Maillage et Éléments Finis

Bertrand Thierry

avr. 24, 2023

Table des matières :

Le code source du cours est disponible librement sur github, n'hésitez pas à envoyer des remarques, typos, corrections, … !

La première section présente la théorie et la mise en oeuvre des éléments finis. La deuxième partie propose un exemple d'implémentation de la méthode des éléments en utilisant le language Python et le logiciel GMSH pour la partie maillage.

Première partie

Notes de cours

Remarques préliminaires :

- Si la théorie mathématique ne vous branche pas, vous pouvez passer directement à la section pratique et revenir plus tard sur la théorie, pour comprendre pourquoi tout cela fonctionne.
- Dans ce cours, un scalaire sera écrit normalement (x) tandis qu'un vecteur sera noté en gras ($\mathbf{x} = (x, y)$). Nous commençons la numérotation par 1 et non par 0.

CHAPITRE 1

Simulation Numérique

La dictature s'épanouit sur le terreau de l'ignorance.

---G. Orwell - *1984*

1.1 Quelques exemples d'EDP

1.1.1 Thermique

Prenons un domaine borné et connexe Ω, par exemple le carré unité, qui représente un studio. En supposant le milieu homogène, la température T au sein du studio vérifie *l'équation de Laplace* :

$$(-\Delta T) = 0, \qquad \text{dans } \Omega,$$

où l'opérateur $\Delta := \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}$ est le **Laplacien** ou **Opérateur de Laplace**.

Supposons maintenant que le studio comporte une source de chaleur, par exemple un radiateur. Nous le modélisons par une fonction q, continue, valant 0 partout sauf sur un petit domaine Ω_R. En notant la conductivité $K`(= constante)dumilieualors : math : `T$ vérifie **l'équation de Poisson** :

$$(-\Delta T) = \frac{q}{K}, \qquad \text{dans } \Omega.$$

Ce problème reste pour l'instant incomplet car il manque des conditions. Contrairement aux problèmes de Cauchy, il n'y a pas de condition initiale, car le régime est stationnaire, mais nous avons besoin de conditions sur le bord $\partial\Omega$ du domaine Ω. Nous parlons alors de *conditions aux limites* et par suite de *problème aux limites*.

Ajoutons une fenêtre à notre studio portée à la douce température de $T_0 = 10°C$ grâce à l'automne frisquet. Le reste des murs est supposé parfaitement isolants, autrement dit le flux à travers les parois est nul. Le flux d'une quantité sur une interface étant donné par

$$\text{flux } := \nabla T \cdot \mathbf{n} = \partial_{\mathbf{n}} T,$$

où **n** est le vecteur unitaire normale sortant au domaine. Le bord de notre appartement, noté $\Gamma := \partial\Omega$, est alors divisé en deux parties :

— Γ_D : la fenêtre sur laquelle la température est imposée : **condition de Dirichlet**.
— Γ_N : les murs supposés isolants sur lesquels le flux est imposé : **condition de Neumann**.
Notre problème s'écrit alors :

$$\left\{\begin{array}{rcll} -\Delta T & = & \dfrac{q}{K}, & \text{dans } \Omega, \\ T & = & T_0, & \text{sur } \Gamma_D, \\ \partial_{\mathbf{n}} T & = & 0, & \text{sur } \Gamma_N. \end{array}\right.$$

Nous verrons dans ce cours que ce problème admet une solution (ouf) et qui est, de plus, unique (re-ouf). Résoudre le problème analytiquement (*i.e.* « à la main ») peut s'avérérer délicat, notamment si la géométrie est complexe : c'est ici que la simulation numérique rentre en jeu et notamment les éléments finis.

1.1.2 Diffusion d'une onde Wifi dans un appartement (Projet 2017-2018)

Cet exemple est tiré du *projet proposé en 2017 - 2018*.

Modèle

La définition Wikipédia d'une onde est la suivante :

> Une onde est la propagation d'une perturbation produisant sur son passage une variation réversible des propriétés physiques locales du milieu. Elle se déplace avec une vitesse déterminée qui dépend des caractéristiques du milieu de propagation. Une onde transporte de l'énergie sans transporter de matière.

Mathématiquement, une onde $\mathscr{E}(\mathbf{x}, t)$ dépend du temps t et de l'espace $\mathbf{x}$, et vérifie l'équation des ondes :

$$\Delta \mathscr{E}(\mathbf{x}, t) = \frac{1}{c^2} \frac{\partial^2 \mathscr{E}}{\partial t^2}(\mathbf{x}, t),$$

où c est la célérité de l'onde dans le milieu (qui peut dépendre de la position $\mathbf{x}$!). Par exemple, dans le cas d'une onde électromagnétique et dans le vide, c est la célérité de la lumière, soit 299792458m.s^{-1}. La quantité $\mathbf{x}$ est un vecteur de dimension $d = 2$ ou $d = 3$ selon le problème considéré : dans notre cas $d = 2$.

Lors d'une excitation périodique, c'est-à-dire lorsque la pulsation ω (en rad.s^{-1}) de l'onde est fixée, l'onde s'écrit alors $\mathscr{E}(\mathbf{x}, t) = \Re\left(u(\mathbf{x}) e^{-\imath\omega t}\right)$ où $\imath = \sqrt{-1}$ et E est une onde *spatiale* satisfaisant l'équation de Helmholtz :

$$\Delta E + \frac{\omega^2}{c^2} E = f.$$

Cette équation s'obtient en remplaçant $\mathscr{E}(\mathbf{x}, t)$ par $E(\mathbf{x}) e^{-\imath\omega t}$ dans l'équation des ondes. Nous notons en général $k = \frac{\omega}{c}$ (en rad.m^{-1}) le nombre d'onde et $\lambda = \frac{2\pi}{k}$ (en m) la longueur d'onde, autrement dit, la distance entre deux amplitudes, de sorte que l'équation de Helmholtz s'écrit

$$\Delta E + k^2 E = f.$$

La source f est ici spatiale, dans le cas d'une source ponctuelle de centre $\mathbf{s}$ la source est alors un Dirac :

$$\Delta E + k^2 E = -\delta_{\mathbf{s}}.$$

Les ondes Wi-Fi qui suivent la norme IEEE 802.11g sont émises à une fréquence variant de 2.4GHz à 2.5GHz. L'appartement tout entier dans lequel est situé notre routeur est noté Ω. Les murs sont supposés être du même matériau : du placo-plâtre. Le domaine $\Omega = \Omega_a \cup \Omega_{\text{mur}}$ est décomposé en deux domaines, Ω_a pour l'air et Ω_{mur} pour les murs.

Une modélisation possible de ce problème est le système d'équations suivant :

En supposant que l'air a les mêmes propriétés électromagnétiques que le vide

$$\left\{\begin{array}{rcll} \Delta E(\mathbf{x}) + k^2 n(\mathbf{x})^2 E(\mathbf{x}) & = & -\delta_{\text{routeur}}(\mathbf{x}) & \text{dans } \Omega, \\ \partial_{\mathbf{n}} E(\mathbf{x}) - \imath k n(\mathbf{x}) E(\mathbf{x}) & = & 0 & \text{sur } \partial\Omega, \end{array}\right.$$

où nous avons :

— δ_{routeur} : position du routeur. Nous l'avons placé dans le salon.
— n : *fonction de contraste* qui prend en compte les caractéristiques électromagnétiques du mur et de l'air :

$$n(\mathbf{x}) = \begin{cases} 1 & \text{si } \mathbf{x} \in \Omega_a \quad (\textit{i.e. } \mathbf{x} \text{ est dans l'air}), \\ 2.4 & \text{si } \mathbf{x} \in \Omega_{\text{mur}} \quad (\textit{i.e. } \mathbf{x} \text{ est dans le mur}). \end{cases}$$

Notez que ces valeurs sont des valeurs physiques et ne sont pas une lubie mathématique.
— La dernière équation, $\partial_{\mathbf{n}} E - \imath k n E = 0$ est une **condition de Fourier-Robin** (ou *de Fourier* ou *de Robin* ou même *d'impédance*). Elle a pour but *d'absorber* (avec un succès mitigé) les ondes sortantes, mimant un mur « transparent » (sans réflexion d'ondes).

Résolution numérique

La résolution d'un tel problème dans un appartement deux pièces avec cuisine séparée (grand luxe Parisien) avec la méthode des éléments finis donne alors ce résultat :

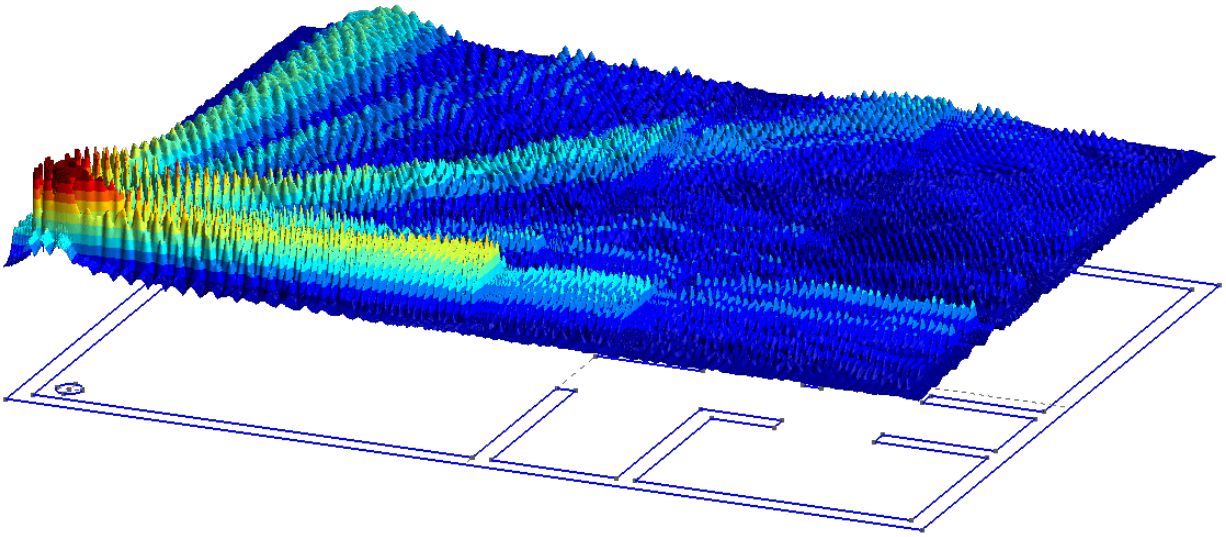

FIG. 1 – Propagation d'une onde Wi-Fi dans un appartement. Après avoir traversé 2 murs, l'onde Wi-Fi semble très amortie. Sous le résultat est affiché le plan de l'appartement et la position du routeur (petit disque à gauche)

Vous voulez tester ?

— Téléchargez le bundle Onelab. Il contient GMSH et GetDP (un solveur éléments finis)
— Téléchargez le code, soit directement soit via *Git* :

```
git clone https://github.com/Bertbk/wifi.git wifi
```

— Dans le dossier et dans un terminal, lancer

```
gmsh wifi.pro
```

— Vous pouvez modifier un peu la géométrie et la fréquence de l'onde, mise à 1GHZ. Attention, cette simulation est très gourmande : testez d'abord avec 1GHz avant de lancer la simulation pour 2.5GHz (au risque de faire crasher votre ordinateur) !

1.1.3 Objectifs du cours

Vous serez capable de résoudre ce genre de problème (et bien d'autres) et, ainsi, d'épater votre famille lors de ces interminables dîners.

1.2 Industrie

À venir…

1.2.1 Combustion

Cerfacs, logiciel AVBP

CHAPITRE 2

Formulations Faibles

> This is the problem of all great revelations : their significance so often exceeds the frame of our comprehension. We understand only after, always after. Not simply when it is too late, but precisely because it is too late.
>
> ---R. Scott Bakker - *The Darkness That Comes Before : Book 1 of the Prince of Nothing* (Titre français : *Autrefois, les Ténèbres*),

2.1 Espaces de Hilbert : rappels

Definition 2.1

Soit V un $\mathbb{R}-$ espace vectoriel, alors l'application $(\cdot,\cdot) : V \times V \to \mathbb{R}$ est un produit scalaire si et seulement si elle vérifie, pour tout $\mathbf{x}, \mathbf{y}, \mathbf{z} \in V$ et tout scalaire $\alpha \in \mathbb{R}$:

1. $(\mathbf{x}, \mathbf{y}) = (\mathbf{y}, \mathbf{x})$
2. $(\mathbf{x} + \mathbf{y}, \mathbf{z}) = (\mathbf{x}, \mathbf{z}) + (\mathbf{y}, \mathbf{z})$
3. $(\alpha\mathbf{x}, \mathbf{y}) = \alpha\,(\mathbf{x}, \mathbf{y})$
4. $(\mathbf{x}, \mathbf{x}) \in \mathbb{R}^+$
5. $(\mathbf{x}, \mathbf{x}) = 0 \Longrightarrow \mathbf{x} = 0$

Definition 2.2

Un $\mathbb{R}-$ espace vectoriel V est dit pré-Hilbertien si il est muni d'un produit scalaire.

Definition 2.3

Soit V un $\mathbb{R}-$ espace vectoriel, alors l'application $\|\cdot\| : V \to \mathbb{R}$ est une norme si et seulement si elle vérifie, pour tout $\mathbf{x}, \mathbf{y} \in V$ et tout scalaire $\alpha \in \mathbb{R}$:

1. Séparation : $\|\mathbf{x}\| = 0 \Longrightarrow x = 0$
2. Absolue homogénéité : $\|\alpha\mathbf{x}\| = |\alpha| \, \|\mathbf{x}\|$
3. Inégalité triangulaire : $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$

Remark 2.1

Un produit scalaire induit une norme sur un espace de Hilbert :

$$\|\mathbf{x}\| := \sqrt{(\mathbf{x}, \mathbf{x})}.$$

Nous rappelons l'inégalité de Cauchy Schwarz :

Proposition 2.1 (Inégalité de Cauchy Schwarz)

Pour tout **x** et **y** appartenant à un espace pré-Hilbertien V :

$$|(\mathbf{x}, \mathbf{y})| \leq \|\mathbf{x}\| \, \|\mathbf{y}\| \, .$$

Definition 2.4

Un espace pré-Hilbertien V est un espace de Hilbert si et seulement si il est complet pour la norme $\|\cdot\|$ induite par son produit scalaire.

Definition 2.5

Soit V un espace de Hilbert. L'application $f : V \times V \to \mathbb{R}$ est une forme bilinéaire sur V si et seulement si, pour tout $\mathbf{x}, \mathbf{y}, \mathbf{z}$ de V et α de $\mathbb{R}$:

1. $f(\mathbf{x}, \mathbf{y} + \alpha\mathbf{z}) = f(\mathbf{x}, \mathbf{y}) + \alpha f(\mathbf{x}, \mathbf{z})$
2. $f(\alpha\mathbf{x} + \mathbf{y}, \mathbf{z}) = \alpha f(\mathbf{x}, \mathbf{z}) + f(\mathbf{y}, \mathbf{z})$

Theorem 2.1 (Représentation de Riesz)

Soit V un espace de Hilbert de produit scalaire $(\cdot, \cdot)$ et de norme induite $\|\cdot\|$. Pour toute forme anti-linéaire continue ℓ, il existe un unique $w \in V$ tel que

$$\ell(v) = (w, v) \, , \quad \forall v \in V.$$

De plus, nous avons

$$\|w\| = \sup_{v \in V \setminus \{0\}} \frac{|\ell(v)|}{\|v\|}.$$

Remark 2.2

Ce théorème montre que la forme ℓ peut être **représentée** par un vecteur w qui est unique. Autrement dit, peu importe v, la quantité $\ell(v)$ peut se calculer par la seule connaissance du vecteur w et d'un « simple » produit scalaire.

2.2 Formulation Faible

2.2.1 Domaine Physique

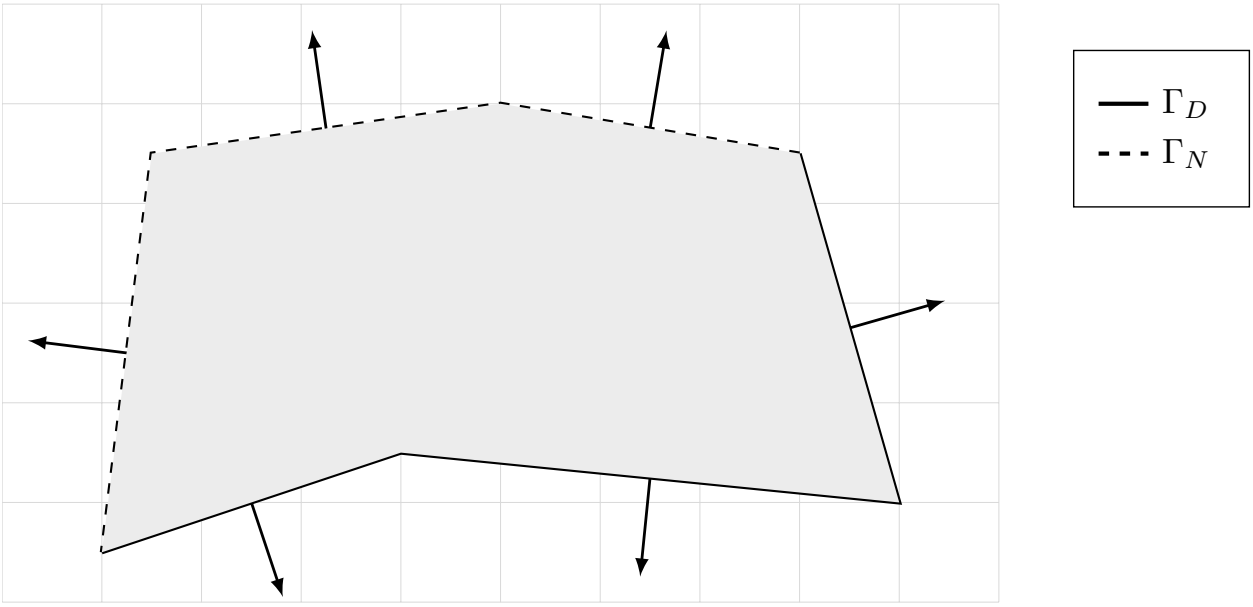

FIG. 1 – Exemple de domaine de calcul avec sa normal unitaire sortante

Dans ce cours, nous considérons un ouvert polygonal Ω de $\mathbb{R}^2$. Nous restons en dimension 2 pour plus de facilité mais l'extension à la dimension 3 est relativement directe et tous les résultats énoncés sont aussi valable en dimension 3. Sur chaque segment du bord $\Gamma := \partial\Omega$ du domaine, on définit le vecteur unitaire normale $\mathbf{n}$ sortant à Ω. Nous noterons que ce vecteur n'existe pas aux intersections entre les segments. Le domaine Ω est supposé ne pas comporter de fissure ni de point de rebroussement. Son bord est divisé en deux parties distinctes : Γ_D et Γ_N, potentiellement non connexe mais d'intersection vide : $\Gamma = \overline{\Gamma_D} \cup \overline{\Gamma_N}$ et $\Gamma_D \cap \Gamma_N = \emptyset$. Selon la partie du bord, une condition sera imposée à la solution :

- Sur Γ_D : *condition de Dirichlet*, c'est à dire que la valeur de la solution y est imposée (eg $u = 0$). En mécanique on dirait que le déplacement est imposée.
- Sur Γ_N : *condition de Neumann*, c'est à dire que le flux de la solution y est imposée (eg $\partial_{\mathbf{n}} u = 0$). En mécanique, on dirait que la force normale est imposée.

