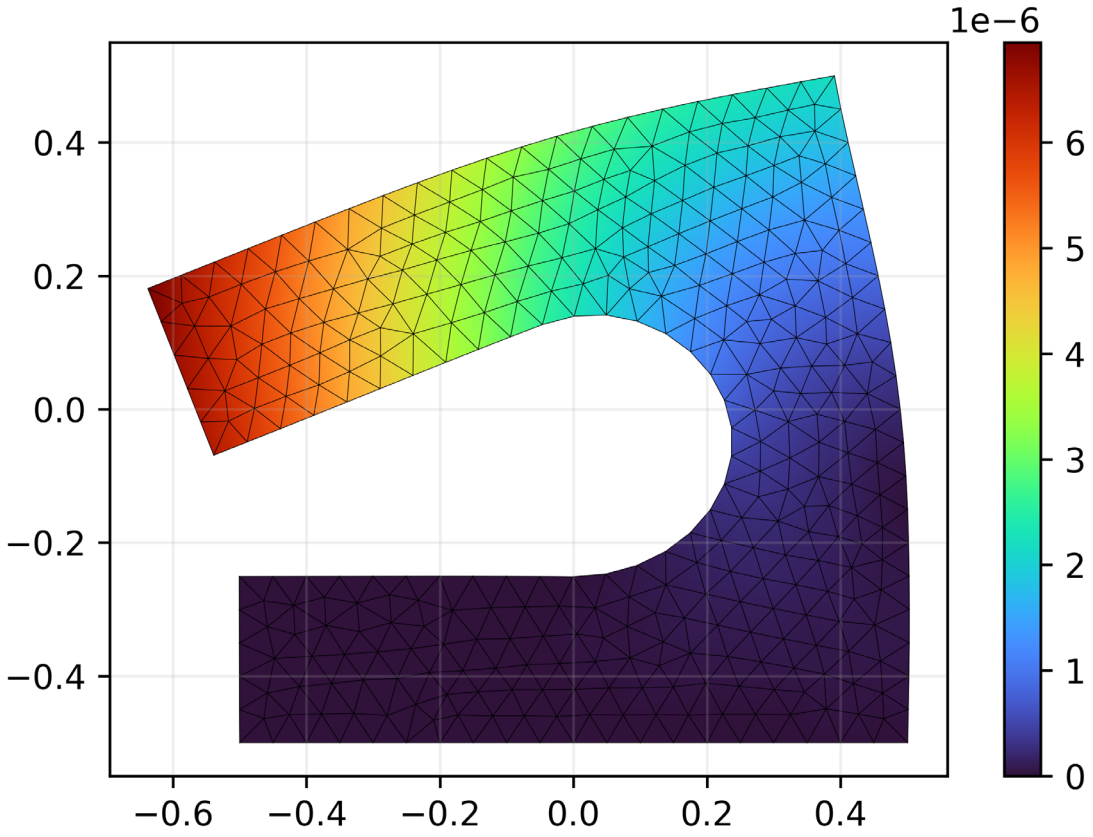


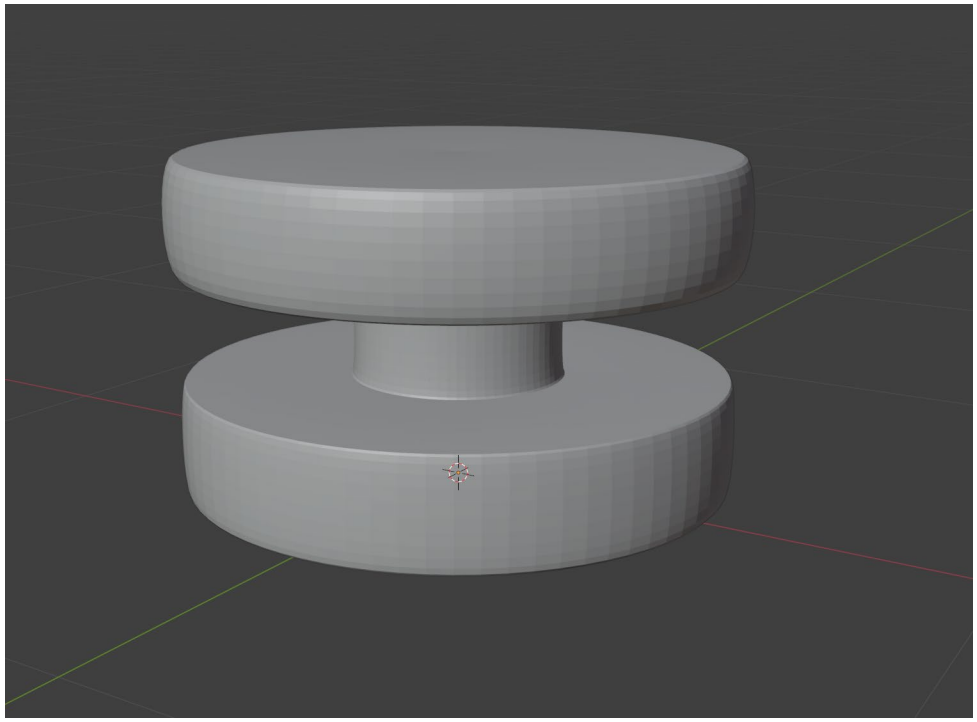

# Avoir 25/20 au projet

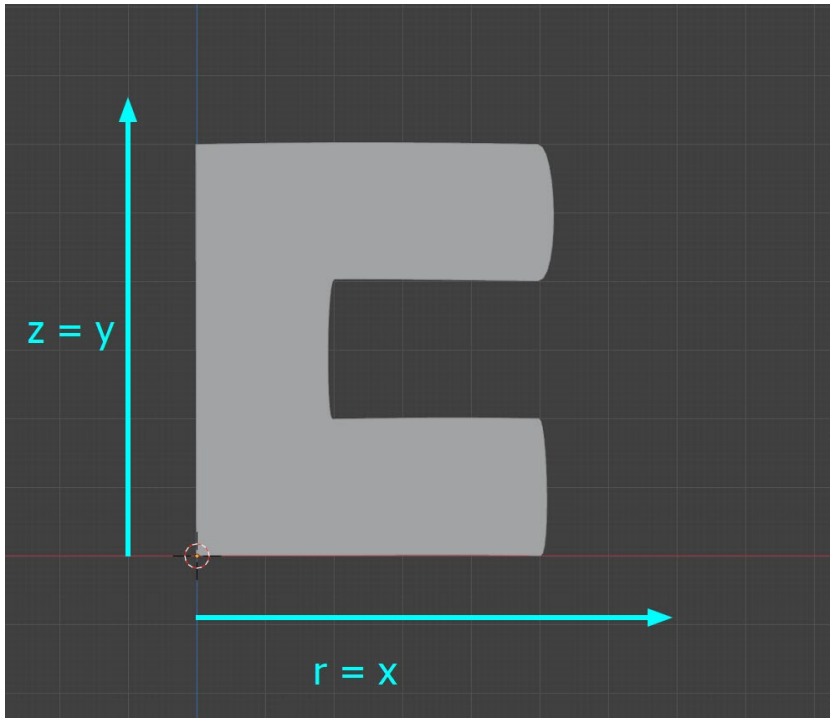
Fiction absurde ou réalité tangible?

# Plan

- L’axisymétrique
- Le calcul des tensions aux nœuds
- Générer un beau maillage
- Le canvas 2.0 du projet
- Le calcul des normales
- Les solveurs feat comment battre le groupe 83
- Des figures de qualité en python

# 1 — L’axysymétrique





- Rappel : solide élastique

$$\underline{\underline{\boldsymbol{\varepsilon}}} \triangleq \frac{1}{2}\left(\nabla \underline{\boldsymbol{u}} + \nabla \underline{\boldsymbol{u}}^T\right)$$

$$\underline{\underline{\boldsymbol{\sigma}}} = 2\mu \underline{\underline{\boldsymbol{\varepsilon}}} + \lambda \operatorname{Tr}\left(\underline{\underline{\boldsymbol{\varepsilon}}}\right) \underline{\underline{\boldsymbol{\delta}}}$$

$$\nabla \cdot \underline{\underline{\boldsymbol{\sigma}}} + f = 0$$

- En coordonnées cylindriques, l'expression du gradient change !

