Application : Calcul de la Tour Perret de Grenoble

Modélisation par éléments poutre multibre

Sophie Capdevielle

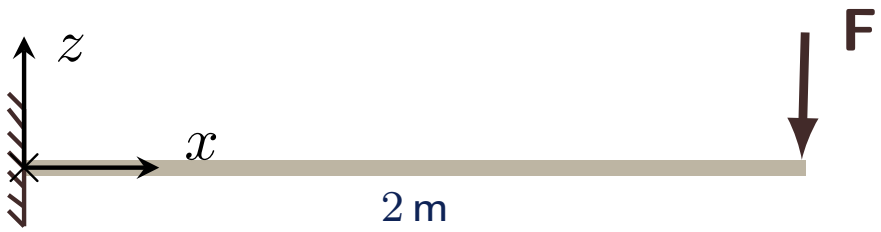
Workshop CSMA Juniors, Giens, 14 mai 2017

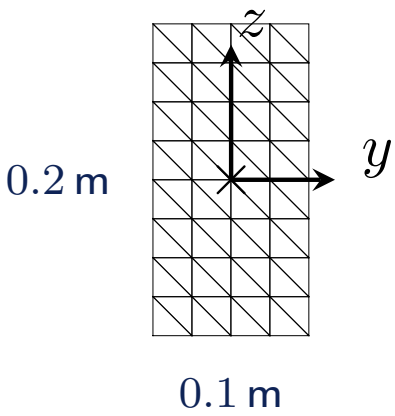
école normale supérieure paris–saclay

1. **Élément poutre multifibre**
   - Définition d'un exemple simple de calcul
   - Programmation de l'élément

2. **Application à la Tour Perret**
   - Définition du modèle pour un calcul linéaire
   - Résultats : déplacements sous l'effet du vent
   - Extension : calcul non linéaire

Exemple simple de calcul : poutre console



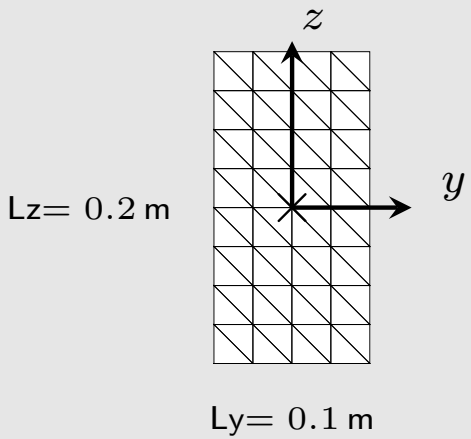


Discrétisation poutre : 2 éléments multifibres

```
%% Donnees du probleme global
nel = 2; %nombre d'elements
nnoeuds = 3; % nombre de noeuds
nddl_noeud = 6;
nddl = nnoeuds*nddl_noeud; % nombre ddl

% definition de la geometrie
Lelem=2/nel; % longueur de chaque element poutre
xyz=cell(nel,1);
for numel=1:nel
    xyz{numel} = [(numel-1)*Lelem,0,0;
                  numel*Lelem, 0, 0];
end
```

Maillage de la section

```
%% Definition de la section
% dimensions
Ly=0.1; Lz=0.2;
% nombre de segments selon y et z
ny=4; nz=8;
% Coordonnees des noeuds
nnoeudsec=0; %initialisation du numero de noeud
YZ=zeros((ny+1)*(nz+1),2);
for j=1:nz+1
    for i=1:ny+1
        nnoeudsec = nnoeudsec+1;
        YZ(nnoeudsec,1:2)=[(i-1)*Ly/ny-Ly/2, (j-1)*Lz/nz-Lz/2];
    end
end
clear nnoeudsec;
% Tableau de connection
nels=0;
CONs=cell(1,ny*nz*2);
for j=1:nz
    for i=1:ny
        nels=nels+2;
        square=[i+(ny+1)*(j-1), i+(ny+1)*(j-1)+1, i+(ny+1)*j, i+(ny+1)*j+1];
        CONs{nels-1}=[square(1),square(2),square(3)];
        CONs{nels}=[square(2),square(4),square(3)];
    end
end
% Association d'un/plusieurs materiau(x) aux elements de la section
MatType{1}=1:nels;
```



Affectation de leurs propriétés aux éléments

```
% Donnees elements
for numel = 1:nel
    ElemParam{numel}.yornt = [0; 1; 0]; %orientation de l'axe local y dans le repere global
    ElemParam{numel}.SecName = 'Sectimo3Dsimple';
    ElemParam{numel}.SecParam{1}.YZ = YZ;
    ElemParam{numel}.SecParam{1}.CONs = CONs;
    ElemParam{numel}.SecParam{1}.ElemType = 'tri3_simple';
    ElemParam{numel}.SecParam{1}.MatType = MatType; % id des elements de meme materiau



end
```

