25 :   3   1 :  1  -1
26 :   2   1 :  0  -1
27 :   1   2 :  1  -1
28 :   0   2 :  0  -1
29 :   1   0 :  0  -1
```

Supprimer les doublons

9 segments internes
12 segments frontières



0	:	11	10	:	6	-1
1	:	9	11	:	7	-1
2	:	8	9	:	5	-1
3	:	7	9	:	4	5
4	:	7	8	:	5	-1
5	:	6	11	:	6	7
6	:	10	6	:	6	-1
7	:	9	6	:	4	7
8	:	6	7	:	4	3
9	:	5	7	:	3	-1
10	:	6	5	:	3	9
11	:	4	5	:	8	-1
12	:	6	3	:	9	-1
13	:	3	5	:	9	8
14	:	3	4	:	8	2
15	:	2	4	:	2	-1
16	:	3	2	:	2	1
17	:	3	1	:	1	-1
18	:	2	1	:	0	1
19	:	0	2	:	0	-1
20	:	1	0	:	0	-1

Il est possible d'effectuer cette opération en place, c'est-à-dire en travaillant dans le même tableau !

```
qsort(edges,n,sizeof(femEdge),edgeCompare);
```

```
 0 :   11    6 :    7   -1
 1 :   11   10 :    6   -1
 2 :   10    6 :    6   -1
 3 :    9    7 :    5   -1
 4 :    9    6 :    4   -1
 5 :    9   11 :    7   -1
 6 :    8    9 :    5   -1
 7 :    7    6 :    3   -1
 8 :    7    8 :    5   -1
 9 :    7    9 :    4   -1
10 :    6    5 :    3   -1
11 :    6    3 :    9   -1
12 :    6
13 :    6
14 :    6
15 :    5
16 :    5
17 :    5
18 :    4
19 :    4
20 :    3    5 :    9   -1
21 :    3    1 :    1
22 :    3    2 :    2
23 :    3    4 :    8
24 :    2    3 :    1
25 :    2    1 :    0
26 :    2    4 :    2
27 :    1    2 :    1
28 :    1    0 :    0
29 :    0    2 :    0   -1
```

Trier en C :-)

```
int edgeCompare(const void* e0, const void *e1)
{
   int diagnostic = ((femEdge*) e0)->node[0] - ((femEdge*) e1)->node[0];
   if (diagnostic < 0)    return 1;
   if (diagnostic > 0)    return -1;
                          return 0;
}
```

Le tri implémenté est un algorithme très efficace !
Pour pouvoir accèder aux données de la structure, il faut un cast
Attention : C permet de caster n'importe quoi en n'importe quoi…

Be careful !

Mais, c'est pas tout à fait le bon tri à faire finalement !

Et détecter les segments frontières !

0	:	11	10	:	6	-1
1	:	9	11	:	7	-1
2	:	8	9	:	5	-1
3	:	7	9	:	4	5
4	:	7	8	:	5	-1
5	:	6	11	:	6	7
6	:	10	6	:	6	-1
7	:	9	6	:	4	7
8	:	6	7	:	4	3
9	:	5	7	:	3	-1
10	:	6	5	:	3	9
11	:	4	5	:	8	-1
12	:	6	3	:	9	-1
13	:	3	5	:	9	8
14	:	3	4	:	8	2
15	:	2	4	:	2	-1
16	:	3	2	:	2	1
17	:	3	1	:	1	-1
18	:	2	1	:	0	1
19	:	0	2	:	0	-1
20	:	1	0	:	0	-1