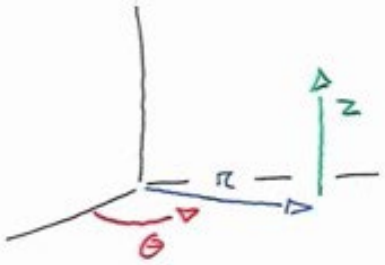
https://en.wikipedia.org/wiki/Del_in_cylindrical_and_spherical_coordinates

# AXISYMMETRIC PROBLEMS

AXISYMMETRY !

$u_\theta = 0$

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



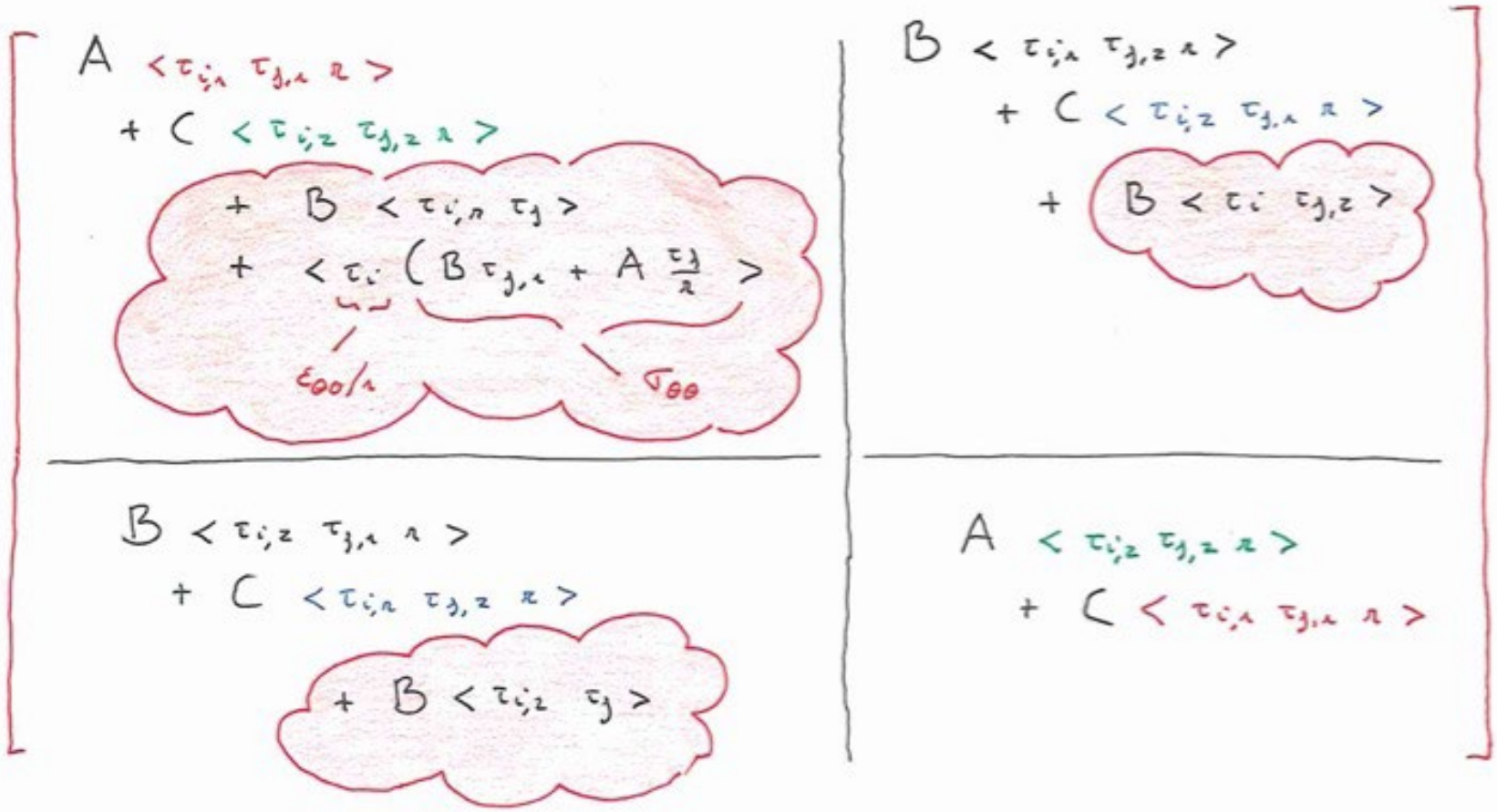

$$\underline{u} = \begin{bmatrix} u_r \\ u_\theta \\ u_z \end{bmatrix} = \begin{bmatrix} u(r,z) \\ 0 \\ w(r,z) \end{bmatrix}$$