Affectation de leurs propriétés aux éléments

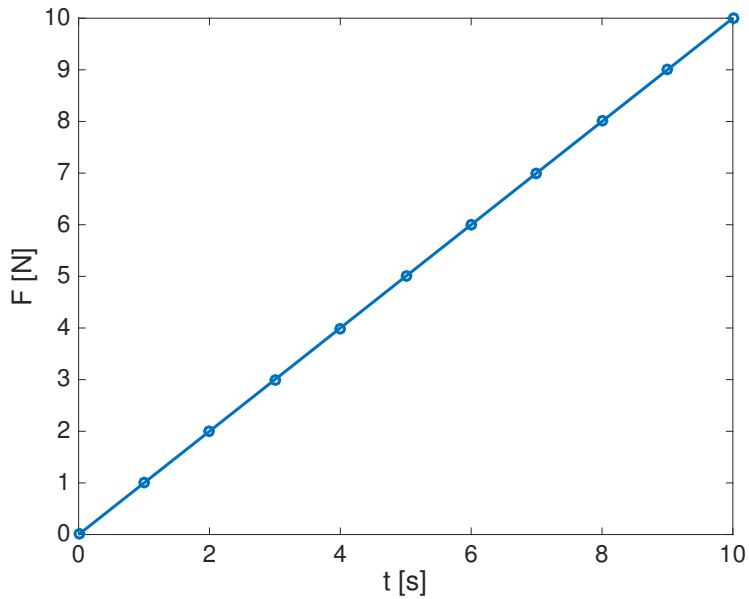
```
% Donnees elements
for numel = 1:nel
    ElemParam{numel}.yornt = [0; 1; 0]; %orientation de l'axe local y dans le repere global
    ElemParam{numel}.SecName = 'Sectimo3Dsimple';
    ElemParam{numel}.SecParam{1}.YZ = YZ;
    ElemParam{numel}.SecParam{1}.CONs = CONs;
    ElemParam{numel}.SecParam{1}.ElemType ='tri3_simple';
    ElemParam{numel}.SecParam{1}.MatType = MatType; % id des elements de meme materiau

    % Loi de comportement
    for m=1:length(CONs)
        ElemParam{numel}.SecParam{1}.MatName(1,:) = 'Elas1D'; % loi materiau
        ElemParam{numel}.SecParam{1}.MatParam{1}{m}.E0 = 210e9;
        ElemParam{numel}.SecParam{1}.MatParam{1}{m}.nu0 = 0.2;
    end
end
```

Conditions aux limites


```
%liste des ddl a CL en deplacement impose
idbc = 1:6;
% liste des numeros des ddl (sans CL)
idddl = 1:nddl;
idddl(idbc) = 0;
% liste complementaire d'idbc (liste des ddl a calculer)
idf = find(idddl~=0);

%table de connectivite
id = cell(nel,1);
for numel=1:nel
    id{numel} = (numel-1)*nddl_noeud+1 : (numel-1)*nddl_noeud+12;
end

% valeur des deplacements imposes au cours du t
uCL(idbc,:) = zeros(length(idbc),length(t));
```

Chargement


```
% Chargement
Tmax=10; dt=1; t=0:dt:Tmax;
F = zeros(nddl,length(t));
F(nel*nddl_noeud+3,:) = t;
% Adaptation du chargement pour enlever les ddl bloques
f = F(idf,:);
```



Résolution

Méthode de Newton Raphson

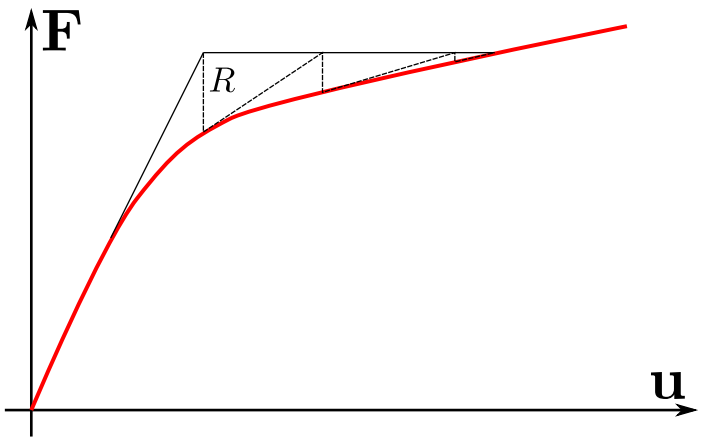
```
1  for pas=1:length(t)-1
2      %% initialisation de la boucle de Newton
3      uk=u(:,pas); % vecteur des ddl
4
5      % Affectation des ddl a chaque element
6      for numel = 1:nel
7          ElemDisp.uel = Affectation(id{numel},uCL(:,↩
               pas+1),idbc,uk,idf);
8          ElemDisp.Duel = zeros(length(id{numel}),1);
9          ElemDisp.DDuel = zeros(length(id{numel}),1)↩
               ;
10         [pel(:,numel),kel(:,:,numel),Hist{numel↩
               }]=...
11             timo3Dsimple(entrees); % entrees vues ↩
                   plus tard dans le cours
12     end
13     % Assemblage des efforts et de la mat. de ↩
           raideur
14     [pf,kf]=Assemblage(nddl,id,pel,kel);
15     % Prise en compte des CL:
16     %suppr. des lignes et des col.(depla. nuls)
17     p=pf(idf); k=kf(idf,idf);
18     % Calcul du residu
19     R=-p+f(:,pas+1);
20     kp=k;
```