Remark

En général, on préfère travailler avec des ouverts *réguliers*, de classe au moins $\mathcal{C}^1$. Un tel ouvert présente l'avantage de pouvoir clairement définir le vecteur unitaire normale $\mathbf{n}$ sortante à Ω. Cependant, après maillage, on se retrouve avec… un polygone ! Alors plutôt que de travailler dans un domaine régulier pour après le casser en (petits) morceaux, nous préférons ici commencer directement avec un polygone et mettre l'accent sur les algorithmes et la mise en oeuvre de la méthode que les spécificités mathématiques.

Remark

Un point $\mathbf{x} \in \mathbb{R}^2$ est parfois noté $\mathbf{x} = (x, y)$ ou $\mathbf{x} = (x_1, x_2)$ selon les besoins. Nous commencerons les indiçages par 1 bien qu'en informatique cela commence souvent par 0.

2.2.2 EDP (problème fort)

Ce cours se concentre sur les équations aux dérivées partielles (EDP) elliptiques du second ordre, qui font appel à l'opérateur de Laplace [1] (ou *Laplacien*) :

$$\Delta := \frac{\partial^2}{\partial x} + \frac{\partial^2}{\partial y} = \frac{\partial^2}{\partial x_1} + \frac{\partial^2}{\partial x_2}.$$

Nous considérons le problème générique suivant, appelé aussi problème de réaction-diffusion :

$$\left\{ \begin{array}{rcll} -\Delta u + u & = & f & (\Omega), \\ \partial_{\mathbf{n}} u & = & 0 & (\Gamma), \end{array} \right. \tag{1}$$

où nous avons défini :

- le terme $\partial_{\mathbf{n}} u$ désigne la *dérivée normale* de u sur le bord, c'est à dire la dérivée de u dans la direction $\mathbf{n}$: $\partial_{\mathbf{n}} u = (\nabla u) \cdot \mathbf{n}$, avec $\nabla u = [\partial_{x_1} u, \partial_{x_2} u]^T$ son vecteur gradient. Vous aurez sans doute remarqué que, entre deux arêtes, le vecteur normal $\mathbf{n}$ n'est pas défini et donc la dérivée normale non plus. Ce « problème » n'en est pas vraiment un et pour l'instant mettez cela de côté, nous y reviendrons !
- $(-\Delta u)$: Terme de diffusion (notez le signe négatif)
- $\partial_{\mathbf{n}} u = 0$: **Condition de Neumann homogène**
- f : une fonction donnée définie sur Ω. Elle joue le rôle de *terme source*, c'est à dire d'apport (positif ou négatif), par exemple de chaleur ou de force surfacique.

Pour le moment, nous ne nous intéressons pas à la régularité de la solution ni même à l'existence et l'unicité de celle-ci : nous supposons que le problème (1) est bien posé. Une fois la méthode des éléments finis apréhendée, nous nous intéresserons à ces questions. Cela va à l'encontre de l'habitude en mathématiques où l'on démontre le caractère bien posé avant de s'y attaquer. Faites moi confiance et tout s'éclairera !

Remark

Pour l'instant, nous imposons une condition aux bords que nous imposons, de type Neumann homogène. Plus tard nous verrons d'autres types de conditions : Dirichlet, où la valeur de solution est imposée, et Fourier, un mélange entre Dirichlet et Neumann.

2.2.3 Théorème de Green

Pour une géométrie arbitraire, nous ne savons pas, en général, obtenir **la solution forte** (ou classique) du problème (1). La méthode des éléments finis se base sur l'approximation numérique de **la solution au sens faible** du problème (1). Nous verrons qu'une solution *faible* est, en fait et en général, *forte*. Commençons par réécrire le problème d'origine sous sa formulation faible ou formulation variationnelle.

Un théorème central dans l'analyse des EDP et qui permet d'obtenir ces formulations faibles est celui de Green [2]

$$\forall u, v, \qquad \int_\Omega (\Delta u)(\mathbf{x}) v(\mathbf{x}) \mathrm{d}\mathbf{x} = -\int_\Omega \nabla u(\mathbf{x}) \cdot \nabla v(\mathbf{x}) \mathrm{d}\mathbf{x} + \int_\Gamma (\partial_{\mathbf{n}} u)(\mathbf{x}) v(\mathbf{x}) \mathrm{d}s(\mathbf{x}).$$

Ce résultat est également valable en dimension 3 pour des domaines polygonaux (ouf). Le produit $\nabla u \cdot \nabla v$ est le produit scalaire euclidien standard. La quantité v est ici laissé non définie, c'est normal : supposez que c'est une fonction de même régularité que u. Pour compacter les équations, nous n'indiquerons plus les quantités d'intégrations :

$$\forall u, v, \qquad \int_\Omega (\Delta u) v = -\int_\Omega \nabla u \cdot \nabla v + \int_\Gamma (\partial_{\mathbf{n}} u) v.$$

1. Pierre-Simon Laplace (1749 – 1827).
2. George Green (1793 – 1841). Mathématicien britannique (quasi) auto-didacte.

Remark

Ce résultat est en quelque sorte une extension multi-dimensionnel de l'intégration par partie sur un segment $\Omega = [a, b]$. En effet, en dimension 1, l'opérateur Δ devient la dérivée seconde. La normale sortante au segment devient un scalaire valant -1 « à gauche » (en a) et 1 « à droite » (en b) et la dérivée normale devient $\partial_{\mathbf{n}} u = \pm u'$:

$$\int_a^b u'' v = -\int_a^b u'v' + u'(b)v(b) - u'(a)v(a) = -\int_a^b u'v' + \partial_{\mathbf{n}} u(b)v(b) + \partial_{\mathbf{n}} u(a)v(a).$$

2.2.4 Formulation faible

Le point de départ de notre analyse est la réécriture sous forme faible du problème (1). Pour cela, la méthode consiste à :

- **Multiplier** l'EDP par une **fonction test** v
- **Intéger** le tout sur Ω
- Appliquer le **Théorème de Green**
- Appliquer les **conditions aux bords**

$$\begin{aligned} -\Delta u + u = f &\implies -\int_\Omega \Delta u v + \int_\Omega uv = \int_\Omega fv \\ &\implies \int_\Omega \nabla u \cdot \nabla - \int_\Gamma \underbrace{(\partial_{\mathbf{n}} u)}_{=0} v + c\int_\Omega uv = \int_\Omega fv \end{aligned}$$

Nous obtenons alors :

$$-\Delta u + u = f \quad (\Omega) \implies \forall v, \quad \int_\Omega \nabla u \cdot \nabla v + \int_\Omega uv = \int_\Omega fv. \tag{2}$$

Ainsi, et toujours sans regarder la régularité de u (ni de v), nous avons que : si u est solution de l'EDP (1) alors u est aussi solution de la formulation faible :

$$\begin{cases} \text{Trouver } u \text{ tel que} \\ \forall v, \quad \displaystyle\int_\Omega \nabla u \cdot \nabla v + \int_\Omega uv = \int_\Omega fv. \end{cases}$$

À gauche du signe égal se trouve l'inconnue (u) et à droite la donnée (f), c'est une convention et plus tard cette équation s'écrira sous la forme d'un système linéaire $AU = B$ où le vecteur B correspondra au membre de droite de (2) et la matrice A à la partie de gauche.

Remark

Attention, sur le bord Γ, $\partial_{\mathbf{n}} u = 0$ n'implique pas $u = 0$!

Nous pouvons maintenant définir plus proprement la quantité v. Appelée *fonction test* elle n'a d'autre rôle que de « tester » la solution. L'idée de la formulation faible est de chercher une solution qui vérifie l'EDP, non pas point à point (au sens fort, donc) mais « en moyenne », via l'intégrale. En mécanique, v est appelé « travaux virtuels » (avec la méthode éponyme qui est, en fait, la formulation faible) : cette quantité est arbitraire et n'est utile que pour écrire le problème faible (2).

2.3 Théorème de Lax-Milgram

Nous pouvons maintenant énoncer le théorème de Lax-Milgram (**à connaître par cœur**).

Theorem 2.2 (de Lax-Milgram)

Soit V un espace de Hilbert de produit scalaire $(\cdot,\cdot)$ et de norme $\|\cdot\|$, et soit la formulation faible suivante

$$\begin{cases} \text{Trouver } u \in V \text{ tel que,} \\ \forall v \in V, \quad a(u,v) = \ell(v). \end{cases} \tag{3}$$

Sous réserve des quatre hypothèses suivantes :

1. ℓ est une forme linéaire continue sur V :
$$\exists C > 0 / \forall v \in V, \quad |\ell(v)| \leq C \|v\| .$$
2. $a(\cdot,\cdot)$ est une forme biquilinéaire sur $V \times V$.
3. $a(\cdot,\cdot)$ est continue :
$$\exists M > 0 / \forall (u,v) \in V \times V, \quad |a(u,v)| \leq M \|u\| \|v\| .$$
4. $a(\cdot,\cdot)$ est coercive (ou elliptique) :
$$\exists \alpha > 0 / \forall u \in V, \quad a(u,u) \geq \alpha \|u\|^2 .$$

Alors la formulation faible (3) admet une unique solution. De plus cette solution dépend continûment de la forme linéaire ℓ :

$$\|u\| \leq \frac{M}{\alpha} C$$

Proof. Comme il est question de forme linéaire, nous allons utiliser le Théorème de représentation de Riesz. En effet, pour tout w de V, l'application $v \to a(w,v)$ est anti-linéaire et continue de V dans $\mathbb{R}$. Il existe donc un unique élément de V, noté $A(w)$ (Théorème de Riesz), tel que

$$\forall v \in V, \quad a(w,v) = (A(w), v) .$$

Nous allons montrer que l'opérateur $A \colon V \to V$ est continue, inversible et d'inverse continu. L'opérateur A est clairement linéaire. En prenant $v = A(w)$ et en utilisant la continuité de $a(\cdot,\cdot)$, nous obtenons :

$$\|A(w)\|^2 = (A(w), A(w)) = a(w, A(w)) \leq M \|w\| \|A(w)\|.$$

Cette relation étant valable pour tout w, elle signifie que A est continue, puisque :

$$\forall w \in V, \qquad \|A(w)\| \leq M \|w\|.$$

Appliquons de nouveau le Théorème de représentation de Riesz au membre de droite, puisque ℓ est une forme anti-linéaire continue :

$$\exists! f \in V \text{ tel que } \|f\| = \|\ell\|_{V'} \text{ et } \forall v \in V, \ell(v) = (f,v) .$$

Comme $A(u) = f$ est équivalent à $\forall v \in V, (A(u), v) = (f, v)$, alors notre formulation faible (3) devient équivalent au problème linéaire :

$$\begin{cases} \text{Trouver } u \in V \text{ tel que} \\ A(u) = f. \end{cases}$$

La question est : A est-elle bijective ? Utilisons la coercivité de l'application $a(\cdot,\cdot)$:

$$\alpha\|w\|^2 \leq |(A(w),w)| \leq \|A(w)\|\|w\|,$$

ce qui implique que

$$\alpha\|w\| \leq \|A(w)\|. \tag{4}$$

Comme $\alpha > 0$, alors A est injective. En dimension finie et comme A est un endomorphisme, nous pourrions en déduire la surjectivité de A. Mais nous sommes malheureusement en dimension infinie, nous devons donc montrer que $\mathrm{Im}(A) = V$, pour cela nous montrons que $\mathrm{Im}(A)$ est fermé dans V et que son orthogonal (dans V) est réduit au singleton nul. Prenons une suite $(A(w_n))_n$ de $\mathrm{Im}(A)$ qui converge dans V. Nous avons, pour tout $n, p \in \mathbb{N}$ et grâce à (4),

$$\alpha\|w_n - w_p\| \leq \|A(w_n) - A(w_p)\|.$$

Quand n et p tendent vers l'infini, alors $\|w_n - w_p\| \to 0$. La suite $(w_n)_n$ est donc une suite de Cauchy dans V, qui est complet (Hilbert), elle est donc convergente et converge vers un élément w de V. Par continuité de A, la suite $(A(w_n))_n$ converge vers $A(w)$, élément de $\mathrm{Im}(A)$. Ce qui implique que $\mathrm{Im}(A)$ est fermé. Prenons maintenant $v \in \mathrm{Im}(A)^\perp$, par la coercivité de $a(\cdot,\cdot)$, nous avons

$$\alpha\,|v|^2 \leq |a(v,v)| = |(A(v),v)| = 0.$$

Autrement dit, $v = 0$ et donc $\mathrm{Im}(A)^\perp = \{0\}$ et nous avons

$$\mathrm{Im}(A) = \overline{\mathrm{Im}(A)} = \left(\mathrm{Im}(A)^\perp\right)^\perp = \{0\}^\perp = V.$$

L'application A est donc bijective. Son inverse A^{-1} existe, et, avec (4), nous obtenons sa continuité :

$$\forall w \in V, \qquad \left\|A^{-1}(w)\right\| \leq \frac{1}{\alpha}\|w\|.$$

Ceci prouve que u dépend continûment du membre de droite f (qui dépend de ℓ).

Remark 2.8

À quoi sert ce théorème ? Sous réserve de 4 hypothèses, nous aurons la garantie que la formulation faible obtenue précédemment admet une solution (ce qui est bien) et que cette solution est unique (encore mieux !). Il est donc d'une importance capitale.

Avant de pouvoir appliquer ce théorème proprement dit, nous devons connaître un peu mieux les espaces de Sobolev : Hilbert ? Norme ? Tant de questions.

2.4 Espaces de Sobolev

2.4.1 Espace $L^2(\Omega)$

Rappelons que l'espace $L^2(\Omega)$ est l'espace des fonctions de carré mesurable (au sens de Lebesgue). Muni du produit scalaire

$$(f,g)_{L^2(\Omega)} = \int_\Omega f(\mathbf{x})g(\mathbf{x})\mathrm{d}\mathbf{x},$$

l'espace $L^2(\Omega)$ est un espace de Hilbert, de norme induite :

$$\|f\|_{L^2(\Omega)} = \left(\int_\Omega |f(\mathbf{x})|^2 \, \mathrm{d}\mathbf{x} \right)^{\frac{1}{2}}.$$

Il est important de remarquer qu'une fonction de $L^2(\Omega)$ est définie **presque partout**. Autrement dit, deux fonctions f et g de $L^2(\Omega)$ peuvent être *égales* tout en ayant des valeurs différentes sur un sous-ensemble ω de Ω, de mesure nulle. Une fonction de $L^2(\Omega)$ désigne en réalité une *classe de fonctions.*

Introduisons l'espace $\mathscr{C}_c^\infty(\Omega)$ des fonctions $\mathscr{C}^\infty$ sur Ω à support compact dans Ω :

$$\mathscr{C}_c^\infty(\Omega) := \{ f \in \mathscr{C}^\infty(\Omega) \mid \text{ supp}(f) \text{ est compact dans } \Omega \}.$$

Remark 2.9

— Ces fonctions (et toutes leurs dérivées) s'annulent nécesairement sur le bord de Ω (qui est ouvert)
— Dans notre cas, $\Omega \in \mathbb{R}^2$ (ou même $\mathbb{R}^3$), ce qui impliqué qu'un compact de Ω est donc un fermé borné - Un exemple d'une telle fonction est la fonction « blob » comme illustré par l'article Wikipédia

---

Nous rappelons/admettons le théorème de densité suivant.

Theorem 2.3 (Densité dans $L^2(\Omega)$)

L'ensemble $\mathscr{C}_c^\infty(\Omega)$ est dense dans $L^2(\Omega)$.

Autrement dit, pour tout élément f de $L^2(\Omega)$, il existe une suite $(f_n)_n$ de fonctions de $\mathscr{C}_c^\infty(\Omega)$ qui converge vers f pour la norme de $L^2(\Omega)$. Ce théorème est extrêmement important : pour démontrer des propriétés de $L^2(\Omega)$, nous utiliserons des propriétés de $\mathscr{C}_c^\infty(\Omega)$ et passerons à la limite dans $L^2(\Omega)$.

Corollary 2.1

Soit f une fonction de $L^2(\Omega)$ telle que

$$\forall \phi \in \mathscr{C}_c^\infty(\Omega), \qquad \int_\Omega f(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x} = (f, \phi)_{L^2(\Omega)} = 0,$$

alors $f(\mathbf{x}) = 0$ presque partout dans Ω.

Proof. D'après le théorème **??**, il existe une suite $(f_n)_n$ de $\mathscr{C}_c^\infty(\Omega)$ qui converge vers f. Nous avons alors

$$0 = \lim_{n\to\infty} \int_\Omega f(\mathbf{x}) f_n(\mathbf{x}) \mathrm{d}\mathbf{x} = \int_\Omega |f(\mathbf{x})|^2 \, \mathrm{d}\mathbf{x} = \|f\|^2_{L^2(\Omega)},$$

d'où f est nulle « au sens de » $L^2(\Omega)$, c'est-à-dire que $f(\mathbf{x}) = 0$ presque partout.

L'espace $L^2(\Omega)$ est un « petit » espace de Hilbert qui contient $\mathscr{C}^1(\Omega)$. Nous nous rapprochons du but… Cependant les fonctions de $L^2(\Omega)$ ne sont pas dérivables ! Elles ne sont donc pas utilisables en pratique dans les formulations faibles. C'est tout l'objet de la section suivante : proposer une nouvelle forme de dérivation *plus faible*, c'est-à-dire ici, qui ne requiert pas de continuité.

2.4.2 Dérivée faible

Definition 2.6

Une fonction de $L^2(\Omega)$ est dérivable au sens faible par rapport à la direction x_i si et seulement si il existe un élément g_i de $L^2(\Omega)$ tel que

$$\forall \phi \in \mathscr{C}_c^\infty(\Omega), \quad \int_\Omega f(\mathbf{x})\partial_{x_i}\phi(\mathbf{x})\mathrm{d}\mathbf{x} = -\int_\Omega g_i(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x}.$$

Nous notons alors $g_i = \partial_{x_i} f = \partial_i f$, qui est unique en vertu du Corollaire 2.4.1.

Nous noterons maintenant $\partial_{x_i} f \in L^2(\Omega)$ ou $\partial_i f \in L^2(\Omega)$ pour signifier que f est dérivable au sens faible par rapport à x_i. De la même manière, nous pouvons définir le gradient faible :

Definition 2.7

Une fonction $f \in L^2(\Omega)$ admet un gradient faible, noté ∇f, si et seulement si f est dérivable au sens faible par rapport à toutes ses variables, et nous avons alors

$$\nabla f = \left(\partial_{x_1} f, \partial_{x_2} f, \ldots, \partial_{x_d} f\right)^T.$$

Proposition 2.2 (Unicité de la dérivée faible)

Si la dérivée faible dans la direction x_i d'une fonction $u \in L^2(\Omega)$ existe alors elle est unique.