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

A, B, C OF PLANAR DEFORMATIONS

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

# IT IS ALMOST LIKE A 2D PROBLEM !



# Et donc concrètement?

$$\boldsymbol{A}_{ij} = \left[\begin{array}{c|c} \langle \tau_{i,r} A \tau_{j,r} \cdot r \rangle + \langle \tau_{i,z} C \tau_{j,z} \cdot r \rangle & \langle \tau_{i,r} B \tau_{j,z} \cdot r \rangle + \langle \tau_{i,z} C \tau_{j,r} \cdot r \rangle \\ + \langle \tau_{i,r} B \tau_j \rangle + \left\langle \tau_i \left( B \tau_{j,r} + A \frac{\tau_j}{r} \right) \right\rangle & + \langle \tau_i B \tau_{j,z} \rangle \\ \hline \langle \tau_{i,z} B \tau_{j,r} \cdot r \rangle + \langle \tau_{i,r} C \tau_{j,z} \cdot r \rangle & \langle \tau_{i,z} A \tau_{j,z} \cdot r \rangle \\ + \langle \tau_{i,z} B \tau_j \rangle & + \langle \tau_{i,r} C \tau_{j,r} \cdot r \rangle \end{array}\right]$$

$$\boldsymbol{B}_i = \begin{bmatrix} \langle \tau_i f_r \cdot r \rangle + \langle\!\langle \tau_i g_r \cdot r \rangle\!\rangle \\ \langle \tau_i f_z \cdot r \rangle + \langle\!\langle \tau_i g_z \cdot r \rangle\!\rangle \end{bmatrix}$$

# Et donc concrètement?

$$\boldsymbol{A}_{ij} = \left[\begin{array}{c|c} \langle \tau_{i,x} A \tau_{j,x} \cdot x \rangle + \langle \tau_{i,y} C \tau_{j,y} \cdot x \rangle & \langle \tau_{i,x} B \tau_{j,y} \cdot x \rangle + \langle \tau_{i,y} C \tau_{j,x} \cdot x \rangle \\ + \langle \tau_{i,x} B \tau_j \rangle + \left\langle \tau_i \left( B \tau_{j,x} + A \frac{\tau_j}{r} \right) \right\rangle & + \langle \tau_i B \tau_{j,y} \rangle \\ \hline \langle \tau_{i,y} B \tau_{j,x} \cdot x \rangle + \langle \tau_{i,x} C \tau_{j,y} \cdot x \rangle & \langle \tau_{i,y} A \tau_{j,y} \cdot x \rangle \\ + \langle \tau_{i,y} B \tau_j \rangle & + \langle \tau_{i,x} C \tau_{j,x} \cdot x \rangle \end{array}\right]$$

$$\boldsymbol{B}_i = \begin{bmatrix} \langle \tau_i f_x \cdot x \rangle + \langle\!\langle \tau_i g_x \cdot x \rangle\!\rangle \\ \langle \tau_i f_y \cdot x \rangle + \langle\!\langle \tau_i g_y \cdot x \rangle\!\rangle \end{bmatrix}$$

# 2 — Obtenir sigma aux nœuds

$$\underline{\underline{\boldsymbol{\varepsilon}}} \triangleq \frac{1}{2}\left(\nabla \underline{\boldsymbol{u}} + \nabla \underline{\boldsymbol{u}}^T\right)$$

$$\underline{\underline{\boldsymbol{\sigma}}} = 2\mu \underline{\underline{\boldsymbol{\varepsilon}}} + \lambda \operatorname{Tr}\left(\underline{\underline{\boldsymbol{\varepsilon}}}\right) \underline{\underline{\boldsymbol{\delta}}}$$

Sigma est fonction du gradient de la déformation

$\nabla \underline{\boldsymbol{u}}^h$ Est discontinu entre les éléments!

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

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

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

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

CFR LINEARITY !