À chaque pas de chargement :

Résolution de

$$P(u) = f$$

par la méthode de Newton



Résolution

Méthode de Newton-Raphson

```
%% Boucle de Newton Raphson
while norm(R)>1e-5
% Calcul de l'increment de la nouvelle cinematique
    du=kp^(-1)*R;
    uk=uk+du;
% Affectation des ddl a chaque element
    for numel = 1:nel
    ElemDisp.uel = ...
    Affectation(id{numel},uCL(:,pas+1),idbc,uk,idf)↩
        ;
    % construction matrices elem (vu apres)
    [pel(:,numel),kel(:,:,numel),Hist{numel}]=...
        timo3Dsimple(entrees);
    end
% Assemblage des efforts et de la mat. de raideur
    [pf,kf]=Assemblage(nddl,id,pel,kel);
% Prise en compte des CL:
% suppr. des lignes et des col.(depl. nuls)
    p=pf(idf);
    k=kf(idf,idf);
% Calcul du residu
    R=-p+f(:,pas+1);
    kp=k;
end
```

À chaque itération :

$$U^{(k+1)} = U^{(k)} + k_p^{-1} R$$

Où $R = f - p\left(U^{(k)}\right)$

p, k_p calculés par assemblage des vecteurs et matrices élémentaires

Assemblage

```
function [pf,kf]=Assemblage(nnd,id,pel,kel)
    pf=zeros(nnd,1);
    kf=zeros(nnd,nnd);
    for jel=1:length(id) % boucle sur les elements
        %Assemblage
        pf(id{jel})=pf(id{jel})+pel(:,jel);
        kf(id{jel},id{jel})=kf(id{jel},id{jel})+kel(:,:,jel);
    end
end
```

Élément poutre multifibre

```
function [p,S,ElemHist] = ...
    timo3Dsimple(Action,elno,xyz,ElemParam,ElemDisp,ElemHist)
```

- Entrées
  - Action : 'init', 'raideur', 'masse', 'ldc'...
  - Numéro de l'élément
  - Coordonnées des noeuds dans le repère global
  - Déplacements aux noeuds de l'élément
  - Paramètres de l'élément
  - Variables historiques du pas de temps convergé précédent
- Sorties
  - Efforts aux noeuds de l'élément
  - Matrice de raideur élémentaire
  - Variables historiques actualisées (servent pour le prochain pas de temps convergé)

Calcul des déformations généralisées

Déplacements et rotations aux noeuds

```
% Deplacements, increments de deplacements aux noeuds
u=ElemDisp.uel; % deplacements
du=ElemDisp.Duel; % increments de deplacements depuis le dernier pas de temps cvge
ddu=ElemDisp.DDuel; % increments de deplacements depuis la derniere iteration

% Passage du repere global au repere local de l'element
[xornt,L]=geomelem(xyz);
yornt=ElemParam{elno}.yornt; yornt=yornt/norm(yornt);
zornt=cross(xornt,yornt);

P=[xornt,yornt,zornt];
ab=[P^-1,zeros(3,9);
    zeros(3,3),P^-1,zeros(3,6);
    zeros(3,6),P^-1,zeros(3,3);
    zeros(3,9),P^-1];

% Deplacements et increments dans le repere local de l'elt
ub=ab*u;
dub=ab*du;
ddub=ab*ddu;
```

Calcul des déformations généralisées

$$\mathbf{e}_s = \boldsymbol{B}_p \mathbf{U}_{el}^p$$

- Avec fonctions d'interpolation linéaires
- Termes de cisaillement sous-intégrés [Pegon, 1994]

$$\mathbf{e}_s = \begin{bmatrix} -\frac{1}{L^e} & 0 & 0 & 0 & 0 & 0 & \frac{1}{L^e} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{L^e} & 0 & 0 & 0 & -\frac{1}{2} & 0 & \frac{1}{L^e} & 0 & 0 & 0 & -\frac{1}{2} \\ 0 & 0 & -\frac{1}{L^e} & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{L^e} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & -\frac{1}{L^e} & 0 & 0 & 0 & 0 & 0 & \frac{1}{L^e} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{L^e} & 0 & 0 & 0 & 0 & 0 & \frac{1}{L^e} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{L^e} & 0 & 0 & 0 & 0 & 0 & \frac{1}{L^e} \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ w_1 \\ \theta_{x\,1} \\ \theta_{y\,1} \\ \theta_{z\,1} \\ u_2 \\ v_2 \\ w_2 \\ \theta_{x\,2} \\ \theta_{y\,2} \\ \theta_{z\,2} \end{bmatrix}$$

Calcul des déformations généralisées

$$\mathbf{e}_s = \boldsymbol{B}_p \mathbf{U}_{el}^p$$

- Avec fonctions d'interpolation linéaires
- Termes de cisaillement sous-intégrés [Pegon, 1994]

```
% Derivees des fonctions d'interpolation,
% avec cisaillement sous-integre
B=[diag(-1/L*ones(6,1)),diag(1/L*ones(6,1))];
B(2,6)=-1/2;
B(2,12)=-1/2;
B(3,5)=1/2;
B(3,11)=1/2;

%Deformations generalisees et increments dans la section
es=B*ub;
des=B*dub;
ddes=B*ddub;

SecDisp.es=es;
SecDisp.des=des;
SecDisp.ddes=ddes;
```