Proof. Soit $u \in L^2(\Omega)$ et supposons que u admette deux dérivées faibles dans la direction x_i : f_i et g_i, toutes deux dans $L^2(\Omega)$. Nous avons alors, par définition, les deux relations suivantes :

$$\forall \phi \in \mathscr{C}_c^\infty(\Omega), -\int_\Omega u(\mathbf{x})\partial_i\phi(\mathbf{x}) = \int_\Omega f_i(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x} = \int_\Omega g_i(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x}.$$

Autrement dit, nous avons

$$\forall \phi \in \mathscr{C}_c^\infty(\Omega), \qquad \int_\Omega (f_i - g_i)(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x} = 0,$$

et le Corollaire 2.4.1 implique que $f_i = g_i$.

Le lien entre *dérivée faible* et *dérivée forte* (ou *classique*) est maintenant présenté :

Proposition 2.3

Soit $u \in \mathscr{C}^1(\overline{\Omega})$ tel que son gradient, au sens classique, ∇u soit dans $\mathscr{C}^0(\overline{\Omega})$, alors u admet un gradient au sens faible $\widetilde{\nabla} u$ et l'on a $\nabla u = \widetilde{\nabla} u$.

Proof. Il suffit de montrer ce résultat pour une direction uniquement, c'est-à-dire montrer que $\widetilde{\partial}_i u = \partial_i u$, si $\widetilde{\partial}_i$ est la dérivée partielle au sens faible. Par intégration par partie, nous avons :

$$\forall \phi \in \mathscr{C}_c^\infty(\Omega), \qquad \int_\Omega \partial_i u(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x} = -\int_\Omega u(\mathbf{x})\partial_i\phi(\mathbf{x})\mathrm{d}\mathbf{x} + \int_{\partial\Omega} u(\mathbf{x})\phi(\mathbf{x})n_i(\mathbf{x})\mathrm{d}s(\mathbf{x}),$$

où n_i est la $i^{\text{ème}}$ composante du vecteur normale $\mathbf{n}$. Comme ϕ est à support compact dans Ω, nous savons que ϕ s'annule sur le bord de Ω. Il vient donc

$$\forall \phi \in \mathscr{C}_c^\infty(\Omega), \qquad \int_\Omega \partial_i u(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x} = -\int_\Omega u(\mathbf{x})\partial_i\phi(\mathbf{x})\mathrm{d}\mathbf{x},$$

Autrement dit u admet une dérivée faible. Celle-ci étant unique, nous avons bien $\partial_i u = \widetilde{\partial}_i u$.

Remark 2.10

Dans la suite, puisque nous ne travaillerons qu'avec des dérivées partielles faibles, nous **omettrons le tilde**.

2.4.3 Espace de Sobolev $H^1(\Omega)$

Nous disposons maintenant des outils nécessaires pour introduire l'espace de Sobolev $H^1(\Omega)$ des fonctions de carré intégrable et dérivables au sens faible dans chaque direction ($d = 2, 3$ est la dimension) :

$$H^1(\Omega) = \left\{ u \in L^2(\Omega) \;\middle|\; \nabla u \in (L^2(\Omega))^d \right\}.$$

Nous munissons cet espace du produit scalaire suivant (pour u et v dans $H^1(\Omega)$)

$$(u, v)_{H^1(\Omega)} = \int_\Omega u(\mathbf{x})v(\mathbf{x})\ \mathrm{d}\mathbf{x} + \int_\Omega \nabla u(\mathbf{x}) \cdot \nabla v(\mathbf{x})\ \mathrm{d}\mathbf{x},$$

et de la norme induite, pour $u \in H^1(\Omega)$:

$$\|u\|_{H^1(\Omega)} = \left(\int_\Omega |u(\mathbf{x})|^2\ \mathrm{d}\mathbf{x} + \int_\Omega |\nabla u(\mathbf{x})|^2\ \mathrm{d}\mathbf{x} \right)^{\frac{1}{2}}.$$

Remark 2.11

Nous pouvons montrer que c'est effectivement un produit scalaire avec les arguments similaires à ceux utilisés pour montrer que la « même » application est un produit scalaire sur $\mathscr{C}^1(\Omega)$.

Remark 2.12

Pour u de $H^1(\Omega)$, nous avons clairement

$$\begin{aligned} \|u\|^2_{H^1(\Omega)} &= \|u\|^2_{L^2(\Omega)} + \sum_{i=1}^d \|\partial_i u\|^2_{L^2(\Omega)} \\ &= \|u\|^2_{L^2(\Omega)} + \|\nabla u\|^2_{(L^2(\Omega))^d}, \end{aligned}$$

et donc les inégalités suivantes :

1. $\|u\|_{H^1(\Omega)} \geq \|u\|_{L^2(\Omega)}$
2. $\|u\|_{H^1(\Omega)} \geq \|\nabla u\|_{(L^2(\Omega))^d} = \left(\sum_{i=1}^d \|\partial_i u\|^2_{L^2(\Omega)} \right)^{1/2}$
3. $\|u\|_{H^1(\Omega)} \geq \|\partial_i u\|_{L^2(\Omega)} \qquad \forall i = 1, 2, \ldots, d$

Nous montrons maintenant que $H^1(\Omega)$ muni de cette norme est complet.

Theorem 2.4 (Complétude de $H^1(\Omega)$)

L'espace $H^1(\Omega)$ est complet pour la norme $\|\cdot\|_{H^1(\Omega)}$.

Proof. Prenons une suite de Cauchy $(u_n)_n$ de $H^1(\Omega)$ et montrons qu'elle converge dans $H^1(\Omega)$. Par définition de la suite de Cauchy, nous avons

$$\forall \varepsilon > 0, \exists N > 0 \text{ tel que } \forall n > N, \forall p > N, \quad \|u_n - u_p\|_{H^1(\Omega)} \leq \varepsilon.$$

Par ailleurs, pour n, p de $\mathbb{N}$ l'inégalité suivante est vérifiée :

$$\|u_n - u_p\|_{L^2(\Omega)} \leq \|u_n - u_p\|_{H^1(\Omega)},$$

ce qui fait de la suite $(u_n)_n$ une suite de Cauchy dans $L^2(\Omega)$, puisque :

$$\forall \varepsilon > 0, \exists N > 0 \text{ tel que } \forall n > N, \forall p > N, \quad \|u_n - u_p\|_{L^2(\Omega)} \leq \|u_n - u_p\|_{H^1(\Omega)} \leq \varepsilon.$$

L'espace $L^2(\Omega)$ étant complet, la suite $(u_n)_n$ converge dans $L^2(\Omega)$ vers $u \in L^2(\Omega)$. Nous appliquons le même raisonnement aux dérivées partielles : pour $i = 1, \ldots, d$, nous avons aussi

$$\|\partial_i u_n - \partial_i u_p\|_{L^2(\Omega)} \leq \|u_n - u_p\|_{H^1(\Omega)}.$$

Ainsi, pour tout i, la suite $(\partial_i u_n)_n$ est aussi de Cauchy dans $L^2(\Omega)$ et converge donc vers un élément $f_i \in L^2(\Omega)$. Il nous faut donc montrer que u est dérivable (au sens faible) et que $f_i = \partial_i u$. Remarquons pour cela que, par définition,

$$\forall \phi \in \mathscr{C}_c^\infty(\Omega), \qquad \int_\Omega \partial_i u_n(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x} = -\int_\Omega u_n(\mathbf{x})\partial_i\phi(\mathbf{x})\mathrm{d}\mathbf{x}.$$

En passant à la limite dans $L^2(\Omega)$ dans cette expression, il vient que :

$$\forall \phi \in \mathscr{C}_c^\infty(\Omega), \qquad \int_\Omega f_i(\mathbf{x})\phi(\mathbf{x})\mathrm{d}\mathbf{x} = -\int_\Omega u(\mathbf{x})\partial_i\phi(\mathbf{x})\mathrm{d}\mathbf{x}.$$

Autrement dit, u est dérivable par rapport à toutes ses variables et $\partial_i u = f_i$, ce qui implique que u est bien dans $H^1(\Omega)$. Nous avons donc montré que la suite $(u_n)_n$ converge dans $L^2(\Omega)$ vers un élément u de $H^1(\Omega)$. Il nous reste à montrer que cette convergence est toujours valable pour la norme de $H^1(\Omega)$. Utilisons la remarque précédente pour décomposer la norme dans $H^1(\Omega)$:

$$\|u_n - u\|^2_{H^1(\Omega)} = \|u_n - u\|^2_{L^2(\Omega)} + \sum_{j=1}^{d} \|\partial_j u_n - \partial_j u\|^2_{L^2(\Omega)} \to 0 (n \to +\infty).$$

La suite de Cauchy $(u_n)_n$ est donc convergente dans $H^1(\Omega)$, ce dernier est donc complet.

Nous en déduisons le corollaire suivant :

Corollary 2.2

$H^1(\Omega)$ est un espace de Hilbert pour le produit scalaire $(\cdot, \cdot)_{H^1(\Omega)}$.

Nous avons également le résultat de densité suivant

Proposition 2.4 (Admis)

L'espace $\mathscr{C}_c^\infty(\Omega)$ est dense dans $H^1(\Omega)$ pour la norme $\|\cdot\|_{H^1(\Omega)}$.

En particulier, l'espace $\mathscr{C}^1(\Omega)$, qui contient $\mathscr{C}_c^\infty(\Omega)$, est dense dans $H^1(\Omega)$ pour la norme $\|\cdot\|_{H^1(\Omega)}$. Ce résultat nous dit que $H^1(\Omega)$ est le « plus petit » espace complet contenant $\mathscr{C}^1(\Omega)$: c'est ce que nous cherchions !

2.5 Application au problème modèle

2.5.1 Formulation faible

Considérons un ouvert polygonal connexe Ω et le problème suivant

$$\left\{\begin{array}{rcll} -\Delta u + cu & = & f & (\Omega) \\ \partial_{\mathbf{n}} u & = & 0 & (\Gamma_N = \Gamma) \end{array}\right. \tag{5}$$

Après mutiplication par des fonctions tests et intégrations par partie, nous obtenons la formulation faible de ce problème.

$$\left\{\begin{array}{l} \text{Trouver } u \in H^1(\Omega) \text{ tel que} \\ \forall v \in H^1(\Omega), a(u,v) = \ell(v) \end{array}\right. \tag{6}$$

avec $a(\cdot,\cdot)\colon H^1(\Omega)\times H^1(\Omega)\to\mathbb{R}$ et $\ell(\cdot)\colon H^1(\Omega)\to\mathbb{R}$ définies pas

$$\left\{\begin{array}{l} a(u,v) = \displaystyle\int_\Omega \nabla u\cdot\nabla v + c\int_\Omega uv \\ \ell(v) = \displaystyle\int_\Omega fv \end{array}\right.$$

2.5.2 Existence et unicité

Tentons d'appliquer le théorème de Lax-Milgram à cette formulation faible

1. $H^1(\Omega)$ est un espace de Hilbert
2. $\ell(\cdot)$ est clairement linéaire (du fait de l'intégrale)
3. $a(\cdot,\cdot)$ est bilinéaire, pour la même raison
4. Continuité de $\ell(\cdot)$: prenons une fonction $v\in H^1(\Omega)$:

$$\begin{aligned} |\ell(v)| &= \underbrace{\left|\int_\Omega fv\right|}_{(f,v)_{L^2(\Omega)}} \\ &\leq \|f\|_{L^2(\Omega)}\|v\|_{L^2(\Omega)} && \text{Cauchy-Schwarz} \\ &\leq \underbrace{\|f\|_{L^2(\Omega)}}_{\text{Constant}}\|v\|_{H^1(\Omega)} && \text{inégalité des normes} \end{aligned}$$

5. Continuité de $a(\cdot,\cdot)$: prenons deux fonctions u et v de $H^1(\Omega)$:

$$
\begin{aligned}
|a(u,v)| &= \left| \int_\Omega \nabla u \cdot \nabla v + c \int_\Omega uv \right| \\
&\leq \underbrace{\left| \int_\Omega \nabla u \cdot \nabla v \right|}_{(\nabla u, \nabla v)_{L^2(\Omega)^d}} + |c| \underbrace{\left| \int_\Omega uv \right|}_{(u,v)_{L^2(\Omega)}} && \text{inégalité classique} \\
&\leq \|\nabla u\|_{(L^2(\Omega))^d} \|\nabla v\|_{(L^2(\Omega))^d} + |c| \|u\|_{L^2(\Omega)} \|v\|_{L^2(\Omega)} && \text{inégalité triangulaire dans} L^2(\Omega) \\
&\leq \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)} + |c| \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)} && \text{inégalité des normes} \\
&\leq (1+c) \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)}
\end{aligned}
$$

6. Coercivité de $a(\cdot,\cdot)$: prenons une fonction $u \in H^1(\Omega)$:

$$
\begin{aligned}
a(u,u) &= \int_\Omega \nabla u \cdot \nabla u + c \int_\Omega uu = \int_\Omega \|\nabla u\|^2 + c \int_\Omega |u|^2 \\
&\geq \min(1,c) \left(\int_\Omega \|\nabla u\|^2 + \int_\Omega |u|^2 \right) \\
&\geq \min(1,c) \|u\|^2_{H^1(\Omega)}
\end{aligned}
$$

Toutes les conditions sont réunies : le problème (6) admet une unique solution d'après le théorème de Lax-Milgram.

Remark 2.13

Dans la démonstration de la continuité de ℓ, n'écrivez pas $\|f\|_{L^2(\Omega)} \leq \|f\|_{H^1(\Omega)}$ car, d'une part nous n'en avons pas besoin, d'autre part, nous ne savons pas si $f \in H^1(\Omega)$!

2.5.3 Conclusion

Schématiquement, nous avons :

- Si u est solution de (5) alors u est solution de (6)
- Le problème (6) admet une unique solution qui appartient (au moins) à $H^1(\Omega)$

Remark 2.14

Pourquoi travailler dans $H^1(\Omega)$ et non dans $\mathscr{C}^1(\overline{\Omega})$? La question est légitime, d'autant que $\|\cdot\|_{H^1(\Omega)}$ est une norme de $\mathscr{C}^1(\overline{\Omega})$! Mais… $\mathscr{C}^1(\overline{\Omega})$ n'est pas complet pour cette norme et n'est donc pas un espace de Hilbert si on lui adjoint cette norme : le théorème de Lax-Milgram ne pourra alors pas s'y appliquer. Il existe des normes qui complètent $\mathscr{C}^1(\overline{\Omega})$, mais les hypothèses du théorème de Lax-Milgram sont elles toujours validées avec ces normes ?

CHAPITRE 3

Éléments Finis Triangulaires

And the word was death

And the word was without light

The new beatitude

Good luck, you're on your own !

---A Perfect Circle - *The Doomed*

3.1 Méthode de Galerkin

3.1.1 Contexte

Dans ce chapitre nous considérons un espace de Hilbert V muni du produit scalaire $(\cdot,\cdot)_V$ et de sa norme associée $\|\cdot\|_V$. Nous considérons la formulation variationnelle suivante

$$\left\{ \begin{array}{l} \text{Trouver } u \in V \text{ tel que} \\ \forall v \in V, \quad a(u,v) = \ell(v). \end{array} \right. \tag{1}$$

Les formes continues $a(\cdot,\cdot)$ et $\ell(\cdot)$ sont respectivement bilinéaire et linéaire, et $a(\cdot,\cdot)$ est de plus coercive. De cette manière, le Théorème de Lax-Milgram s'applique et le problème (1) admet une unique solution.

Nous noterons $(\cdot,\cdot)_V$ et $\|\cdot\|_V$ respectivement le produit scalaire et la norme sur V.

3.1.2 Dimension finie

Obtenir une solution de (1) est compliqué car V est (a priori) de dimension infinie. La méthode de Galerkin consiste à « approcher » l'espace fonctionnel V par un espace $V_h \subset V$, de **dimension finie**, mais toujours de Hilbert, et ce pour le même produit scalaire ! La formulation faible (1) est alors résolue dans V_h uniquement, avec pour solution u_h :

$$\begin{cases} \text{Trouver } u_h \in V_h \text{ tel que} \\ \forall v_h \in V_h, \quad a(u_h, v_h) = \ell(v_h). \end{cases} \tag{2}$$

On espère alors que cette solution approchée u_h soit une bonne estimation de la solution exacte u, c'est-à-dire que

$$\lim_{h \to 0} \|u_h - u\|_V = 0.$$

Remarquons tout d'abord que la formulation faible (2) admet une unique solution.

Lemma 3.1

Le problème « approché » (2) admet une unique solution.

Proof. L'espace $V_h \subset V$ est un sous-espace de Hilbert de V, nous pouvons donc appliquer le Théorème de Lax-Milgram, dont les hypothèses sur $a(\cdot, \cdot)$ et $\ell(\cdot)$ sont toujours vérifiées sur V_h.

Travailler dans un espace de dimension finie présente un très grand avantage : on peut en extraire une **base de taille finie** et ramener le calcul de u_h à la **résolution d'un système linéaire**, pour lequel les outils (numériques) ne manquent pas. Citons par exemple les bibliothèques suivantes :

- MUMPS : solveur open-source direct parallèle
- Pardiso : solveur direct parallèle privatif d'Intel
- PETSc : Bibliothèque contenant entres autres de nombreux solveurs directs (dont MUMPS) ou itératifs (GMRES, …)

Lemma 3.2

Soit V un espace de Hilbert et V_h un sous espace de dimension finie. Soit $a(\cdot, \cdot)$ une forme bilinéaire continue et coercive sur V, $\ell(\cdot)$ une forme linéaire continue sur V. Le problème approché (2) admet une unique solution. De plus, cette solution s'obtient par la résolution d'un système linéaire de matrice définie positive.