? $\underline{u}^h$

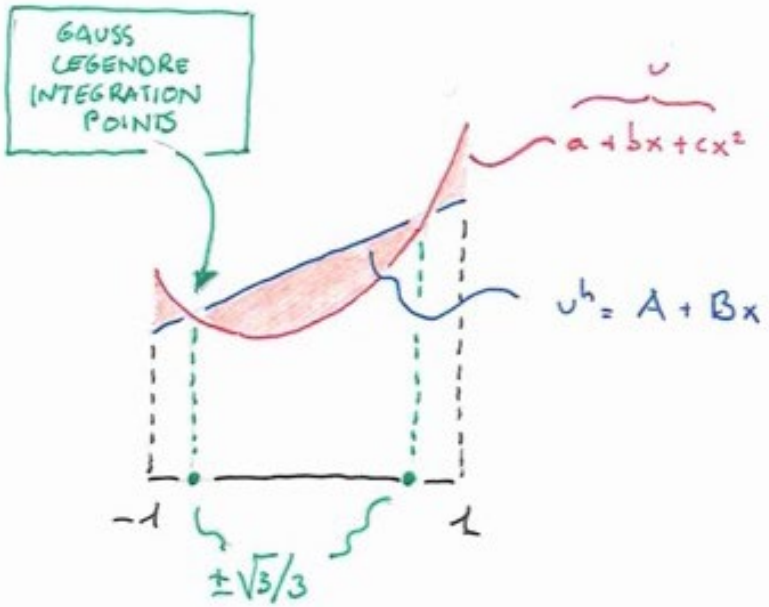
$a(\hat{\underline{v}}^h, \underline{u}^h - \underline{u}) = 0$
$\forall \hat{\underline{v}}^h \in \hat{\mathcal{U}}^h$

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

? $\underline{u}^h$

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

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




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

? $A, B$ SUCH THAT

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

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

$$\begin{cases} B^2\frac{2}{3} - Bb\frac{2}{3} = 0 \rightarrow B = b \\ 2A^2 - 2Aa - Ac\frac{2}{3} = 0 \rightarrow A = a + \frac{c}{3} \end{cases}$$

# Et donc concrètement?

$$\sum_j \underbrace{\langle \tau_i \tau_j \rangle}_{M_{ij}} \varepsilon_{xx}^j = \left\langle \tau_i \frac{\partial u^h}{\partial x} \right\rangle$$

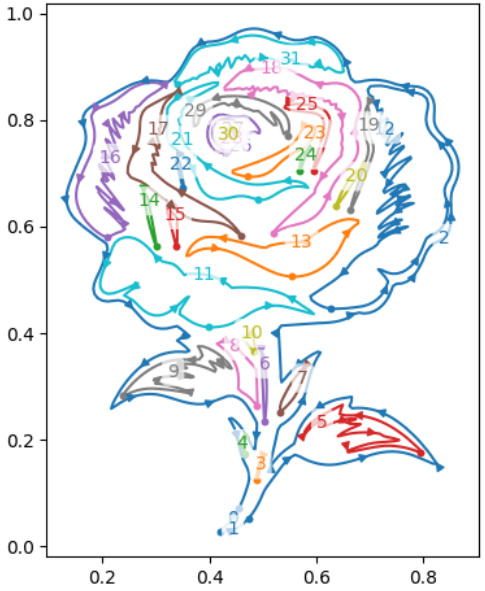
$$\sum_j \underbrace{\langle \tau_i \tau_j \rangle}_{M_{ij}} \varepsilon_{yy}^j = \left\langle \tau_i \frac{\partial v^h}{\partial y} \right\rangle$$

$$\sum_j \underbrace{\langle \tau_i \tau_j \rangle}_{M_{ij}} \varepsilon_{xy}^j = \frac{1}{2} \left\langle \tau_i \left( \frac{\partial u^h}{\partial y} + \frac{\partial v^h}{\partial x} \right) \right\rangle$$

Dans la partie “PostProcessor”

# 3 — Générer un maillage complexe

- Installer gmsh : *le module python*
- Premier loop = loop externe
- Orientation des courbes!
- Échelle!