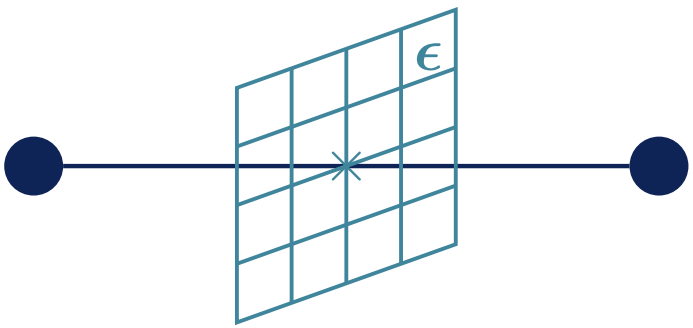
Passage dans la section

```
% Recuperation des donnees de la section
SecParam=ElemParam{elno}.SecParam{1};

% Passage des variables historiques
SecHist.Pres=ElemHist.Pres.Sec{1};
SecHist.Past=ElemHist.Past.Sec{1};

% Appel de la section
[Ps,Ks,SecHist] = feval (ElemParam{elno}.SecName,Action,...
    SecParam,SecDisp,SecHist);
```

- Entrées
  - Action : 'init', 'raideur', 'masse', 'ldc'...
  - Paramètres de la section
  - Déformations généralisées
  - Variables historiques
- Sorties
  - Effort résistant de la section
  - Matrice de raideur de la section
  - Variables historiques actualisées (servent pour le prochain pas de temps convergé)

Passage dans la section

Calcul des déformations ϵ dans chaque fibre



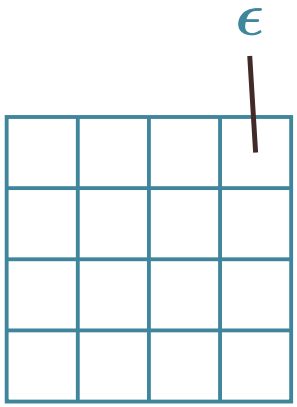
$$\begin{bmatrix} \epsilon_{xx} \\ 2\epsilon_{xy} \\ 2\epsilon_{xz} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & z & -y \\ 0 & 1 & 0 & -z & 0 & 0 \\ 0 & 0 & 1 & y & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{du}{dx} \\ \frac{dv}{dx} - \theta_z \\ \frac{dw}{dx} + \theta_y \\ \frac{d\theta_x}{dx} \\ \frac{d\theta_y}{dx} \\ \frac{d\theta_z}{dx} \end{bmatrix}$$

$$\boldsymbol{\epsilon} = \mathbf{a}_s(y,z)\,\mathbf{e}_s(x)$$

```
for k=1:length(MatType) %number of materials
    for m=MatType{k} % loop over all elements of material k

        %%% Geometrie des fibres
        [~,~,ElemHist] = feval(ElemType,'init',[],m,YZ(CONs{m},:),[],[],[]);
        y   = ElemHist.YZp(1); z   = ElemHist.YZp(2);
        % matrice de passage des deformations generalisees aux
        % deformations des fibres
        as=[eye(3),[0 z -y; -z 0 0; y 0 0]];
        %%% Calcul des deformations increments de deformations,
        eps = as*es;
        deps = as*des;
        ddeps = as*ddes;
```

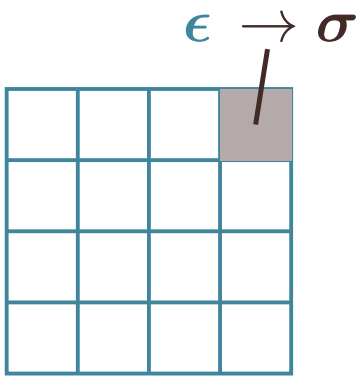
Passage dans la section

Calcul des contraintes σ dans chaque fibre



Passage dans la section

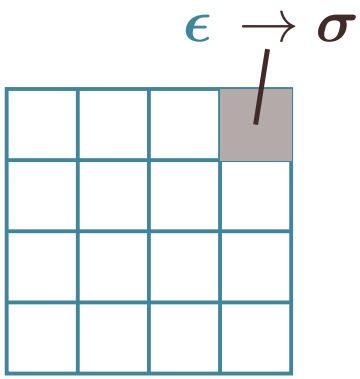
Calcul des contraintes σ dans chaque fibre




```
%% Appel de la loi materiau
% affectation des deformations
MatEps.eps = [eps,deps,ddeps];
% variables historiques pas de temps converge precedent
MatEtat.Past  = SecHist.Past.Mat{m};
% appel de la loi materiau
[~, ~, MatEtat] = feval (MatName(k,:),'stif',MatParam{k}{m},MatEps,MatEtat);
% Matrice de "raideur" secante du materiau
Kmat = MatEtat.Pres.Ct;
% contraintes dans la fibre
Sig = MatEtat.Pres.sig;
```