Proof. Le problème (2) admet toujours une unique solution d'après le Théorème de Lax-Milgram. Comme V_h est de dimension finie, notée N_h, nous pouvons en extraire une base $(\varphi_1, \varphi_2, \ldots, \varphi_{N_h})$ et écrire

$$u_h = \sum_{I=1}^{N_h} u_I \varphi_I.$$

La formulation faible peut alors se réécrire sur les fonctions de cette base uniquement :

$$\forall I, \qquad \sum_{J=1}^{N_h} a(\varphi_J, \varphi_I) u_J = \ell(\varphi_I),$$

ou encore

$$A_h U_h = B_h,$$

avec $A_h = (a(\varphi_J, \varphi_I))_{1\leq I,J\leq N_h}$, $U_h = (u_I)_{1\leq J\leq N_h}$ et $B_h = (\ell(\varphi_I))_{1\leq I\leq N_h}$. Montrons maintenant que la matrice A_h est définie positive :

$$\begin{aligned}\forall W_h \in \mathbb{R}^{N_h}, W_h = (w_I)_{1\leq I\leq N_h},\\ (W_h, A_h W_h) = W_h^T A_h W_h &= \sum_{I=1}^{N_h}\sum_{J=1}^{N_h} w_j a(\varphi_I, \varphi_J) w_I \\ &= \sum_{I=1}^{N_h}\sum_{J=1}^{N_h} a(w_I\varphi_I, w_j\varphi_J) \\ &= a\left(\sum_{I=1}^{N_h} w_I\varphi_I, \sum_{J=1}^{N_h} w_j\varphi_J\right)\end{aligned}$$

L'indice J étant muet, nous pouvons changer son intitulé :

$$(W_h, A_h W_h) = a\left(\sum_{I=1}^{N_h} w_I\varphi_I, \sum_{I=1}^{N_h} w_I\varphi_I\right)$$

Nous utilisons maintenant la coercivité de $a(\cdot,\cdot)$:

$$(W_h, A_h W_h) \geq \alpha \left\|\sum_{I=1}^{N_h} w_I\varphi_I\right\|_V^2.$$

Comme $\alpha > 0$, alors le terme $(W_h, A_h W_h)$ est nul si et seulement si $\left\|\sum_{I=1}^{N_h} w_I\varphi_I\right\|_V$ est nulle et donc si et seulement si $\sum_{I=1}^{N_h} w_I\varphi_I$ est la fonction nulle. Comme la famille $(\varphi_I)_{1\leq I\leq N_h}$ forme une base de V_h, cela revient à dire que $w_I = 0$ pour tout I et donc que W_h est le vecteur nul. Nous avons donc montré que

$$\forall W_h \in \mathbb{R}^{N_h} \setminus \{0\}, \qquad (W_h, A_h W_h) > 0.$$

Remark 3.1

Quelques remarques :

- La matrice A_h *discrétise* l'opérateur $a(\cdot,\cdot)$ au sens où elle est de taille finie.
- La **coercivité** d'une forme $a(\cdot,\cdot)$ est, en quelque sorte, l'équivalent de la **définie positivité** de sa matrice. La coercivité s'applique au domaine « continu » (les *fonctions* ou *opérateurs*) tandis que la définie positivité est un terme appliqué au domaine « algébrique » (les *matrices* (infinies ou non)).
- L'hypothèse de Lax-Milgram sur la **coercivité** de $a(\cdot,\cdot)$ est une **hypothèse forte** puisque la matrice A_h discrétisant $a(\cdot,\cdot)$ doit être **définie positive** !

Remark 3.2

La méthode des différences finies discrétise l'opérateur différentiel (Δ) tandis que les éléments finis (issue de la méthode de Galerkin) approche l'espace fonctionnel. C'est une différence majeure !

3.2 Espace $\mathbb{P}^1$-Lagrange

La méthode des éléments finis est basée sur la méthode de Galerkin, ou d'approximation interne. L'idée est d'approcher l'espace fonctionnel $H^1(\Omega)$ par un espace de dimension finie : l'espace éléments finis. Nous nous intéressons à un tel premier espace : $\mathbb{P}^1-$ Lagrange ou plus simplement $\mathbb{P}^1$, composés des fonctions linéaires par triangles.

3.2.1 Maillage triangulaire (ou triangulation)

Nous découpons maintenant le domaine en triangles pour obtenir un maillage triangulaire (ou triangulation) conforme de Ω. Un maillage est conforme s'il suit les quelques règles simples suivantes. Une illustration est proposée sur la figure 1.

— L'union des N_t triangles doit couvrir Ω sans le dépasser : $\Omega = \bigcup_{p=1}^{N_t} K_p$.
— L'intersection de deux triangles est soit vide, soit une arête commune complète à chacun des deux triangles, soit un sommet de chacun des deux triangles.
— Une arête d'un triangle est soit une arête (complète) d'un autre triangle, soit une partie de Γ, auquel cas ce segment est complètement inclus soit dans Γ_D soit dans Γ_N (il n'y a pas d'arête appartenant à la fois à Γ_D et à Γ_N).

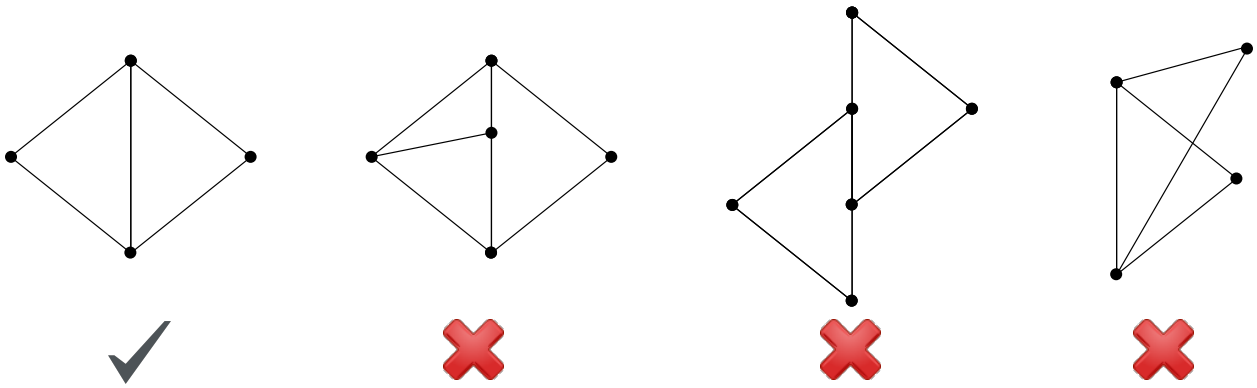

FIG. 1 – Différents maillages, conforme ou non.

Une telle triangulation sera noté $\mathscr{T}_h = \{K_p, p = 1, \dots, N_t\}$, l'indice h faisant référence à la **finesse de maillage**, que l'on définit par le grand diamètre des triangles :

$$h := \max_{K \in \mathscr{T}_h} (\text{diam}(K)) = \max_{p=1,\dots,N_t} (\text{diam}(K_p)).$$

Le diamètre d'un triangle est la distance maximale entre deux points du triangle. Nous notons de plus $\mathscr{S}_h$ et $\mathscr{A}_h$ les ensembles respectivement des sommets et des arêtes de $\mathscr{T}_h$. Pour un triangle arbitraire K, nous noterons $[\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2]$ ses sommets ordonnés. De même, pour un triangle K_p du maillage, ses sommets ordonnées seront notés $[\mathbf{s}_0^p, \mathbf{s}_1^p, \mathbf{s}_2^p]$.

Remark 3.3

Il existe aujourd'hui des mailleurs automatiques open-source, un des plus connu et que nous utiliserons est GMSH (un tutoriel est fourni par moi même). Le maillage automatique reste un métier à part entière tant la complexité est importante notamment en 3D et avec des géométries complexes, non forcément polygonales. D'autres parts, de nombreuses questions sont encore ouvertes aujourd'hui dans ce domaine comme la construction automatique d'un maillage composé de quadrangles. Nous n'entrerons pas dans les détails dans ce sujet, nous serons de simple « utilisateurs et utilisatrices ».

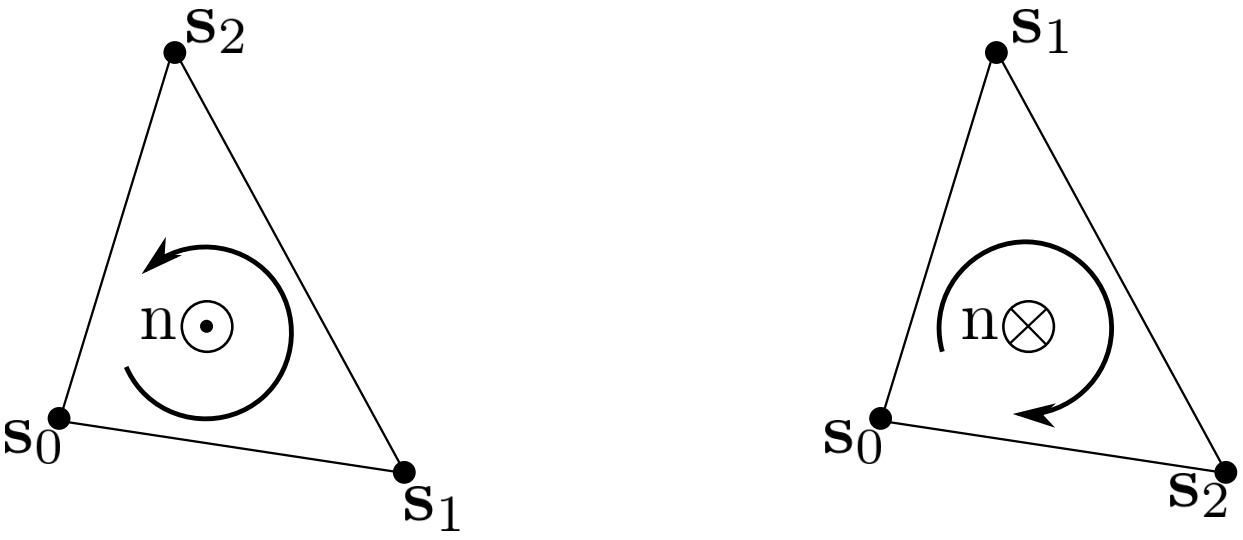

FIG. 2 – Deux orientations possibles pour un triangle. Dans les maillages considérés, tous les triangles ont la même orientation.

3.2.2 Fonction linéaire sur un triangle

Cas du segment (1D). Regardons tout d'abord le cas $1D$ d'un segment $[\alpha, \beta]$ et d'une fonction p linéaire sur ce segment : $p(x) = ax + b$. Les coefficients a et b caractérisent la fonction p et sont, de plus, définis de manière unique dès lors que l'on connait la valeur de p en α et en β (2 équations à 2 inconnues, linéairement indépendantes). Cette propriété reste naturellement vraie pour un segment $[\vec{\alpha}, \vec{\beta}]$ « plongé » en dimension 2. Un point $\mathbf{x}$ de ce segment est décrit par ses coordonées curvilignes : $\mathbf{x}(s) = (1-s)\vec{\alpha} + s\vec{\beta}$, pour $s \in [0,1]$, et un polynôme p de degré 1 sur $[\vec{\alpha}, \vec{\beta}]$ s'écrit alors $p(\mathbf{x}(s)) = (1-s)p(\vec{\alpha}) + sp(\vec{\beta})$ pour $s \in [0,1]$. On voit clairement qu'un polynôme de degré 1 sur un segment est défini de manière unique par ses valeurs aux extrémités [1].

Cas du triangle (2D). Revenons maintenant dans un triangle K non plat et notons $\mathbb{P}^1$ l'espace des polynômes réels de degré 1 sur $\mathbb{R}^2$, de dimension 3 :

$$\mathbb{P}^1(\mathbb{R}) = \left\{ p \colon \mathbb{R}^2 \to \mathbb{R} \;\middle|\; \exists! a, b, c \in \mathbb{R} \text{ tels que } \forall (x,y) \in \mathbb{R}^2, p(x,y) = a + bx + cy \right\}$$

L'espace $\mathbb{P}^1(K)$ des fonctions linéaires (ou des polynômes de degré 1) sur K est lui aussi de dimension 3 (car K n'est pas plat) :

$$\mathbb{P}^1(K) = \{ p \colon K \to \mathbb{R} \mid \exists! a, b, c \in \mathbb{R} \text{ tels que } \forall (x,y) \in K, p(x,y) = a + bx + cy \}$$

Une fonction p de $\mathbb{P}^1(K)$ est définie de manière unique par ses 3 coefficients a, b, c. Inversement, ces trois coefficients sont calculables dès lors que l'on connait la valeur de p sur trois points non alignés, comme les 3 sommets du triangle (voir la proposition suivant). Une fonction $p \in \mathbb{P}^1(K)$ est donc définie de manière unique soit par la connaissance de ses trois coefficients, soit par la connaissance de sa valeur sur les trois sommets du triangle.

Proposition 3.1

Soit K un triangle non dégénéré de $\mathbb{R}^2$ de sommets $\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3$. Alors, pour tout jeu de données $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{R}$, il existe un unique polynôme de $p \in \mathbb{P}^1(K)$ tels que $p(\mathbf{s}_i) = \alpha_i$ pour $i = 1, 2, 3$.

---

1. Au lycée on disait « entre deux points ne passe qu'une et une seule droite ».

Proof. En notant $\mathbf{s}_i = (x_i, y_i)$ et $p(x, y) = ax + by + c$ avec $a, b, c \in \mathbb{R}$, alors le problème revient à résoudre le système linéaire

$$\left\{ \begin{array}{rcl} ax_1 + by_1 + c & = & \alpha_1 \\ ax_2 + by_2 + c & = & \alpha_2 \\ ax_3 + by_3 + c & = & \alpha_3 \end{array} \right. \iff \left(\begin{array}{ccc} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{array} \right) \left(\begin{array}{c} a \\ b \\ c \end{array} \right) = \left(\begin{array}{c} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{array} \right)$$

Le déterminant d'un tel système n'est autre que deux fois l'aire du triangle K qui n'est pas dégénéré :

$$\Delta = \left| \begin{array}{ccc} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{array} \right| = 2\text{Aire}(K) \neq 0$$

Le système est donc bien inversible et admet une unique solution (a, b, c).

Remark 3.4

Soit une fonction $v \in \mathbb{P}^1(K)$, linéaire sur le triangle K. Sa restriction $v|_\sigma$ à une arête σ de K est elle même une fonction linéaire sur σ. Elle est donc complètement caractérisée par sa valeur aux sommets de l'arête, qui sont aussi des sommets de K.

3.2.3 Fonctions linéaires par éléments (= par triangles)

Nous pouvons maintenant introduire l'espace fonctionnel $\mathbb{P}^1-$ Lagrange, souvent abrégé $\mathbb{P}^1$ et noté dans ce cours V_h, contient les fonctions **continues** sur $\overline{\Omega}$ (le fermé de Ω) et **linéaires sur chaque triangle** :

$$V_h := \left\{ v_h \in \mathscr{C}^0(\overline{\Omega}) \;\middle|\; \forall K \in \mathscr{T}_h, v_h|_K \in \mathbb{P}^1(K) \right\}.$$

Caractérisons maintenant les fonctions de cet espace. Le premier résultat montre que deux fonctions de V_h sont égales si et seulement si elles coïncident sur tous les sommets de la triangulation $\mathscr{T}_h$.

Lemma 3.3

Si $u_h, v_h \in V_h$ vérifient $u_h(\mathbf{s}) = v_h(\mathbf{s})$ pour tout sommet $\mathbf{s}$ de $\mathscr{T}_h$, alors $u_h = v_h$ sur Ω.

Proof. En se plaçant sur le triangle $K = (\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3)$ de $\mathscr{T}_h$, nous avons $u_h(\mathbf{s}_i) = v_h(\mathbf{s}_i)$ pour $i = 1, 2, 3$. La proposition 3.2.2 implique que $u_h|_K = v_h|_K$. Le triangle K étant arbitraire, cette relation vaut sur tous les éléments de la triangulation. Le même raisonnement peut être effectué sur chaque arête pour obtenir que $u_h - v_h$ est nulle sur Ω tout entier.

Proposition 3.2

Pour tout jeu de données réelles $(\alpha_i)_{i=1,\ldots,N_s}$, il existe une unique fonction $v_h \in V_h$ vérifiant $v_h(\mathbf{s}_I) = \alpha_i$ pour tout $i = 1, \ldots, N_s$.

Proof. L'unicité est démontrée par le lemme 3.2.3, il manque donc l'existence. Prenons un triangle $K_p = (\mathbf{s}_1^p, \mathbf{s}_2^p, \mathbf{s}_3^p)$ de $\mathscr{T}_h$ et le jeu de valeurs associé $(\alpha_1^p, \alpha_2^p, \alpha_3^p) \in \mathbb{R}$. La proposition 3.2.2 montre qu'il existe un unique polynôme p_{K_p} de

$\mathbb{P}^1(K_p)$ tel que $p_{K_p}(\mathbf{s}_i^p) = \alpha_i^p$ pour $i = 1, 2, 3$. Nous pouvons répéter cette opération pour tous les triangles K_p et nous introduisons u_h tel que

$$\forall p = 1, \dots, N_t, \quad u_h|_{K_p} = p_{K_p}.$$

La fonction v_h est affine sur chaque triangle, il nous faut montrer que $u_h \in \mathscr{C}^0(\overline{\Omega})$ pour conclure sur son appartenance à V_h. Comme v_h est continue en chaque sommet $\mathbf{s}$, il reste à montrer la continuité sur les arêtes.

Prenons 2 triangles K_p et K_q de $\mathscr{T}_h$ ayant une arête Σ en commun. Quitte à renuméroter, notons $\mathbf{s}_1 = (x_1, y_1)$ et $\mathbf{s}_2 = (x_2, y_2)$ les deux sommets de l'arête Σ et notons

$$\sigma(t) = \mathbf{s}_1 + t(\mathbf{s}_2 - \mathbf{s}_1) = (x_1 + t(x_2 - x_1), y_1 + t(y_2 - y_1))$$

une paramétrisation de Σ. Si $p_{K_p}(x, y) = ax + by + c$, nous avons alors, pour tout $t \in [0, 1]$:

$$\begin{aligned}
p_{K_p}(\sigma(t)) &= a(x_1 + t(x_2 - x_1)) + b(y_1 + t(y_2 - y_1)) + c \\
&= a(x_1 + t(x_2 - x_1)) + b(y_1 + t(y_2 - y_1)) + c + t(c - c) \\
&= [ax_1 + by_1 + c] + t([ax_2 + by_2 + c] + [ax_0 + by_0 + c]) \\
&= p_{K_p}(\mathbf{s}_1) + t(p_{K_p}(\mathbf{s}_2) - p_{K_p}(\mathbf{s}_1)) \\
&= p_{K_q}(\mathbf{s}_1) + t(p_{K_q}(\mathbf{s}_2) - p_{K_q}(\mathbf{s}_1)) \\
&= p_{K_q}(\sigma(t)).
\end{aligned}$$

Autrement dit, les deux polynômes p_{K_p} et p_{K_q} sont égaux sur l'arête Σ. La fonction v_h est donc continue sur toutes les arêtes de $\mathscr{T}_h$ en plus de l'être sur tous les triangles et tous les sommets : v_h est donc bien **continue** sur tout $\overline{\Omega}$.

3.2.4 Base de V_h : les fonctions de forme

Au vue de ce qui précède, deux fonctions de V_h sont identiques si et seulement si elles possèdent la même valeur sur chaque sommet de $\mathscr{T}_h$. En notant $N_s = \text{card}(\mathscr{S}_h)$ le nombre de sommets du maillage, introduisons la famille des **fonctions de forme** $(\varphi_I)_{1 \leq I \leq N_s}$ de V_h, qui sont **nulles sur chaque sommet sauf un** :

$$\forall I, J = 1, ..., N_s, \quad \varphi_I(\mathbf{s}_J) = \delta_{I,J} = \begin{cases} 1 & \text{si } I = J \\ 0 & \text{sinon.} \end{cases}$$

Ces fonctions sont la généralisation en 2D des *fonctions chapeau* unidimensionnelles (elles ressemblent d'ailleurs encore plus à un « chapeau » !).

Proposition 3.3

La famille $(\varphi_I)_{1 \leq I \leq N_s}$ est une base de V_h, qui est alors de dimension N_s, le nombre de sommets de la triangulation $\mathscr{T}_h$.