https://github.com/MiguelDLC/femtools/

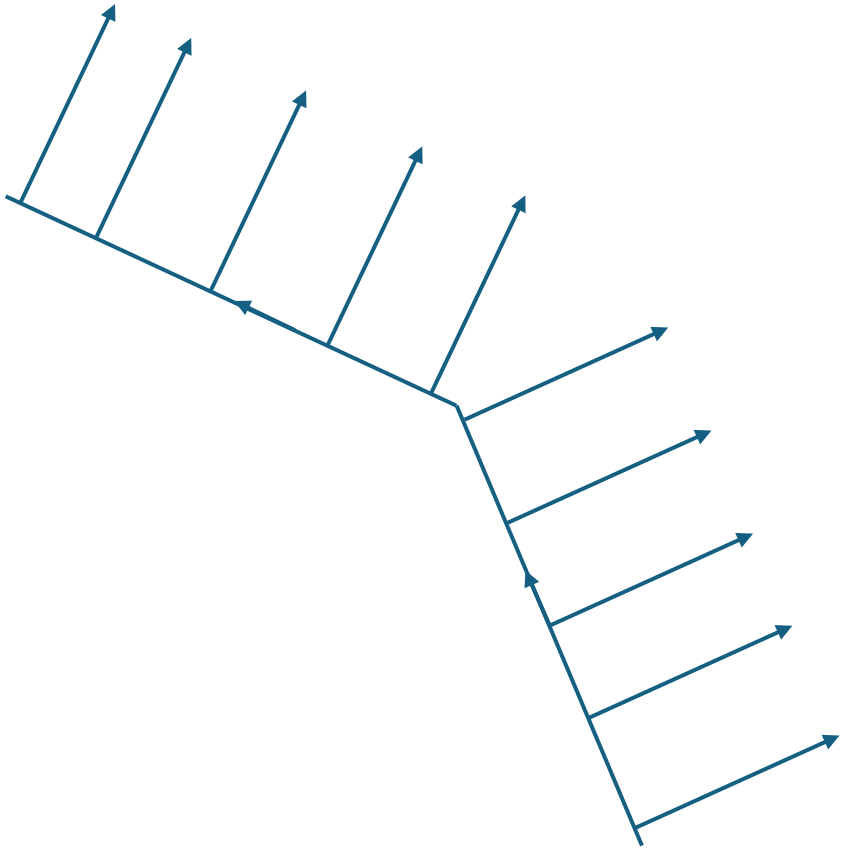
# 4 — Le canvas du projet (enfin)

- Le seul dossier qui compte : Project
- Veiller à **SCRUPULEUSEMENT** respecter les path, sous peine de voir la correction automatique vous donner un zéro

- Dans la soumission finale, ne pas remettre
  - gmsh
  - build
  - __pycache__
  - .vscode
  - .idea
  - …