Passage dans la section

Calcul des contraintes σ dans chaque fibre



Matériau

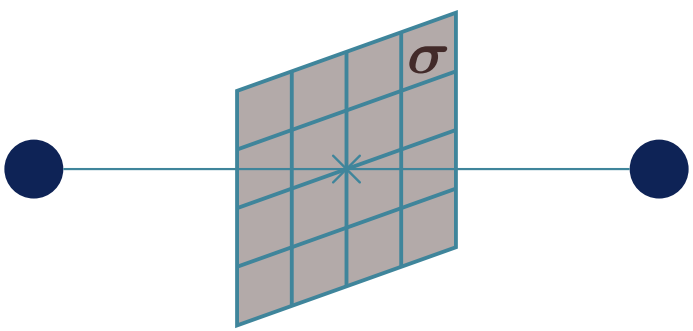
- Linéaire élastique
- Loi de comportement 1D
- $E = 210\,\text{GPa}$

```
1  function [sig,Kt,MatEtat] = Elas1D (MatParam,MatEps,MatEtat)
2  %Elas1D Loi de comportement elastique lineaire 1D
3  E0  = MatParam.E0;     % Module de Young
4  nu0 = MatParam.nu0;    % coefficient de Poisson
5
6  eps = MatEps.eps;
7  sig = [E0*eps(1);E0/(2*(1+nu0))*eps(2);E0/(2*(1+nu0))*eps(3)];
8  Kt  = [E0 0 0; 0 E0/(2*(1+nu0)) 0; 0 0 E0/(2*(1+nu0))];
9  MatEtat.Pres.sig = sig;
10 MatEtat.Pres.Ct  = Kt;
11 end
```

Passage dans la section

Calcul des efforts généralisés et matrice de raideur de la section

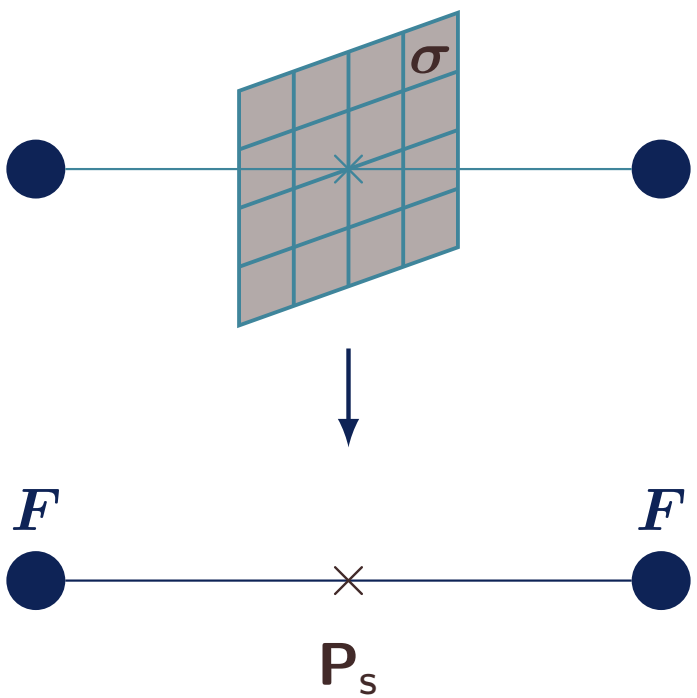
```
%%% Matrice de raideur de la section Ks
Ks=Ks+Aw*as'*Kmat*as;

%%% Efforts generalises de la section
fs=fs+Aw*as'*Sig;
```



Passage dans la section

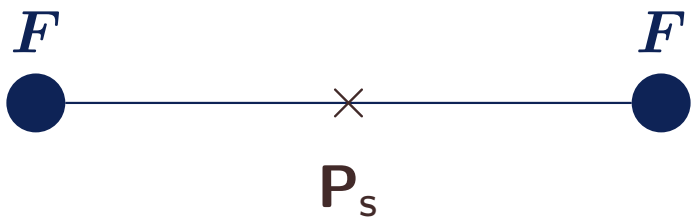
Calcul des efforts généralisés et matrice de raideur de la section

```
%%% Matrice de raideur de la section Ks
Ks=Ks+Aw*as'*Kmat*as;

%%% Efforts generalises de la section
fs=fs+Aw*as'*Sig;
```