Proof. Montrons que la famille des fonctions de forme est une base de V_h. Commençon par le caractère libre de cette famille en prenant N_s données $(\alpha_i)_{1 \leq i \leq N_s}$,

$$\begin{aligned}
\sum_{I=1}^{N_s} \alpha_I \varphi_I = 0 &\implies \forall j = 1, \dots, N_s, \quad \sum_{I=1}^{N_s} \alpha_I \varphi_I(\mathbf{s}_j) = 0 \\
&\implies \forall J = 1, \dots, N_s, \quad \alpha_J \times 1 + \sum_{I=1, I \neq J}^{N_s} (\alpha_J \times 0) = 0 \\
&\implies \forall J = 1, \dots, N_s, \quad \alpha_J = 0
\end{aligned}$$

La famille de fonctions $(\varphi_I)_{1\leq I\leq N_s}$ est libre. Pour montrer qu'elle est génératrice, prenons une fonction $u_h \in V_h$ et plaçons nous sur le triangle $K = (\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3)$ (quitte à renuméroter). Le polynôme $\left(\sum_{I=1}^{3} u_h(\mathbf{s}_I)\varphi_I\right)\Big|_K$ coïncide avec le polynôme $u_h|_K$ sur les sommets du triangle K. Les deux étant de degré 1, nous avons alors l'égalité de ces polynômes sur tout le triangle :

$$u_h|_K = \left(\sum_{I=1}^{3} u_h(\mathbf{s}_I)\varphi_I\right)\Big|_K .$$

Cette relation étant valable sur un triangle arbitraire, elle est vraie sur Ω. La famille de fonctions $(\varphi_I)_I$ est donc une base de V_h.

Proposition 3.4 (Admis pour le moment)

L'espace V_h est inclus dans $H^1(\Omega)$.

Lemma 3.4

Le support d'une fonction de forme φ_I est l'union des triangles ayant pour sommet $\mathbf{s}_I$:

$$\text{supp}(\varphi_I) = \{K \in \mathscr{T}_h \mid \mathbf{s}_I \text{ est un sommet de } K\}.$$

Autrement dit, en dehors de ces triangles, la fonction φ_I est nulle.

Proof. Prenons une fonction de forme φ_I associée au sommet $\mathbf{s}_I$, et un triangle K tel que $\mathbf{s}_I$ n'est pas un sommet de K. Dans ce cas, φ_I est nulle sur les trois sommets de K, et est donc nulle sur le triangle tout entier.

Une illustration du support des fonctions de forme est donnée sur la Figure 3.

3.2.5 Conclusion

Pour une fonction u_h de V_h, retenons que :

- u_h est (par définition) continue et linéaire sur chaque triangle
- La dimension de V_h est égale au nombre de sommets N_s du maillage. Plus le maillage est fin, plus la dimension est grande.
- La famille $(\varphi_I)_I$ des fonctions de forme est une base de V_h. Autrement dit, il existe N_s uniques coefficients $(u_I)_I$, tels que

$$u_h = \sum_{I=1}^{N_s} u_I \varphi_I$$

- Une fonction u_h de V_h est caractérisée par sa valeur aux N_s sommets
- Les coefficients sont en fait la valeur de u_h aux sommets : $u_I = u_h(\mathbf{s}_I)$:

$$\sum_{I=1}^{N_s} u_h(\mathbf{s}_I)\varphi_I(\mathbf{s}_J) = u_h(\mathbf{s}_J)\varphi_J(\mathbf{s}_J) = u_h(\mathbf{s}_J).$$

- Le support d'une fonction de forme φ_I est l'union des triangles ayant pour sommets $\mathbf{s}_I$. Il est donc très petit par rapport à Ω.

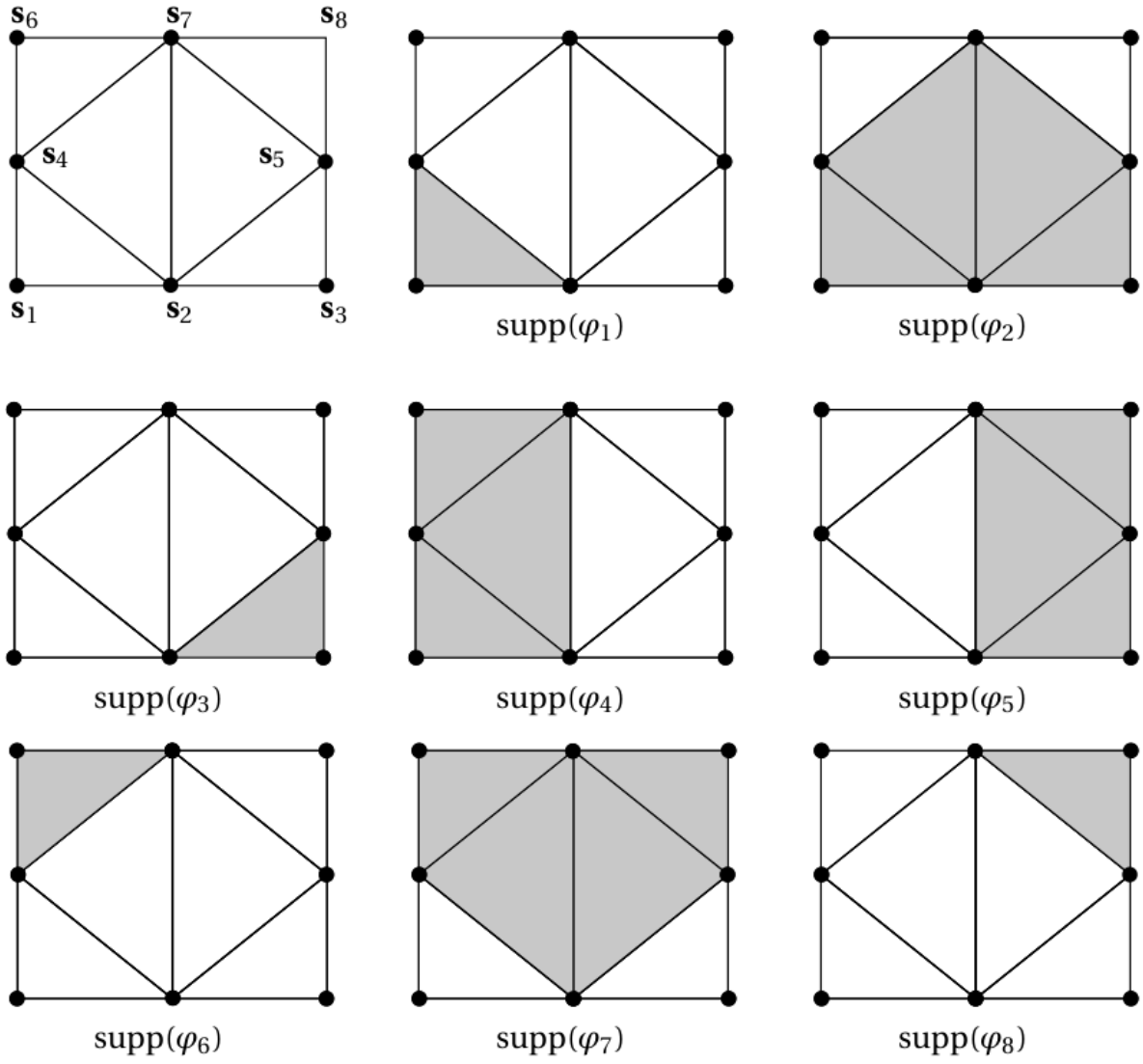

FIG. 3 – Support des fonctions de forme pour le maillage d'un carré.

3.3 Assemblage des Matrices

Nous devons maintenant calculer effectivement les coefficients $A_{I,J}$ de la matrice A et B_I du vecteur B. Nous nous intéressons pour l'instant uniquement à la matrice A.

3.3.1 Algorithme « brut-force »

Prenons deux indices de sommets I et J et rappelons la valeur du coefficient $A_{I,J}$:

$$A_{I,J} = a(\varphi_J, \varphi_I) = \int_\Omega \nabla\varphi_J \cdot \nabla\varphi_I + c\int_\Omega \varphi_J\varphi_I$$

Chaque intégrale sur Ω peut être décomposée comme une somme sur les triangles K_p :

$$A_{I,J} = \sum_{p=1}^{N_t} \int_{K_p} \nabla\varphi_J \cdot \nabla\varphi_I + c\sum_{p=0}^{N_t-1} \int_{K_p} \varphi_J\varphi_I$$
$$B_I = \sum_{p=1}^{N_t} \int_{K_p} f(x)\varphi_I(x)\mathrm{d}x.$$

Soit deux sommets $\mathbf{s}_I$ et $\mathbf{s}_J$ n'appartenant pas un même triangle, alors $\text{supp}(\varphi_I) \cap \text{supp}(\varphi_J) = \emptyset$. Autrement dit, $\varphi_I\varphi_J$ est toujours nul et donc le coefficient $A_{I,J}$ est nul ! Vue autrement, si deux sommets $\mathbf{s}_I$ et $\mathbf{s}_J$ ne sont pas connectés par une arête, alors $A_{I,J=0}$.

Les coefficients de A sont donc majoritairement nuls car deux sommets pris au hasard (dans le million d'un maillage) ne sont pas connectés. En moyenne de manière empirique, un nœud (ou sommet) est connecté au maximum à 6 à 8 autres nœuds (en 2D). Une conséquence directe est que **la matrice A est creuse**, c'est-à-dire qu'un nombre important de ses coefficients sont nuls. Une stratégie de stockage creux est donc à utiliser, ce que nous verrons plus loin. Une manière pratique est d'utiliser le format COO pour l'assemblage puis le format CSR pour l'algèbre linéaire et la résolution du système.

Nous devons bien entendu construire cette matrice : calculer chacun de ses coefficients et les stocker. Un algorithme naïf ou brut-force (mais naturel) pour calculer chaque coefficient est de boucler sur les sommets et et de remplir la matrice au fur et à mesure, c'est-à-dire de remplir les coefficients les uns après les autres. Il est présenté dans *l'algorithme brut-force*.

Il est à noter que la boucle sur les triangles pourraient être simplifiée en ne bouclant que sur les triangles ayant pour sommet $\mathbf{s}_I$ et $\mathbf{s}_J$. Cependant, cet algorithme a tout de même un coût en $O\left(N_s^2\right)$ ce qui est trop important pour être utilisable en pratique.

Code source 1 – Algorithme brut-force

```
For I = 1:N_s
  For J = 1:N_s
    A(I,J) = 0
    For p = 1:N_t
      A(I,J) += ∫_{K_p} (∇ ϕ_J·∇ ϕ_I) +∫_{K_p}(ϕ_J ϕ_I)
    EndFor
  EndFor
  B(I) = 0
  For p = 1:N_t
    B[I] += ∫_{K_p} (f ϕ_I)
  EndFor
EndFor
```

3.3.2 Algorithme d'assemblage

Une autre manière de procéder, que l'on appelle **assemblage**, se base sur une boucle sur les triangles plutôt que sur les sommets. Le principe est de parcourir les triangles et de calculer des **contributions élémentaires**, qui vont s'ajouter petit à petit dans la matrice A. Reprenons l'expression du coefficient $A_{I,J}$:

$$A_{I,J} = \sum_{p=1}^{N_t} \underbrace{\int_{K_p} \nabla\varphi_J \cdot \nabla\varphi_I}_{\text{Contrib. élémentaire}} + c \sum_{p=0}^{N_t-1} \underbrace{\int_{K_p} \varphi_J \varphi_I}_{\text{Contrib. élémentaire}}$$

Introduisons $a_p(\cdot,\cdot)$ la famille de forme bilinéaire suivante, pour $p = 1, \ldots, N_t$:

$$a_p(\varphi_J, \varphi_I) = \int_{K_p} \nabla\varphi_J(\mathbf{x}) \cdot \nabla\varphi_I(\mathbf{x})\mathrm{d}\mathbf{x} + c\int_{K_p} \varphi_J(\mathbf{x})\varphi_I(\mathbf{x})\mathrm{d}\mathbf{x}$$

Ensuite, nous réécrivons la matrice A sous la forme suivante

$$A = \sum_{I=1}^{N_s} \sum_{j=0}^{N_s-1} a(\varphi_J, \varphi_I)\mathbf{e}_I^T \mathbf{e}_J,$$

où $\mathbf{e}_I$ est le vecteur de la base canonique de $\mathbb{R}^{N_s}$. Nous avons alors

$$\begin{aligned} A &= \sum_{I=1}^{N_s}\sum_{J=1}^{N_s} a(\varphi_J, \varphi_I)\mathbf{e}_I^T \mathbf{e}_J \\ &= \sum_{I=1}^{N_s}\sum_{J=1}^{N_s}\sum_{p=1}^{N_t} a_p(\varphi_J, \varphi_I)\mathbf{e}_I^T \mathbf{e}_J \\ &= \sum_{p=1}^{N_t}\sum_{I=1}^{N_s}\sum_{J=1}^{N_s} a_p(\varphi_J, \varphi_I)\mathbf{e}_I^T \mathbf{e}_J \end{aligned} \tag{3}$$

Nous remarquons maintenant que $a_p(\varphi_J, \varphi_I)$ est nul dès lors que $\mathbf{s}_I$ ou $\mathbf{s}_J$ ne sont pas des sommets de K_p (car $\varphi_I\varphi_J = 0$ sur K_p). Finalement, la somme sur tous les sommets du maillage se réduit à une somme sur les 3 sommets du triangle K_p considéré.

Nous comprenons que nous devons maintenant travailler **localement** dans chaque triangle. Pour cela, nous avons besoin d'introduire une **numérotation locale** de chaque sommet une fonction `L2G` (*Local To Global*)permettant de basculer du local vers le global une fonction telle que, pour $p = 1, \ldots, N_t$ et $i = 1, 2, 3$:

$$\texttt{L2G}(p, i) = I \iff \mathbf{s}_i^p = \mathbf{s}_I$$

Ainsi, pour un triangle K_p, ses sommets sont numérotés $[\mathbf{s}_1^p, \mathbf{s}_2^p, \mathbf{s}_3^p]$ en numérotation locale ou $[\mathbf{s}_{\texttt{L2G}(p,1)}, \mathbf{s}_{\texttt{L2G}(p,2)}, \mathbf{s}_{\texttt{L2G}(p,3)}]$ en numérotation globale, comme le montre la figure 4. Nous distinguerons la numérotation globale par des lettres capitales (I, J) et la numérotation locale par des minuscules (i, j). Nous introduisons aussi les fonctions de forme locales :

$$\varphi_i^p = \varphi_{\texttt{L2G}(p,i)}|_{K_p}.$$

Remark 3.5

Pour mieux comprendre la différence entre numérotation locale et globale, une *application est disponible en ligne*.

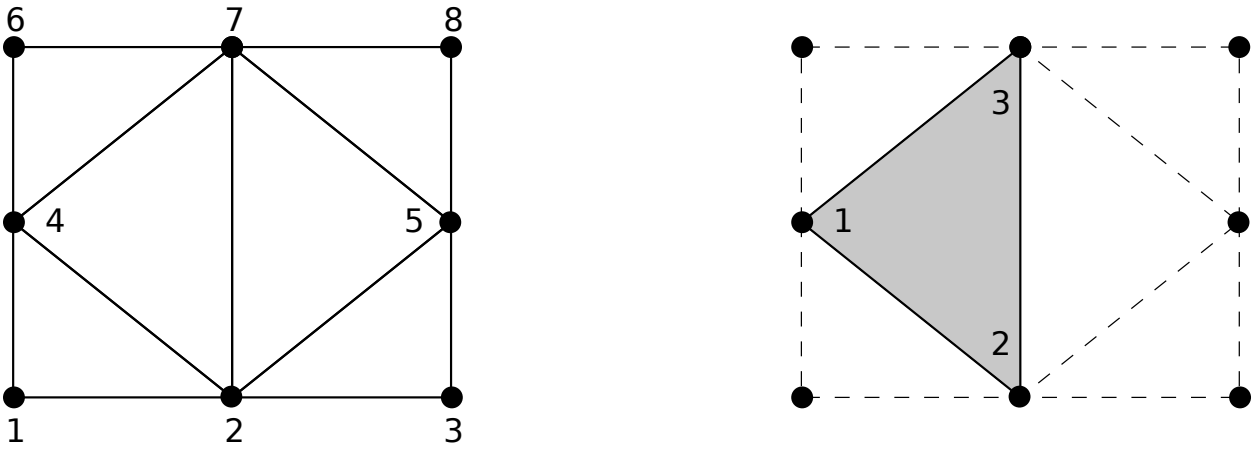

FIG. 4 – Numérotation locale et globale

Utilisons ces nouvelles notations dans l'équation (3), en ramenant la somme sur les sommets à uniquement les sommets du triangle considéré :

$$A = \sum_{p=1}^{N_t} \sum_{i=1}^{3} \sum_{j=1}^{3} a_p(\varphi_j^p, \varphi_i^p) \mathbf{e}_{\mathtt{L2G}(p,i)}^T \mathbf{e}_{\mathtt{L2G}(p,j)}$$

L'algorithme d'assemblage est alors complet ! Une version pseudo-code est présenté par *l'algorithme d'assemblage*. Sa complexité est en $O(N_t) \ll O\left(N_s^2\right)$. Comme *le premier algorithme*, il possède en plus l'avantage d'être parallélisable.

Code source 2 – Algorithme d'assemblage

```
A = 0
B = 0
For p = 1:N_t
  For i = 1:3
    I = L2G(p,i)
    For j = 1:3
      J = L2G(p,j)
      A(I,J) += a_p(ϕ_j^p,ϕ_i^p)
    EndFor
    B(I) += l_p(ϕ_i^p)
  EndFor
EndFor
```

Remark 3.6

Cet algorithme n'est pas encore utilisable, nous devons calculer la valeur de $a_p(\varphi_j^p, \varphi_i^p)$ et $\ell_p(\varphi_i^p)$. De plus, il manque encore les conditions de Dirichlet.

3.4 Calcul des Matrices Élémentaires

3.4.1 Matrices de Masse et de Rigidité

La matrice A peut être décomposée en deux matrices : la masse et la rigidité :

$$A = D + cM,$$

— M : la matrice de masse (ou de volume), de coefficient

$$M_{I,J} = \int_\Omega \varphi_J \varphi_I.$$

— D : la matrice de rigidité, de coefficient

$$D_{I,J} = \int_\omega \nabla\varphi_J \nabla\varphi_I.$$

Remark 3.7

Dans la littérature, cette matrice est souvent notée K, mais nous l'appelons D pour éviter toute confusion avec les triangles, nommés K également.

Remark 3.8

La matrice de masse M représente l'opérateur Identité dans la base des fonctions de forme (qui n'est pas orthogonale ni normée !). Pour s'en convaincre, il faut regarder « l'équation » $u = f$ (ou $Id.u = f$) et appliquer la méthode des éléments finis pour obenir la « formulation faible »

$$\forall v_h, \quad \int_\Omega U v_h = \int_\Omega f v_h,$$

qui aboutit au système linéaire suivant : $MU = B$. L'opérateur Identité, appliqué à u, est bien discrétisé en M.