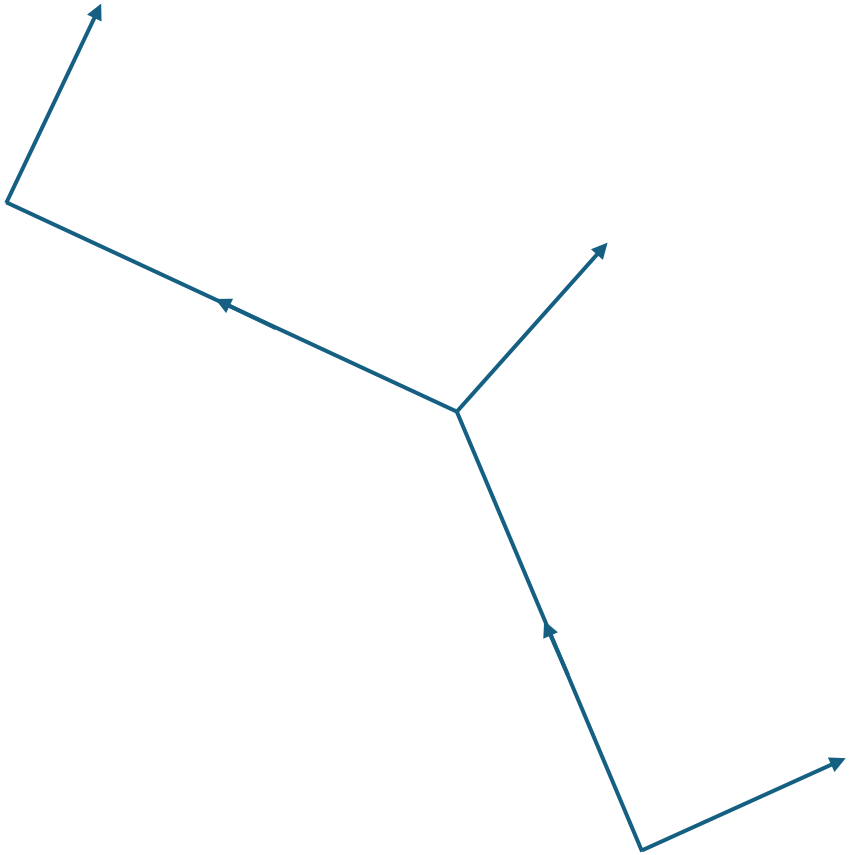
# 5 — Le calcul des normales

NEUMANN



Normale calculée au segment

DIRICHLET



Normale calculée au nœud
(On l’a fait pour vous)

# 6 — Les solveurs*

| Solveur | Complexité | Vitesse | Notes |
|---|---|---|---|
| Dense classique | Déjà fourni | -- | |
| Bande no renum | Facile | - | |
| Bande renum X/Y | Moyen | + | Très honnête |
| Bande + RCMK | Moyen + | ++ | |
| Frontal | Hard | ++ | Gr 07 |
| Conjugate Gradient | Facile | --- | Pire que tout |
| CG + CSR | Moyen- | +++ | |
| Direct sparse + AMD | Hard | ++++ | Gr 83 |
| Multigrid? | HARD | ? | Optimal en théorie |

https://github.com/MiguelDLC/femtools/

*Vous voulez faire du parallèle ou des choses bizzaroides? C'est possible à condition que ce soit validé par Miguel

# Les gradients conjugués

$\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$
if $\mathbf{r}_0$ is sufficiently small, then return $\mathbf{x}_0$ as the result
$\mathbf{p}_0 := \mathbf{r}_0$
$k := 0$
repeat

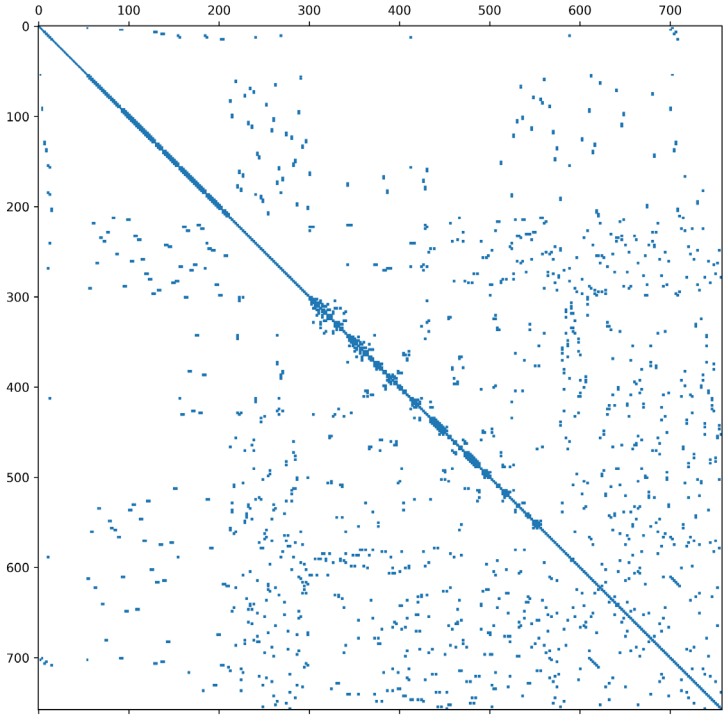
$$\alpha_k := \frac{\mathbf{r}_k^\mathsf{T}\mathbf{r}_k}{\mathbf{p}_k^\mathsf{T}\mathbf{A}\mathbf{p}_k}$$

$\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{p}_k$
$\mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{p}_k$
if $\mathbf{r}_{k+1}$ is sufficiently small, then exit loop

$$\beta_k := \frac{\mathbf{r}_{k+1}^\mathsf{T}\mathbf{r}_{k+1}}{\mathbf{r}_k^\mathsf{T}\mathbf{r}_k}$$

$\mathbf{p}_{k+1} := \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$
$k := k + 1$
end repeat
return $\mathbf{x}_{k+1}$ as the result

Produit A·p




Matrice est essentiellemnt vide!

https://en.wikipedia.org/wiki/Conjugate_gradient_methoc

# 7 — Animations python