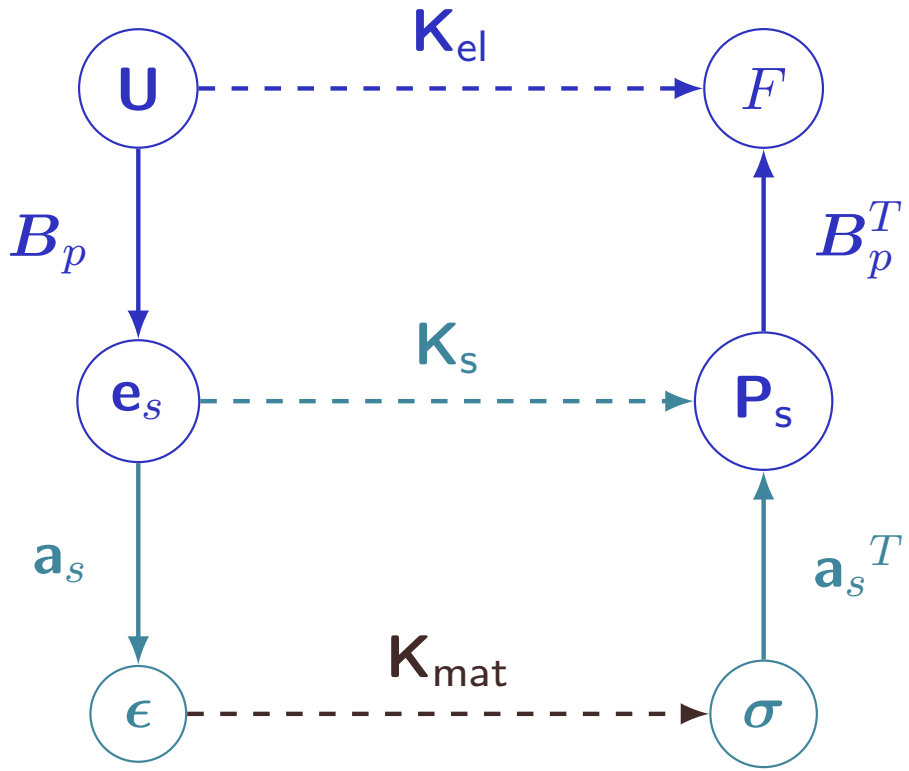
Retour à l'élement

Calcul des efforts généralisés aux noeuds et matrice de raideur de l'élément




```
% Matrice de raideur (secante) de l'elt:
kb=w*L/2*B'*Ks*B; % repere local

% Effort resistant
pb=w*L/2*B'*Ps;

% Matrice de raideur
Sb=coeffdyn(1)*kb;

% Passage en coordonnees globales
S=ab'*Sb*ab;
p=ab'*pb;
```

Bilan



- - - -► Échelle de l'élément

- - - -► Échelle de la section

- - - -► Échelle du point d'intégration de la section

À vous de jouer !

1. Élément poutre multifibre
   - Définition d'un exemple simple de calcul
   - Programmation de l'élément

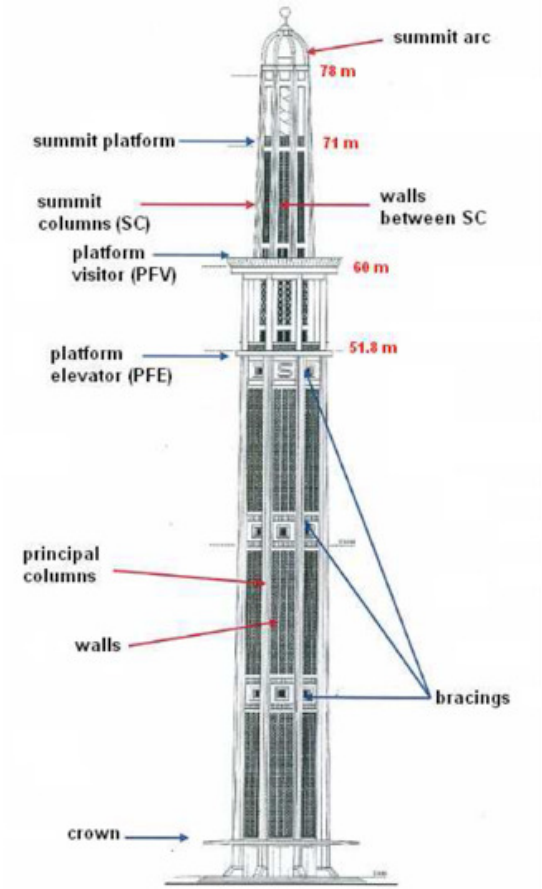
2. **Application à la Tour Perret**
   - Définition du modèle pour un calcul linéaire
   - Résultats : déplacements sous l'effet du vent
   - Extension : calcul non linéaire

Tour Perret de Grenoble

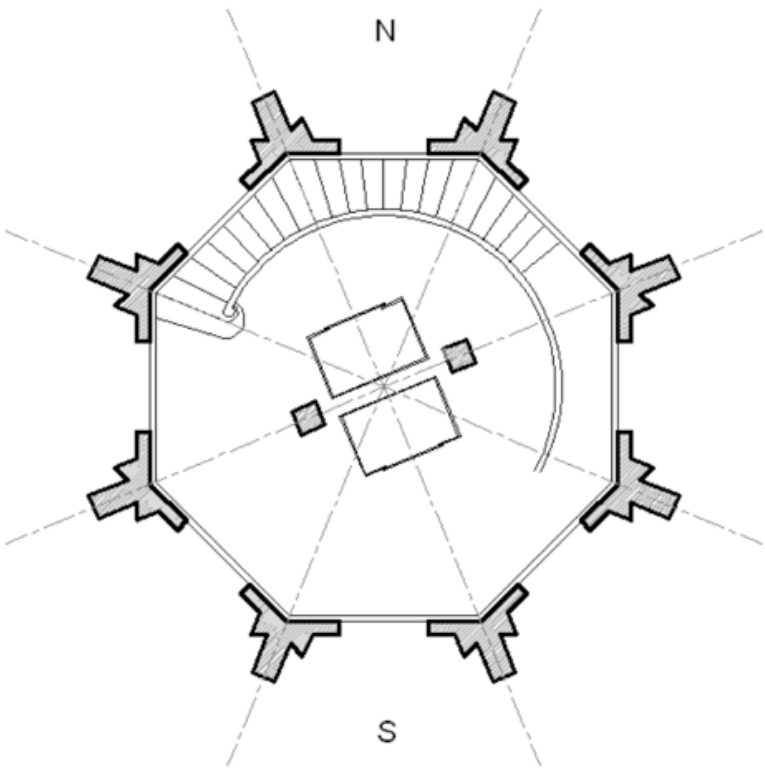
- Hauteur $= 76\,\mathrm{m}$
- Diamètre moyen $= 7\,\mathrm{m}$