Les **contributions élémentaires**, c'est à dire les quantités $a_p(\varphi_j^p, \varphi_i^p)$ et $\ell_p(\varphi_i^p)$, peuvent elles aussi être décomposées en deux parties. Pour rappel, les sommets d'un triangle K_p seront notés $[\mathbf{s}_0^p, \mathbf{s}_1^p, \mathbf{s}_2^p]$ et ordonnés dans le sens trigonométrique. Nous noterons $\mathbf{s}_i^p = (x_i^p, y_i^p)$ un sommet de K_p et φ_i^p la fonction de forme locale associée. Nous notons M_p^e et D_p^e les matrices de masse et de rigidité élémentaire du triangle K_p, de coefficient respectif $(M_p^e)_{i,j}$ et $(D_p^e)_{i,j}$ donnés par

$$(M_p^e)_{i,j} = \int_{K_p} \varphi_j^p \varphi_i^p$$

$$(D_p^e)_{i,j} = \int_{K_p} \nabla\varphi_j^p \cdot \nabla\varphi_i^p.$$

3.4.2 Matrice de masse élémentaire

Nous nous focalisons sur la matrice de masse, le principe est similaire pour la matrice D et est détaillé juste après.

Pour construire la matrice M, nous avons vu qu'il était préférable de parcourir les triangles plutôt que les sommets, autrement dit, plutôt que de calculer $M_{I,J}$ directement, mieux vaut calculer, pour tout triangle p, la **contribution élémentaire** $(M_p^e)_{i,j}$ pour $i, j = 1, 2, 3$, définie par :

$$(M_p^e)_{i,j} = \int_{K_p} \varphi_j^p(\mathbf{x})\ \varphi_i^p(\mathbf{x}) \mathrm{d}\mathbf{x}. \tag{4}$$

Chaque contribution élémentaire $(M_p^e)_{i,j}$ est ensuite ajoutée à $M_{I,J}$, avec $I = \texttt{L2G}(p, i)$ et $J = \texttt{L2G}(p, j)$.

Triangle de référence

Pour calculer la quantité élémentaire (4), plaçons nous tout d'abord dans un triangle « simple » $\widehat{K}$, appelé **triangle de référence**. Celui-ci est souvent choisi comme étant le triangle rectangle de sommets $\widehat{\mathbf{s}}_1 = (0,0)$, $\widehat{\mathbf{s}}_2 = (1,0)$ et $\widehat{\mathbf{s}}_3 = (0,1)$, ordonnés dans le sens trigonométrique. Pour différencier ce triangle d'un triangle du maillage, nous lui adjoignons un repère (ξ, η) dit **repère paramétrique**.

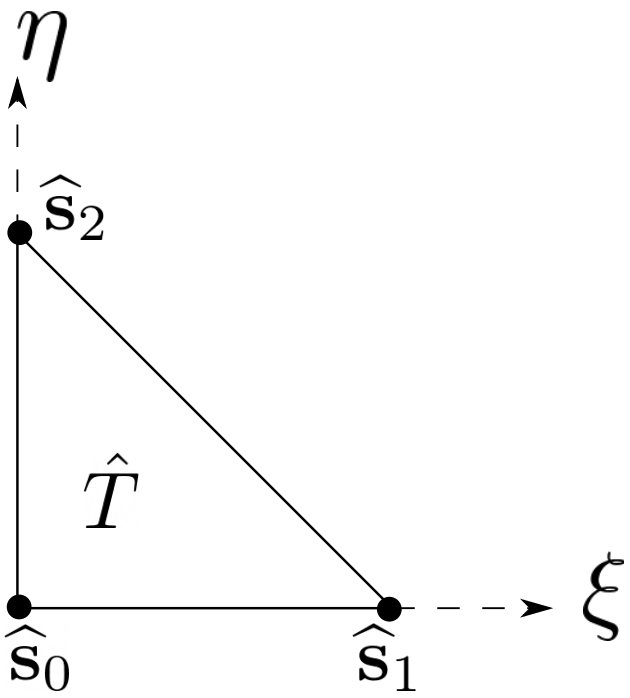

FIG. 5 – Triangle de référence $\widehat{K}$ et son repère paramétrique (ξ, η).

Nous notons $\widehat{\varphi}_i \in \mathbb{P}^1(\widehat{K})$ les trois fonctions de forme associées aux sommets $\widehat{\mathbf{s}}_i$, pour $i = 1, 1, 3$, définies par $\widehat{\varphi}_i(\widehat{\mathbf{s}}_j) = \delta_{ij}$. Ces fonctions $\widehat{\varphi}_i$ étant des polynômes de degré un, nous pouvons les calculer analytiquement :

$$\left\{ \begin{array}{l} \widehat{\varphi}_1(\xi,\eta) = 1 - \xi - \eta \\ \widehat{\varphi}_2(\xi,\eta) = \xi \\ \widehat{\varphi}_3(\xi,\eta) = \eta \end{array} \right.$$

Lemma 3.5

Dans le triangle $\widehat{K}$, la matrice de masse élémentaire $\widehat{M}^e = (\widehat{M}^e_{i,j})_{1 \le i,j \le 3}$ de coefficient

$$\widehat{M}^e_{i,j} = \int_{\widehat{K}} \widehat{\varphi}_j \widehat{\varphi}_i \mathrm{d}(\xi,\eta),$$

est donnée par

$$\widehat{M}^e = \frac{1}{24} \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}.$$

Proof. Prenons tout d'abord le cas $i = j = 2$, soit $\widehat{\varphi}_i(\xi, \eta) = \widehat{\varphi}_j(\xi, \eta) = \xi$. Dans ce cas :

$$\int_{\widehat{K}} \xi^2 \mathrm{d}(\xi, \eta) = \int_0^1 \int_0^{1-\xi} \xi^2 \mathrm{d}\eta \mathrm{d}\xi = \int_0^1 (1-\xi)\xi^2 \mathrm{d}\xi = \left[\frac{\xi^3}{3} - \frac{\xi^4}{4}\right]_0^1 = \frac{1}{3} - \frac{1}{4} = \frac{1}{12}.$$

Les calculs sont similaires pour $i = 1$ et $i = 3$. Prenons maintenant $i \neq j$, par exemple $i = 3$ et $j = 2$:

$$\int_{\widehat{K}} \xi\eta \mathrm{d}(\xi, \eta) = \int_0^1 \left(\int_0^{1-\xi} \eta \mathrm{d}\eta \right) \xi \mathrm{d}\xi = \frac{1}{2} \int_0^1 (1-\xi)^2 \xi \mathrm{d}\xi = \frac{1}{2} \left[\frac{1}{2} - \frac{2}{3} + \frac{1}{4} \right] = \frac{1}{24}.$$

Les calculs sont similaires pour les autres combinaisons.

Triangle quelconque

Changement de coordonnées. Soit un triangle K_p du maillage et supposons que nous disposions d'une transformation bijective et linéaire T_p permetteant de transformer le triangle de référence $\widehat{K}$ en K_p avec en plus $T_p(\widehat{\mathbf{s}}_i) = \mathbf{s}_i^p$ (conservation de l'ordre des sommets). Cette fonction T_p transforme les **coordonnées paramétriques** (ξ, η) en **coordonnées physiques** (x, y) avec $(x, y) = T_p(\xi, \eta) \in K_p$, et conserve « l'ordre des sommets ».

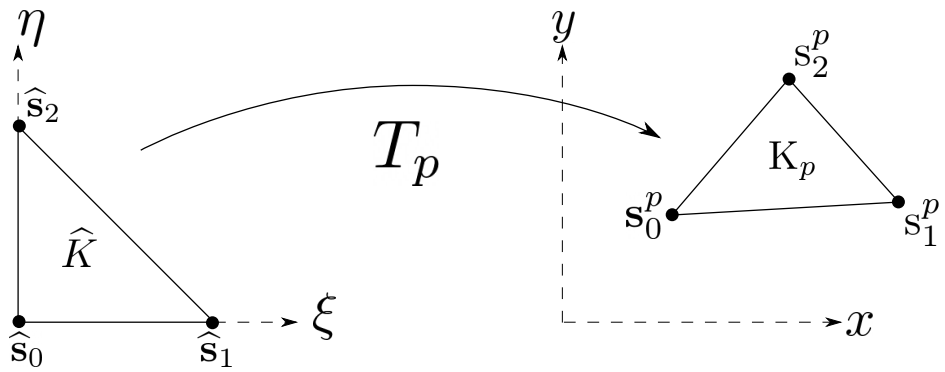

FIG. 6 – Transformation entre le triangle de référence $\widehat{K}$ et un triangle quelconque K_p.

Nous avons $\varphi_j^p(x, y) = \varphi_j^p(T_p(\xi, \eta))$ avec $\varphi_j^p \circ T_p \in \mathbb{P}^1(\widehat{K})$ et $\varphi_j^p \circ T_p(\widehat{\mathbf{s}}_i) = \delta_{ij}$, soit exactement les mêmes propriétés que les $\widehat{\varphi}_i$. Par unicité, nous avons $\varphi_j^p \circ T_p = \widehat{\varphi}_j$.

En notant J_p la matrice Jacobienne de T_p, alors la quantité $(M_p^e)_{i,j}$ peut alors s'écrire, par changement de variables :

$$(M_p^e)_{i,j} = \int_{K_p} \varphi_j^p(x, y)\varphi_i^p(x, y) \mathrm{d}(x, y) = |\det(J_p)| \underbrace{\int_{\widehat{K}} \widehat{\varphi}_j(\xi, \eta)\widehat{\varphi}_i(\xi, \eta) \mathrm{d}(\xi, \eta)}_{\text{Déjà calculé !}}$$

Ainsi, pour calculer la matrice élémentaire d'un triangle K_p quelconque, nous n'avons besoin que du déterminant de la Jacobienne : $\det(J_p)$.

Expression et Jacobienne de la transformation. La transformation que nous cherchons, T_p, est linéaire et « conserve » les sommets et leur ordre. Pour obtenir son expression, nous construisons des fonctions **d'interpolation géométrique**, $(\widehat{\psi}_i)_{1 \leq i \leq 3}$, linéaires sur $\widehat{K}$ et telles que :

$$\forall i, j = 1, 2, 3 \quad \widehat{\psi}_i(\widehat{\mathbf{s}}_j) = \delta_{ij}.$$

La transformation aura alors pour expression :

$$\begin{array}{rccl} T_p : & \widehat{K} & \to & K_p \\ & (\xi, \eta) & \mapsto & T_p(\xi, \eta) = (x, y) = \widehat{\psi}_1(\xi, \eta)\mathbf{s}_1^p + \widehat{\psi}_2(\xi, \eta)\mathbf{s}_2^p + \widehat{\psi}_3(\xi, \eta)\mathbf{s}_3^p. \end{array}$$

En d'autres termes, les fonctions d'interpolation géométrique $\widehat{\psi}_i$ sont ici identiques aux fonctions de forme $\widehat{\varphi}_i$:

$$\begin{cases} \widehat{\psi}_1(\xi,\eta) = 1-\xi-\eta \\ \widehat{\psi}_2(\xi,\eta) = \xi \\ \widehat{\psi}_3(\xi,\eta) = \eta \end{cases}$$

La matrice Jacobienne de la transformation est alors donnée par

$$J_p = \begin{pmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial x}{\partial \eta} \\ \dfrac{\partial y}{\partial \xi} & \dfrac{\partial y}{\partial \eta} \end{pmatrix} = \begin{pmatrix} x_2^p - x_1^p & x_3^p - x_1^p \\ y_2^p - y_1^p & y_3^p - y_1^p \end{pmatrix},$$

et son déterminant vaut

$$\begin{aligned} |\det(J_p)| &= |(x_2^p - x_1^p)(y_3^p - y_1^p) - (x_3^p - x_1^p)(y_2^p - y_1^p)| \\ &= 2|K_p| \neq 0, \end{aligned}$$

ce qui implique que le déterminant est non nul puisque le triangle n'est pas dégénéré : la transformation T_p est bien inversible.

Remark 3.9

Quand $\widehat{\psi}_i = \widehat{\varphi}_i$, nous parlons d'éléments finis **isoparamétriques**. Il convient de retenir que ce choix n'est pas obligatoire et les fonctions $\widehat{\psi}_i$ et $\widehat{\varphi}_i$ sont **indépendantes**. En particulier, pour obtenir des éléments courbes, les fonctions $\widehat{\psi}_i$ pourraient être quadratiques par exemple.

Expression finale de la matrice élémentaire.

Lemma 3.6

La matrice de masse élémentaire $M_p^e = ((M_p^e)_{i,j})_{0\leq i,j\leq 2}$ du triangle K_p a pour expression

$$M_p^e = \frac{|K_p|}{12} \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}.$$

3.4.3 Matrice de rigidité élémentaire

Nous appliquons la même procédure pour la matrice de rigidité D, autrement dit, nous calculons les matrices de rigidité élémentaire D_p^e définies par

$$(D_p^e)_{i,j} = \int_{K_p} \nabla\varphi_j^p(x,y) \cdot \nabla\varphi_i^p(x,y) \mathrm{d}(x,y).$$

Triangle de référence

Lemma 3.7

Dans le triangle de référence $\widehat{K}$, la matrice de rigidité élémentaire $\widehat{D} = (\widehat{D}_{i,j})_{1\leq i,j\leq 3}$ de coefficient

$$\widehat{D}_{i,j} = \int_{\widehat{K}} \nabla\widehat{\varphi}_j(\xi,\eta)\cdot\nabla\widehat{\varphi}_i(\xi,\eta)\mathrm{d}(\xi,\eta),$$

a pour expression

$$\widehat{D} = \frac{1}{2}\begin{pmatrix} 2 & -1 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$$

Proof. Les gradients des fonctions de forme $\widehat{\varphi}_j$ sont donnés par :

$$\nabla_{\xi,\eta}\widehat{\varphi}_0 = \begin{pmatrix}-1\\-1\end{pmatrix}, \quad \nabla_{\xi,\eta}\widehat{\varphi}_1 = \begin{pmatrix}1\\0\end{pmatrix}, \quad \nabla_{\xi,\eta}\widehat{\varphi}_2 = \begin{pmatrix}0\\1\end{pmatrix}.$$

La matrice étant symétrique, nous pouvons limiter les calculs à la partie triangulaire supérieure :

$$\begin{aligned}
\widehat{D}_{1,1} &= \int_{\widehat{K}} \nabla\widehat{\varphi}_1\cdot\nabla\widehat{\varphi}_1\mathrm{d}(\xi,\eta) = \int_{\widehat{K}}(-1,-1)\begin{pmatrix}-1\\-1\end{pmatrix}\mathrm{d}(\xi,\eta) = 2\int_{\widehat{K}}\mathrm{d}(\xi,\eta) &&= 1\\
\widehat{D}_{2,2} &= \int_{\widehat{K}} \nabla\widehat{\varphi}_2\cdot\nabla\widehat{\varphi}_2\mathrm{d}(\xi,\eta) = \int_{\widehat{K}}(1,0)\begin{pmatrix}1\\0\end{pmatrix}\mathrm{d}(\xi,\eta) = \int_{\widehat{K}}\mathrm{d}(\xi,\eta) &&= \frac{1}{2} = \widehat{D}_{3,3}\\
\widehat{D}_{1,2} &= \int_{\widehat{K}} \nabla\widehat{\varphi}_1\cdot\nabla\widehat{\varphi}_2\mathrm{d}(\xi,\eta) = \int_{\widehat{K}}(-1,-1)\begin{pmatrix}1\\0\end{pmatrix}\mathrm{d}(\xi,\eta) = -\int_{\widehat{K}}\mathrm{d}(\xi,\eta) &&= -\frac{1}{2}\\
\widehat{D}_{1,3} &= \int_{\widehat{K}} \nabla\widehat{\varphi}_1\cdot\nabla\widehat{\varphi}_3\mathrm{d}(\xi,\eta) = \int_{\widehat{K}}(-1,-1)\begin{pmatrix}0\\1\end{pmatrix}\mathrm{d}(\xi,\eta) = -\int_{\widehat{K}}\mathrm{d}(\xi,\eta) &&= -\frac{1}{2}\\
\widehat{D}_{2,3} &= \int_{\widehat{K}} \nabla\widehat{\varphi}_2\cdot\nabla\widehat{\varphi}_3\mathrm{d}(\xi,\eta) = \int_{\widehat{K}}(1,0)\begin{pmatrix}0\\1\end{pmatrix}\mathrm{d}(\xi,\eta) &&= 0.
\end{aligned}$$

Triangle quelconque

Pour calculer les dérivées partielles selon x et y de $\widehat{\varphi}_j$, nous utilisons la dérivée de fonction composée :

$$\begin{pmatrix}\dfrac{\partial\varphi_j^p}{\partial x}\\[2mm] \dfrac{\partial\varphi_j^p}{\partial y}\end{pmatrix} = \begin{pmatrix}\dfrac{\partial\xi}{\partial x} & \dfrac{\partial\eta}{\partial x}\\[2mm] \dfrac{\partial\xi}{\partial y} & \dfrac{\partial\eta}{\partial y}\end{pmatrix}\begin{pmatrix}\dfrac{\partial\widehat{\varphi}_j}{\partial\xi}\\[2mm] \dfrac{\partial\widehat{\varphi}_j}{\partial\eta}\end{pmatrix}$$

En notant B_p la matrice de passage, nous avons

$$\nabla_{x,y}\varphi_j^p(x,y) = B_p\nabla_{\xi,\eta}\widehat{\varphi}_j(\xi,\eta).$$

L'opération « inverse » nous donne :

$$\begin{pmatrix}\dfrac{\partial\widehat{\varphi}_j}{\partial\xi}\\[2mm] \dfrac{\partial\widehat{\varphi}_j}{\partial\eta}\end{pmatrix} = \begin{pmatrix}\dfrac{\partial x}{\partial\xi} & \dfrac{\partial y}{\partial\xi}\\[2mm] \dfrac{\partial x}{\partial\eta} & \dfrac{\partial y}{\partial\eta}\end{pmatrix}\begin{pmatrix}\dfrac{\partial\varphi_j^p}{\partial x}\\[2mm] \dfrac{\partial\varphi_j^p}{\partial y}\end{pmatrix} \iff \nabla_{\xi,\eta}\widehat{\varphi}_j(\xi,\eta) = (J_p)^T\nabla_{x,y}\varphi_j^p(x,y).$$

Nous en déduisons que $B_p = (J_p^T)^{-1}$, en particulier, dans le cas d'une transformation linéaire de triangle, nous obtenons :

$$B_p = \frac{1}{\det(J_p)} \begin{pmatrix} y_3^p - y_1^p & y_1^p - y_2^p \\ x_1^p - x_3^p & x_2^p - x_1^p \end{pmatrix}.$$

Au final, comme $X \cdot Y = X^T Y$, nous obtenons

$$\int_{K_p} (\nabla \varphi_j^p)^T \nabla \varphi_i^p \mathrm{d}(x,y) = |\det(J_p)| \int_{\widehat{K}} (\nabla \widehat{\varphi}_j)^T (B_p^T B_p) \nabla \widehat{\varphi}_i \mathrm{d}(\xi,\eta). \tag{5}$$

En éléments finis $\mathbb{P}^1$, les fonctions de forme sont linéaires et leur gradient est donc constant. Nous pouvons alors sortir les termes $\nabla \widehat{\varphi}_i$ et $\nabla \widehat{\varphi}_j$ de l'intégral pour obtenir le lemme suivant.