Croquis de la Tour Perret

extrait de [Omar, 2011], fourni par la Mairie de Grenoble



(a)



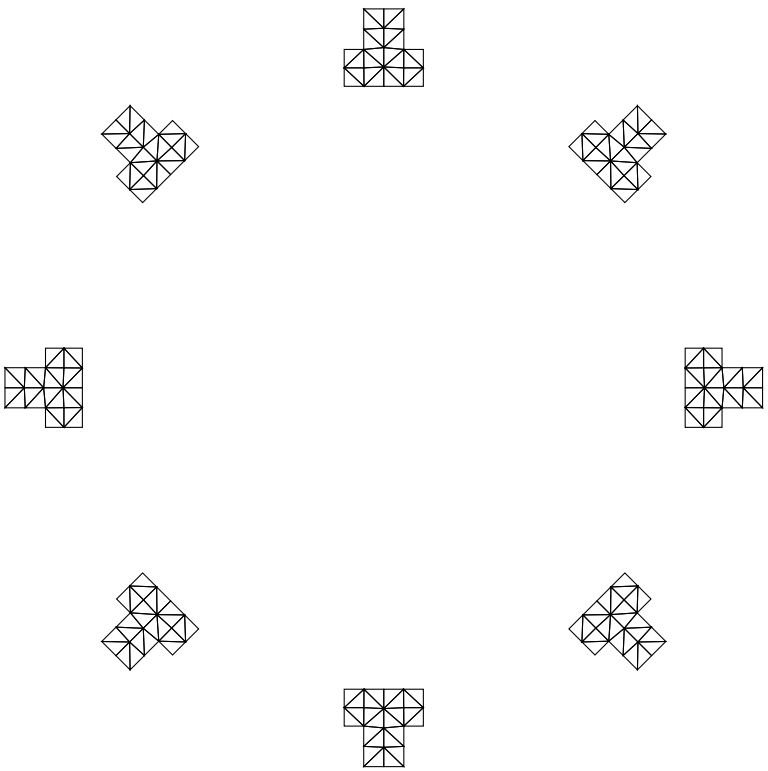
(b)

Discrétisation en éléments poutre

```
%% Maillage de la Tour en elements poutre

%%% Definition de la geometrie: Discretisation en poutres
H=[3.8 3.05 3.05 3.05 3.05 3.8 3.05 3.05 3.05 3.05 3.8 3.05...
       3.05 3.05 3.05 3.8 4.1 4.1 4.1 3.05 3.8 5.5];
   % longueur de chaque elt poutre, noeud 18 correspond a la plateforme
nel = length(H); % nb d'elts poutres
nnoeud = length(H) + 1;
nddl_noeud = 6;
nddl = nnoeud*nddl_noeud; % nombre ddl

%%% Coordonnees des noeuds
xyz = cell(nel,1);
xyz{1} = [0, 0, 0;
          0, 0, H(1)];
for numel = 2:nel
    xyz{numel} = [0, 0, sum(H(1:numel-1));
                  0, 0, sum(H(1:numel))];
end

%table de connectivite
id = cell(nel,1);
for numel=1:nel
    id{numel} = (numel-1)*nddl_noeud+1 : (numel-1)*nddl_noeud+12;
end
```