Lemma 3.8

Les coefficients a matrice de rigidité élémentaire $D_p^e = ((D_p^e)_{i,j})_{1 \le i,j \le 3}$ sont obtenus pas la relation suivante

$$\begin{aligned}(D_p^e)_{i,j} &= \int_{K_p} \nabla \varphi_j^p(x,y) \cdot \nabla \varphi_i^p(x,y) \mathrm{d}(x,y), \\ &= |K_p| \, (\nabla \widehat{\varphi}_j)^T (B_p^T B_p) \nabla \widehat{\varphi}_i.\end{aligned}$$

Proof. Pour les éléments finis $\mathbb{P}^1$, les gradients $\nabla \widehat{\varphi}_j$ sont constants et peuvent être sortis de l'intégrale. De plus, comme $|\det(J_p)| = 2\,|K_p|$ et $\left|\widehat{K}\right| = \frac{1}{2}$, nous avons

$$\int_{K_p} \nabla \varphi_j^p \cdot \nabla \varphi_i^p \mathrm{d}\mathbf{x} = |K_p| \, (\nabla \widehat{\varphi}_j)^T (B_p^T B_p) \nabla \widehat{\varphi}_i.$$

3.4.4 Second membre (ou RHS ou Membre de droite)

Étudions maintenant les termes du membre de droite comme

$$\int_{K_p} f(\mathbf{x}) \varphi_i^p(\mathbf{x}) \mathrm{d}\mathbf{x}.$$

Sauf pour certaines fonctions f particulières, nous ne pourrons certainement pas calculer explicitement ce terme, nous devons approcher cette intégrale à l'aide d'une formule de quadrature en passant à l'éléments de référence :

$$\begin{aligned}\int_{K_p} f(\mathbf{x}) \varphi_i^p(\mathbf{x}) \mathrm{d}\mathbf{x} &= |\det(J_p)| \int_{\widehat{K}} f(\mathbf{x}(\xi,\eta)) \widehat{\varphi}_i(\xi,\eta) \mathrm{d}(\xi,\eta) \\ &\simeq |\det(J_p)| \sum_{m=0}^{M-1} \omega_m f(\mathbf{x}(\xi_m,\eta_m)) \widehat{\varphi}(\xi_m,\eta_m).\end{aligned}$$

Les points (ξ_m, η_m) sont appelés **points de quadrature** (parfois **points de Gauss**, même si la règle de quadrature utilisée n'est pas de Gauss) et les quantités $\omega_m \in \mathbb{R}$ les **poids** associés. Notons que le point $\mathbf{x}(\xi_m, \eta_m)$ s'obtient par l'expression vue précédemment :

$$\mathbf{x}(\xi_m,\eta_m) = \sum_{i=0}^{2} \mathbf{s}_i^p \widehat{\psi}_i(\xi_m,\eta_m).$$

Nous présentons ici deux règles de quadrature pour l'intégrale $\int_{\widehat{K}} \widehat{g}(\mathbf{x})\mathrm{d}\mathbf{x}$ sur $\widehat{K}$ d'une fonction g quelconque. La première règle est exacte pour des polynômes de degré 1, la deuxième pour des polynômes de degré 2 (règles de Hammer) :

ξ_m	η_m	ω_m	Degré de précision
1/3	1/3	1/6	1
1/6	1/6	1/6	2
4/6	1/6	1/6	
1/6	4/6	1/6	

Remark 3.10

Les formules de quadrature ont évidemment un impact sur la qualité de l'approximation, toutefois, elles jouent un rôle relativement mineur par rapport aux autres approximations (et l'on peut choisir plus de points d'intégration !).

3.5 Matrice Creuse

La méthde des éléments finis mène à la résolution d'un problème linéaire du type :

$$Ax = b,$$

où la matrice A est **creuse**, c'est-à-dire que, majoritairement, les coefficients de A sont nuls. Pour minimiser la mémoire occupée par la matrice, seuls les coefficients non-nuls sont stockés. Ceci permet également d'améliorer notablement les performances du produit matrice-vecteur, en passant d'une complexité de $O(N^2)$ à $O(\texttt{nnz})$ où `nnz` est le nombre de coefficients non-nuls (`nnz` = number of non-zeros).

Il existe plusieurs formats de matrices creuses. Parmi les plus connus et utilisés : les formats COO (*COOrdinates*) et CSR (*Compressed Sparse Row* ou CRS pour *Compressed Row Storage*).

3.5.1 Format COO

Principe

Relativement naturel et simple à comprendre et utiliser. La matrice est stockée sous la forme de trois tableaux `row`, `col` et `val`, tous trois de taille `nnz` et contenant respectivement l'indice ligne, colonne et le coefficient non nuls de la matrice. En d'autre termes, pour i = 0, …, (`nnz`-1),

$$A(\texttt{row}[i], \texttt{col}[i]) = \texttt{val}[i]. \tag{6}$$

L'avantage de ce format est la facilité d'implémentation et la possibilité d'ajouter des coefficients « à la volée ». En effet, les tableaux `row`, `col` et `val` n'ont pas besoin d'être triés selon l'ordre indices.

Prenons la matrice exemple suivante avec `nnz` =10 :

$$A = \begin{pmatrix} 3 & 0 & 0 & 2 & 1 \\ 0 & 0 & 5 & 8 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 0 & 0 & 9 & 0 & 0 \\ 0 & 0 & 10 & 4 & 0 \end{pmatrix}. \tag{7}$$

Le stockage COO de cette matrice prendra alors la forme suivante :

Indice	0	1	2	3	4	5	6	7	8	9
`row`	0	0	0	1	1	2	2	3	4	4
`col`	0	3	4	2	3	1	2	2	2	3
`val`	3	2	1	5	8	1	2	9	10	4

Doublons

Le format COO peut autoriser les doublons, c'est-à-dire des coefficients ayant les mêmes indices ligne et colonne qu'un autre. En reprenant l'exemple ci-dessus et en divisant le dernier coefficient en deux, nous pourrions obtenir le stockage suivant :

Indice	0	...	8	9	10
`row`	0	...	4	4	4
`col`	0	...	2	3	3
`val`	3	...	10	1	3

Cette propriété est extrêmement pratique pour les éléments finis et l'algorithme d'assemblage ! En effet, chaque contributions élémentaires peut être ajouté aux tableaux `row`, `col` et `val`.

Du COO au Dense

Pour reconstuire la matrice sous format dense, le pseudo-code ci-dessous fonctionnerait et autorise d'avoir une dupplication de coefficients (du fait du +=) :

```
A = zeros(N,N)
for (i = 0; i < row.size(); i++)
  A(row[i], col[i]) = val[i]
end
```

Produit Matrice-Vecteur

Un pseudo code serait le suivant :

```
// y = A*x
y = zeros(n) // vecteur nul
for (i = 0; i < row.size(); i++)
  y[row[i]] += val[i] * x[col[i]]
end
```

Triplets

Plutôt que 3 tableaux, une matrice au format COO peut aussi être stockée sous forme d'un tableau de triplets (i,j,val), ce qui donnerait pour la matrice (7) :

Indice	0	1	2	3	4	5	6	7	8	9
Triplets	[0,0,3]	[0,3,2]	[0,4,1]	[1,2,5]	[1,3,8]	[2,1,1]	[2,2,2]	[3,2,9]	[4,2,10]	[4,3,4]

Conclusion

Le format COO est très souple et permet de construire une matrice aisément, cependant il présente les défauts suivants :
- Deux adressages indirects sont nécessaires pour effectuer le produit matrice vecteur
- Les accès aux données ne sont pas *a priori* connus
- Absence de méthode rapide pour obtenir un terme de la matrice connaissant ses indices ligne et colonne

Dans la pratique, le format COO est souvent utilisée comme format « tampon » pour stocker la matrice au format CSR, bien plus efficace pour les opérations d'algèbre linéaire. Le stockage **sous forme de triplets** est alors le plus pratique.

3.5.2 Format CSR

Principe

Le format CSR est spécialisé dans les opérations d'algèbres linéaires et pallie les défauts du COO. Son nom vient du fait que le tableau `row` est *compressé*. Une matrice au format CSR est composée des deux tableaux `col` et `val`, comme pour le COO et ordonnés par « lignes », et le tableau `row` est défini ainsi :
- Sa **taille est fixée à n+1** (n=nombre de lignes de la matrice)
- `row[i]` est maintenant **l'indice du premier élément non nul de la ligne** `i` **dans les tableaux** `col` **et** `val`

Par exemple, le stockage CSR de la matrice (7) est :

Indice	0	1	2	3	4	5	6	7	8	9
`row`	0	3	5	7	8	10				
`col`	0	3	4	2	3	1	2	2	2	3
`val`	3	2	1	5	8	1	2	9	10	4

Le tableau `row` est **compressé** par rapport au format COO puisque sa taille est maintenant de n+1, bien inférieure à `nnz` ! Sur une petite matrice, le gain mémoire est très faible, mais sur une matrice à plusieurs millions d'entrée, cette stratégie devient payante. D'autre part, l'absence de doublon de coefficients et le fait que les tableaux sont triés permettent d'améliorer significativement les opérations d'algèbres linéaires.

Du CSR au Dense

Le pseudo code pour reconstruire la matrice dense associé ressemblerait à ceci :

```
A = zeros(N,N)
for (i = 0; i < row.size() - 1; i++)
  for (j = row[i]; j < row[i+1]; j++)
    A(i, col[j]) = val[j]
  end
end
```

Produit Matrice - Vecteur

Le pseudo-code est le suivant

```
// y = A*x
y = zeros(row.size() - 1)
for (i = 0; i < row.size()-1; i++)
  for (j = row[i]; j < row[i+1]; j++)
    // Parcours des indices colonnes de la ligne i
    y[i] += val[j]*x[col[j]];
  end
end
```

Nous noterons que, cette fois-ci, les coefficients des vecteurs sont parcourus consécutivement.

Conclusion

Le format CSR est rigide : il est très coûteux d'ajouter des éléments dans la matrice. Ainsi et afin de ne pas perdre en efficacité, il est nécessaire de **connaître à l'avance l'emplacement des coefficients non nuls** de la matrice avant de la construire. En revanche, une fois construite, cette forme de stockage est très efficace.

3.5.3 Du COO au CSR

Principe

La souplesse du format COO permet de construire la matrice en ajoutant les triplets des coefficients (i,j,val) au fur et à mesure. Ensuite, une fois tous les triplets sauvegardés, ils sont triés (ou *assemblés*) et les doublons fusionnés. Il ne reste alors plus qu'à extraire les tableaux `row`, `col` et `val` du tableau de triplets et à compresser le vecteur `row` pour obtenir une matrice CSR.

Utilisation

En supposant les fonctions existantes, le pseudo-code suivant permet de passer d'une matrice A au format COO à une matrice B au format CSR :

```
MatriceCOO A(n) // COO
MatriceCSR B(n) // CSR
// Ajout des triplets
A.addTriplet(0,0,2.);
A.addTriplet(0,1,-1.1);
[...]
// Convertisseur en CSR
B = A.to_csr();
```

CHAPITRE 4

Conditions aux bords

Thug : Where is the money Lebowski ?

The Dude : It's uh… uh… it's down there somewhere, let me take another look.

---The Big Lebowski (movie)

4.1 Conditions de Neumann hétérogène

4.1.1 Théorie

Rajoutons maintenant la condition de Neumann hétérogène à notre problème ($g_N \neq 0$) :

$$\left\{ \begin{array}{rcll} -\Delta u + u & = & f & (\Omega), \\ \partial_{\mathbf{n}} u & = & g_N & (\Gamma := \partial\Omega). \end{array} \right.$$

Après multiplication par des fonctions test et intégration par partie, nous obtenons la formulation variationnelle

$$\int_\Omega \nabla u \cdot \nabla v + \int_\Omega uv - \int_{\partial\Omega=\Gamma} (\partial_{\mathbf{n}} u)v = \int_\Omega fv.$$

En utilisant la condition $\partial_{\mathbf{n}} u = g_N$ sur Γ, nous obtenons la formulation variationnelle suivante :

$$\left\{ \begin{array}{l} \text{Trouver } u \in H^1(\Omega) \text{ tel que} \\ \forall v \in H^1(\Omega), \quad a(u,v) = \ell(v), \end{array} \right. \tag{1}$$

avec

$$\begin{aligned} a(u,v) &:= \int_\Omega \nabla u \cdot \nabla v + \int_\Omega uv \\ \ell(v) &:= \int_\Gamma g_N v + \int_\Omega fv. \end{aligned}$$

Pour pouvoir appliquer le Théorème de Lax-Milgram, nous savons par le cas de Neumann homogène que l'application $a(\cdot,\cdot)$ est continue et coercive. Rien de neuf sous le soleil me direz-vous ? Oui mais non :

— Nous ne savons pas si $\ell(\cdot)$ est continue
— Pire encore, est-ce que le terme $\int_\Gamma g(\mathbf{x})v(\mathbf{x})\mathrm{d}s(\mathbf{x})$ a un sens quand v est dans $H^1(\Omega)$?

Nous n'avons en effet pas (encore) donné de sens à la trace (= la « restriction ») sur Γ d'une fonction de $H^1(\Omega)$, c'est-à-dire à $v|_\Gamma$. C'est l'objet du théorème ci-dessous (admis).

Theorem 4.1 (Continuité de la Trace (admis))

Soit $\Gamma \subset \partial\Omega$ une partie du bord de mesure non nulle au sens de la mesure de surface. Alors il existe une unique application $\gamma_\Gamma \colon H^1(\Omega) \to L^2(\Gamma)$ qui est continue au sens de $\|\cdot\|_{H^1(\Omega)}$:

$$\exists C > 0 \text{ tel que } \forall v \in H^1(\Omega),\ \|\gamma_\Gamma(v)\|_{L^2(\partial\Omega)} \leq C \|v\|_{H^1(\Omega)}.$$

Cette application est de plus caractérisée par

$$\forall \varphi \in \mathscr{C}^1(\overline{\Omega}), \qquad \gamma_\Gamma(\varphi) = \varphi|_\Gamma.$$

Ce théorème nous permet de montrer que la forme ℓ a un sens (chaque quantité existe) et est bien continue puisque, pour tout v de $H^1(\Omega)$:

$$\begin{aligned}
|\ell(v)| &\leq \left|\int_\Omega f(\mathbf{x})v(\mathbf{x})\mathrm{d}\mathbf{x}\right| + \left|\int_{\partial\Omega} g_N(\mathbf{x})v(\mathbf{x})\mathrm{d}\mathbf{x}\right| && \text{Inégalité Triangulaire}\\
&\leq \left|\int_\Omega f(\mathbf{x})v(\mathbf{x})\mathrm{d}\mathbf{x}\right| + \left|\int_{\partial\Omega} g_N(\mathbf{x})\gamma_{\partial\Omega}(v(\mathbf{x}))\right| && \text{Réécriture}\\
&\leq \|f\|_{L^2(\Omega)} \|v\|_{L^2(\Omega)} + \|g\|_{L^2(\partial\Omega)} \|\gamma_{\partial\Omega}(v)\|_{L^2(\partial\Omega)} && \text{Cauchy-Schwarz}\\
&\leq \left(\|f\|_{L^2(\Omega)} + C\|g\|_{L^2(\partial\Omega)}\right) \|v\|_{H^1(\Omega)} && \text{Cont. Trace.}
\end{aligned}$$

4.1.2 Implémentation dans le cas $\mathbb{P}^1$

Nous discrétisons la formulation faible (1) en remplaçant formellement $H^1(\Omega)$ par V_h pour obtenir

$$\begin{cases} \text{Trouver } u_h \in V_h \text{ tel que} \\ \forall v_h \in H^1(\Omega), \quad a(u_h, v_h) = \ell(v_h), \end{cases} \tag{2}$$

Nous appliquons la méthode vue précédemment pour obtenir un système linéaire équivalent à (2) :

$$AU = B.$$

Les coefficients de la matrice A et du vecteur B sont donnés par

$$\begin{aligned}
A = (A_{I,J})_{I,J}, \quad A_{I,J} = a(\varphi_J, \varphi_J) &= \int_\Omega \nabla\varphi_J \cdot \nabla\varphi_I + \int_\Omega \varphi_J\varphi_I\\
B = (B_I)_I, \qquad B_I = \ell(\varphi_I) &= \int_\Omega f\varphi_I \underbrace{+ \int_\Gamma g_N\varphi_I}_{\text{Nouveau !}}
\end{aligned}$$

, Au final, seule le membre de droite est modifié par rapport au cas de la condition de Neumann homogène. Autrement dit, la matrice A est identique et il nous suffit de savoir calculer $\int_\Gamma g_N\varphi_I$ pour obtenir le second membre : pour cela, nous utilisons une règle de quadrature sur des segments. La technique est la même que pour le calcul de $\int_\Omega f\varphi_I$.