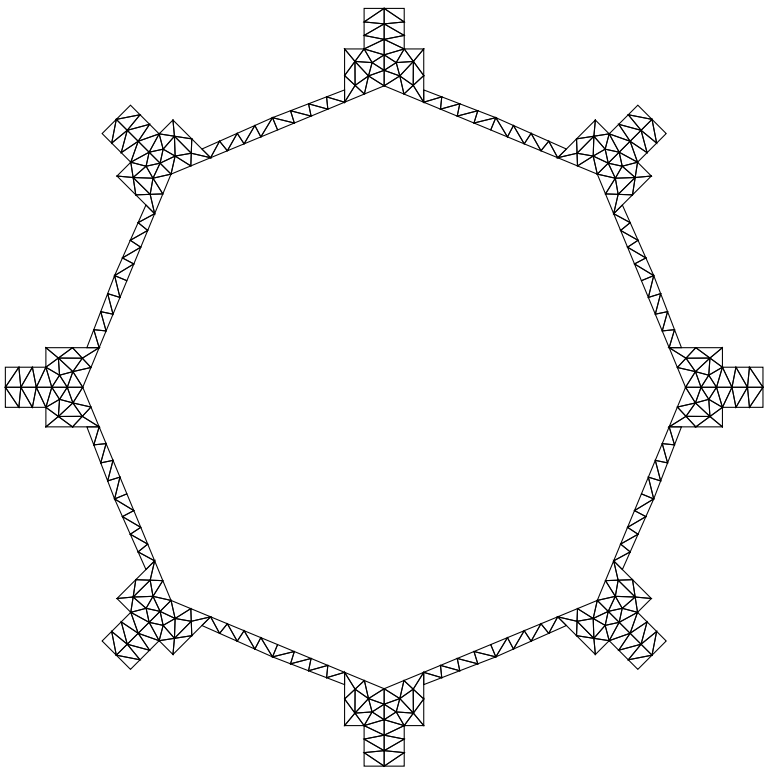
Maillages de la section

Poteaux



$f_1 = 0{,}63\,\text{Hz}$ (avec $E = 34\,\text{GPa}$)

Poteaux + Claustra



$f_1 = 0{,}75\,\text{Hz}$ (avec $E = 34\,\text{GPa}$)

Première fréquence propre mesurée sur la structure : 0,75 Hz direction NS [Grange, 2011]

Tests du code

- Chargement de vent
  - Défini à partir du chargement réglementaire, région de Grenoble
- Trois types de maillage

Modèle matériau béton non linéaire

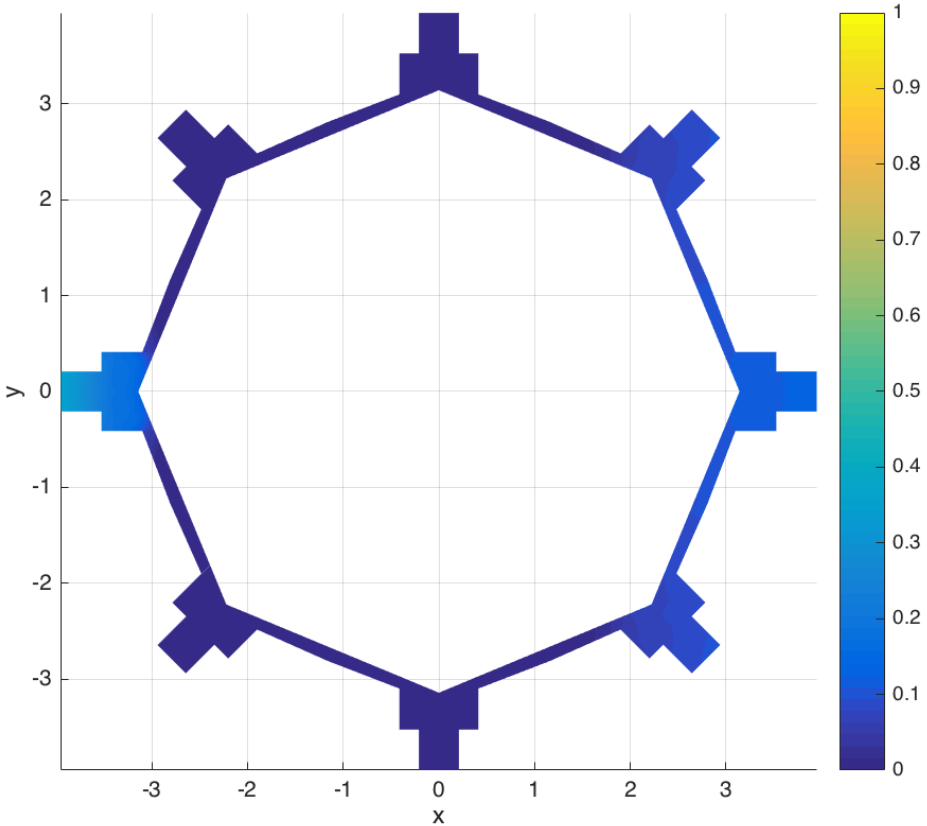
Paramètres à définir :

- Module d'Young E
- Coefficient de Poisson ν
- Seuils d'endommagement y_{c0}, y_{t0}
- Paramètres A_c, B_c, A_t, B_t gouvernant la forme du comportement après endommagement

```
ElemParam{numel}.SecParam{1}.MatName(1,:) = 'MU3Dwk'; % loi materiau
for m=1:length(CONs)
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.e = 34e9;
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.ANU = 0.17;
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.a1 = 1;
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.a2 = 0.85;
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.b1 = 11000;
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.b2 = 490;
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.yt = 4e6;
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.yc = -2e6;
    ElemParam{numel}.SecParam{1}.MatParam{1}{m}.k1 = 1;
end
```

Résultats : endommagement dans la section

- Chargement :
  - 5 × chargement de vent
  - Appliqué en 20 pas
- Endommagement à la base de la tour, au pas 13 :



Limites de la modélisation présentée

Aspects non modélisés, à prendre en compte :

- Chargement gravitaire
- Aciers longitudinaux

Améliorations possibles

- Variation de la section le long de la tour
- Introduction de l'alea matériau