Décomposons Γ en N_Γ segments (qui sont aussi des arêtes de triangles !) σ_q, $q = 1, \ldots, N_\Gamma$. Chaque segment a deux sommets indicés $[\mathbf{s}_1^{\sigma_q}, \mathbf{s}_2^{\sigma_q}]$. Nous notons $\varphi_i^{\sigma_q} = \varphi_I|_{\sigma_q}$ la restriction de la fonction de forme φ_I au segment q, tel que

$\mathbf{s}_i^{\sigma_q} = \mathbf{s}_I$. Nous pouvons alors décomposer B comme une somme de contributions élémentaires sur les triangles et les segments.

$$B = \sum_{p=1}^{N_t} \sum_{i=1}^{3} \int_{K_p} f\varphi_i^p + \sum_{q=1}^{N_\Gamma} \sum_{i=1}^{2} \int_{\sigma_q} g_N \varphi_i^{\sigma_q}.$$

Nous savons comment approcher la quantité de gauche avec une formule de quadrature adaptée. Le terme de droite s'approche lui aussi avec une formule de quadrature 1D, par exemple la formule « 1/3 Simpson », qui est exacte pour des polynômes de degré 2. Nous notons $|\sigma| = \left\|\mathbf{s}_1^{\sigma_q} - \mathbf{s}_2^{\sigma_q}\right\|$ la taille du segment et $\mathbf{s}_{12} = \frac{\mathbf{s}_1^{\sigma_q} + \mathbf{s}_2^{\sigma_q}}{2}$ le milieu du segment, alors la formule est la suivante :

$$\int_\sigma g \approx \frac{|\sigma|}{6} \left(g(\mathbf{s}_1^{\sigma_q}) + 4g(\mathbf{s}_{12}^{\sigma_q}) + g(\mathbf{s}_2^{\sigma_q}) \right)$$

4.2 Condition de Dirichlet

4.2.1 Homogène

Formulation Faible

Soit le problème suivant (notez l'absence du terme en u)

$$\begin{cases} -\Delta u &= f \quad (\Omega), \\ u &= 0 \quad (\Gamma := \partial\Omega). \end{cases} \tag{3}$$

Multiplions l'EDP (3) par des fonctions tests v, intégrons sur Ω et appliquons le Théorème de Green :

$$\begin{aligned} -\Delta u = f &\implies \forall v, \quad (-\Delta u)v = fv \\ &\implies \forall v, \quad -\int_\Omega (\Delta u)v = \int_\Omega fv \\ &\implies \forall v, \quad \int_\Omega \nabla u \cdot \nabla v - \int_\Gamma (\partial_{\mathbf{n}} u)v = \int_\Omega fv \end{aligned}$$

Nous sommes théoriquement bloqué. Nous sommes tentés de dire que $\partial_{\mathbf{n}} u = 0$ mais non seulement nous ne le savons pas, mais en plus c'est très probablement faux ! Utiliser les conditions aux bords est en revanche la bonne idée. Nous savons que u est nul sur le bord Γ. Autrement dit, nous ne cherchons pas la valeur de la solution sur ce bord, nous la connaissons déjà. Afin de conserver la symétrie entre u et v, imposons à v d'être aussi nul sur le bord et regardons ce que l'on obtient :

$$-\Delta u = f \implies \forall v, v|_\Gamma = 0 \quad \int_\Omega \nabla u \cdot \nabla v = \int_\Omega fv$$

En terme de dérivabilité, l'espace $H^1(\Omega)$ est suffisant pour la solution u et les fonctions tests v. Il manque toutefois la condition de Dirichlet, dite **essentielle**, qui doit être incluse dans l'espace fonctionnel. Pour cela, en rappelant que γ est l'application trace de $H^1(\Omega)$ sur $L^2(\Gamma)$, nous introduisons un espace de Sobolev qui prend en compte cette condition de Dirichlet.

$$H_0^1(\Omega) = \left\{ u \in H^1(\Omega) \;\middle|\; \gamma u = 0 \right\},$$

La formulation faible s'écrit alors

$$\begin{cases} \text{Trouver } u \in H_0^1(\Omega) \text{ tel que} \\ \forall v \in H_0^1(\Omega), \quad a(u,v) = \ell(v), \end{cases}$$

avec

$$\begin{array}{rccc} a\colon & H_0^1(\Omega)\times H_0^1(\Omega) & \longrightarrow & \mathbb{R} \\ & (u,v) & \longmapsto & \displaystyle\int_\Omega \nabla u\cdot\nabla v \\ \ell\colon & H_0^1(\Omega) & \longrightarrow & \mathbb{R} \\ & v & \longmapsto & \displaystyle\int_\Omega fv \end{array}$$

Remark 4.1

Attention, c'est parce que v est nul sur Γ que l'intégrale sur Γ s'annule. Ce n'est pas parce que $\partial_{\mathbf{n}} u = 0$! D'ailleurs, sauf si $u = 0$ partout, il y a fort à parier que $\partial_{\mathbf{n}} u \neq 0$!

Démontrons maintenant que cette formulation faible admet une unique solution. Commençons tout d'abord par montrer que $H_0^1(\Omega)$ est un espace de Hilbert.

Lemma 4.1

L'espace $H^1(\Omega)$ est de Hilbert

Ensuite, la continuité de ℓ a déjà été démontrée dans $H^1(\Omega)$ et donc dans $H_0^1(\Omega)$. Occupons nous de $a(\cdot,\cdot)$.

— Continuité de $a(\cdot,\cdot)$.

$$\begin{aligned} \forall u,v\in H_0^1(\Omega), |a(u,v)| &= \left|\int_\Omega \nabla u\cdot\nabla v\right| \\ &\leq \|\nabla u\|_{L^2(\Omega)}\|\nabla v\|_{L^2(\Omega)} && \text{Cauchy Schwarz} \\ &\leq \|u\|_{H^1(\Omega)}\|v\|_{H^1(\Omega)} && \text{Inégalité des normes} \end{aligned}$$

— Coercivité de $a(\cdot,\cdot)$:

$$\forall u\in H_0^1(\Omega), a(u,u) = \int_\Omega \nabla u\cdot\nabla u = \|\nabla u\|_{L^2(\Omega)}^2 \geq \dots$$

La coercivité est en réalité compliquée à obtenir puisque nous aimerions avoir :

$$\|\nabla u\|_{L^2(\Omega)}^2 \geq C\|u\|_{H^1(\Omega)}^2 = C\left(\|u\|_{L^2(\Omega)}^2 + \|\nabla u\|_{L^2(\Omega)}^2\right)$$

L'inégalité de Poincaré vient alors à notre rescousse !

Proposition 4.1 (Inégalité de Poincaré (admise))

Il existe une constante C ne dépendant que de Ω telle que

$$\forall u\in H_0^1(\Omega), \qquad \|\nabla u\|_{L^2(\Omega)} \geq C\|u\|_{H^1(\Omega)}$$

Remark 4.2

L'inégalité de Poincaré est également valable si la condition de Dirichlet n'est posée que sur une partie Γ_D du bord Γ. Dans ce cas, l'espace considéré est $H^1_{\Gamma_D}(\Omega) := \left\{v\in H^1(\Omega) \;\middle|\; \gamma_{\Gamma_D} v = 0\right\}$ où $\gamma_{\Gamma_D}\colon H^1(\Omega)\to L^2(\Gamma_D)$ est l'application trace sur Γ_D. À noter que $H^1_{\Gamma_D}(\Omega)$ est un Hilbert pour les mêmes raisons $H_0^1(\Omega)$ l'est.

Remark 4.3

L'inégalité de Poincaré montre que la semi-norme $v \mapsto \|\nabla v\|_{L^2(\Omega)}$ est une norme sur $H^1(\Omega)$ et est équivalente à la norme usuelle $\|\cdot\|_{H^1(\Omega)}$, puisque l'on a $\|\nabla v\|_{L^2(\Omega)} \geq C \|v\|_{H^1(\Omega)} \geq C \|\nabla v\|_{L^2(\Omega)}$.

L'inégalité de Poincaré implique la coercivité de $a(\cdot,\cdot)$. Toutes les hypothèses du théorème de Lax-Milgram sont vérifiées et la formulation faible du problème de Dirichlet homogène admet bien une unique solution.

Implémentation

Si V_h est l'espace des éléments finis $\mathbb{P}^1$ sur Ω, alors une discrétisation naturelle de $H^1_0(\Omega)$ est l'espace $V_{h,0}$ défini par

$$V_{h,0} = \{u \in V_h \mid \ u|_\Gamma = 0\}$$

Nous pouvons aussi raisonner sur le système linéaire directement. Nous séparons les degrés de liberté en deux sous-ensembles :

1. Ceux qui appartiennent à Ω ou à Γ_N : nous les noterons avec un indice I (pour Intérieur) : u_I
2. Ceux qui appartiennent à Γ_D, ils seront notés avec un indice D : u_D

Quitte à renuméroter, le vecteur U de degrés de liberté se réécrit

$$U = \begin{pmatrix} u_I \\ u_D \end{pmatrix},$$

et le système linéaire $AU = B$ devient :

$$AU = B \iff \begin{pmatrix} A_{I,I} & A_{I,D} \\ A_{D,I} & A_{D,D} \end{pmatrix} \begin{pmatrix} u_I \\ u_D \end{pmatrix} = \begin{pmatrix} B_I \\ B_D \end{pmatrix}$$

Les degrés de liberté u_D sont en réalité fixés à 0 du fait de la condition de Dirichlet, autrement dit, le système à résoudre se résume à ($I_{D,D}$ étant la matrice identité) :

$$AU = B \iff \begin{pmatrix} A_{I,I} & A_{I,D} \\ 0 & I_{D,D} \end{pmatrix} \begin{pmatrix} u_I \\ u_D \end{pmatrix} = \begin{pmatrix} B_I \\ 0 \end{pmatrix} \tag{4}$$

Informatiquement, nous devons donc rendre les lignes et colonnes associées aux degrés de liberté de Dirichlet, nulles, sauf sur la diagonale avec la valeur 1. Cette opération peut être effectuée après l'assemblage de la matrice ou lors de l'algorithme directement.

Remark 4.4

La valeur de 1 sur la diagonale est finalement arbitraire : nous pouvons choisir n'importe quelle valeur. Pour des raisons de précision numérique, il peut être plus pertinent de choisir comme valeur la moyenne de la somme de la diagonale de $A_{I,I}$ (sa trace). Cette technique peu coûteuse permet d'éviter de polluer le conditionnement de la matrice par des valeurs potentiellement trop grande ou trop petite par rapport à la « moyenne ».

Remark 4.5

Dans le cas de condition de Dirichlet homogène, ce système ce simplifie :

$$AU = B \iff \begin{pmatrix} A_{I,I} & 0 \\ 0 & I_{D,D} \end{pmatrix} \begin{pmatrix} u_I \\ u_D \end{pmatrix} = \begin{pmatrix} B_I \\ 0 \end{pmatrix},$$

ou encore, plus simplement : $A_{I,I}U_I = B_I$. Le système obtenu est de plus petite taille : c'est logique, l'espace $V_{h,0}$ est de dimension le nombre de sommets du maillage moins le nombre de sommets sur le bord Γ.

4.2.2 Condition hétérogène

Notion de relèvement

Nous considérons maintenant le cas d'une condition de Dirichlet non homogène, autrement dit, si $g \neq 0$:

$$\left\{ \begin{array}{rcll} -\Delta u & = & f & (\Omega) \\ u & = & g & (\Gamma) \end{array} \right. \tag{5}$$

Nous pouvons introduire l'ensemble suivant

$$H^1_{g_D}(\Omega) = \left\{ u \in H^1(\Omega) \;\middle|\; \gamma u = g \right\},$$

mais ce **n'est pas un espace vectoriel** ! Pour remédier à ce problème, nous nous ramenons au cas d'une condition de Dirichlet homogène en introduisant un **relèvement** (= une « extension », l'inverse d'une « restriction ») u_g de g : une fonction de $H^1(\Omega)$ telle que $\gamma u_g = g$. Nous ne nous préoccuperons pas de savoir si une telle fonction existe et supposons que tel est le cas. Le problème devient alors de chercher $u_t = u - u_g$ satisfaisant :

$$\left\{ \begin{array}{rcll} -\Delta u_t & = & f + \Delta u_g & (\Omega) \\ u_t & = & 0 & (\Gamma) \end{array} \right. \tag{6}$$

Nous avons vu plus haut que ce problème admet une unique solution, ce qui implique que (5) admet également une unique solution.

Remark 4.6

Le relèvement n'est pas unique, puisque si $u_0 \in H^1_0(\Omega)$ alors $u_g + u_0$ est aussi un relèvement acceptable.

Remark 4.7

Pour que le relèvement existe, il suffit que $g \in H^{1/2}(\Gamma)$. Cet espace est composé des traces sur Γ des fonctions de $H^1(\Omega)$:

$$H^{1/2}(\Gamma) = \left\{ \gamma v \;\middle|\; v \in H^1(\Omega) \right\}$$

Il contient naturellement $L^2(\Gamma)$ puisque $\gamma v \in L^2(\Gamma)$.

Relèvement en $\mathbb{P}^1$

En éléments finis $\mathbb{P}^1$, un relèvement naturel est la fonction $u_{h,g}$ de V_h telle que

$$u_{h,g}(\mathbf{s}_j) = \left\{ \begin{array}{ll} g(\mathbf{s}_j) & \text{si } \mathbf{s}_j \in \Gamma_D, \\ 0 & \text{sinon.} \end{array} \right.$$

Cette fonction n'est pas un relèvement de g puisqu'elle ne coïncide avec g que sur les sommets, mais pas nécessairement entre ceux-ci. Toutefois, au niveau discret, elle remplit ce rôle : c'est **un relèvement de l'interpolée** de g dans V_h (voir la section suivante). Nous notons g_h le vecteur de même taille que B_D et de coefficient $g(\mathbf{s}_I)$ avec $\mathbf{s}_I \in \Gamma$. En pratique, appliquer la condition de Dirichlet hétérogène se traduit par la décomposition de la matrice ainsi :

$$\begin{pmatrix} A_{I,I} & A_{I,D} \\ 0 & I_{D,D} \end{pmatrix} \begin{pmatrix} u_I \\ u_D \end{pmatrix} = \begin{pmatrix} B_I \\ g_h \end{pmatrix}.$$

La quantité g_h est ici à voir comme un vecteur de coefficient $g(\mathbf{s})$. La matrice obtenue est non symétrique, ce qui peut poser des problèmes (par ex. augmentation du coût de stockage mémoire). Une astuce simple consiste à réécrire sous la forme suivante :

$$\begin{pmatrix} A_{I,I} & 0 \\ 0 & I_{D,D} \end{pmatrix} \begin{pmatrix} u_I \\ u_D \end{pmatrix} = \begin{pmatrix} B_I - A_{I,D} g_h \\ g_h \end{pmatrix}.$$

Remark 4.8

Comme pour Dirichlet homogène, nous pouvons aussi nous contenter de résoudre un système plus petit : $A_{I,I}u_I = B_I - A_{I,D}g_h$.

Remark 4.9

Le terme $A_{I,D}g_h$ est la version discrète du terme Δu_g qui apparait dans (6). En effet, la matrice A discrétise l'opérateur $a(\cdot,\cdot)$ qui, ici, représente le laplacien sous sa forme faible $\int_\Omega \nabla u \cdot \nabla v$. Gardez à l'esprit que $A_{I,D}$ n'est pas carré et prend en argument un vecteur de la taille le nombre de sommets de Γ pour retourner un vecteur de taille le nombre de sommets du maillage.

4.3 Condition de Fourier

4.3.1 Problème

Étudions le problème suivant pour f et g suffisamment régulières :

$$\left\{\begin{array}{rcll} -\Delta u + u &=& f & (\Omega), \\ \partial_{\mathbf{n}} u + u &=& g & (\Gamma := \partial\Omega). \end{array}\right. \tag{7}$$

La condition de Fourier (ou Robin ou Fourier-Robin) s'écrit aussi $\partial_{\mathbf{n}} u = g - u$ sur Γ. Après calcul, la formulation variationnelle s'écrit

$$\left\{\begin{array}{c} \text{Trouver } u \in H^1(\Omega) \text{ tel que} \\ \forall v \in H^1(\Omega), a(u,v) = \ell(v), \end{array}\right.$$

avec (γ est l'application trace sur Γ) :

$$\begin{array}{rccl} a\colon & H^1(\Omega)\times H^1(\Omega) & \to & H^1(\Omega) \\ & (u,v) & \mapsto & \displaystyle\int_\Omega \nabla u\cdot\nabla v + \int_\Omega uv + \int_\Gamma \gamma(u)\gamma(v) \\ \ell\colon & H^1(\Omega) & \to & H^1(\Omega) \\ & v & \mapsto & \displaystyle\int_\Omega fv + \int_\Gamma g\gamma(v) \end{array}$$

4.3.2 Existence et unicité

Nous avons vu que dans la section 4.1 consacrée à Neumann hétérogène que les intégrales sur le bord ont un sens du fait de l'existence de l'opérateur trace. Pour une condition de Neumann, l'opérateur ℓ est le même et nous avons déjà vu dans la section 4.1 qu'il vérifie les hypothèses du Théorème de Lax-Milgram. Il ne nous reste qu'à vérifier que $a(\cdot,\cdot)$ est bilinéaire (trivial), continue et coercive.

— Continuité de $a(\cdot,\cdot)$ pour tout $u, v \in H^1(\Omega)$:

$$
\begin{aligned}
|a(u,v)| &= \left| \int_\Omega \nabla u \cdot \nabla v + \int_\Omega uv + \int_\Gamma \gamma(u)\gamma(v) \right| \\
&\leq \left| \int_\Omega \nabla u \cdot \nabla v + \int_\Omega uv \right| + \left| \int_\Gamma \gamma(u)\gamma(v) \right| && \text{Inégalité Triang.} \\
&\leq \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)} + \|\gamma(u)\|_{L^2(\Gamma)} \|\gamma(v)\|_{L^2(\Gamma)} && \text{Cauchy-Schwarz} \\
&\leq \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)} + C^2 \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)} && \text{Cont. Trace} \\
&\leq (1 + C^2) \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)}
\end{aligned}
$$

La constante C est la constante de continuité de l'opérateur Trace sur Γ définie dans le théorème **??**.

— Coercivité de $a(\cdot,\cdot)$, avec $u \in H^1(\Omega)$:

$$
\begin{aligned}
a(u,u) &= \int_\Omega \nabla u \cdot \nabla u + \int_\Omega uu + \int_\Gamma \gamma(u)\gamma(u) \\
&= \|u\|^2_{H^1(\Omega)} + \underbrace{\int_\Gamma |\gamma(u)|^2}_{\geq 0} \\
&\geq \|u\|^2_{H^1(\Omega)}
\end{aligned}
$$

Le problème admet donc une unique solution.

4.3.3 Matrice de masse sur le bord

Après discrétisation dans la base éléments finis, nous sommes ramenés à la résolution du système linéaire

$$AU = b,$$

où la matrice A et le vecteur b sont donnés par

$$
\begin{aligned}
A(I,J) &= \int_\Omega \nabla\varphi_J \cdot \nabla\varphi_I + \int_\Omega \varphi_J\varphi_I + \int_\Gamma \varphi_J|_\Gamma \varphi_I|_\Gamma \\
B(I) &= \int_\Omega f\varphi_I + \int_\Gamma g\varphi_I|_\Gamma
\end{aligned}
$$

Le vecteur B se calcule grâce aux formules de quadratures vues dans les paragraphes 3.4.4 pour les triangles et dans 4.1.2 pour les segments. La matrice A est obtenue par la somme de la matrice de rigidité D, de masse M et d'une dernière matrice M_Γ de coefficients :

$$M_\Gamma(I,J) = \int_\Gamma \varphi_J|_\Gamma \varphi_I|_\Gamma.$$

Cette matrice correspond à une matrice de masse sur le bord Γ. Nous pouvons tout d'abord remarquer que $\varphi_I|_\Gamma = 0$ dès que $\mathbf{s}_I$ n'est pas sur Γ. Comme toujours, nous préférons la décomposer en contributions élémentaires où, ici, un élément sera un segment :

$$M_\Gamma(I,J) = \sum_{\sigma\in\Gamma} \int_\sigma \varphi_J|_\sigma \varphi_I|_\sigma.$$

Nous pouvons maintenant remarquer que la somme sur les arêtes n'en est pas une puisque l'intégrale sur σ est nulle dès que $\mathbf{s}_I$ ou $\mathbf{s}_J$ n'est pas un sommet de l'arête. Cependant, n'oublions pas que nous ne calculons pas les coefficients un à un mais que nous *assemblons* la matrice, autrement dit, nous parcourons chaque segment, calculons toutes les contributions élémentaires associées à ce dernier, et additionnons le tout dans la grande matrice du système.

Autrement dit et quitte à renuméroter, nous considérons une arête $\sigma = [\mathbf{s}^\sigma_1, \mathbf{s}^\sigma_2]$, nous cherchons à calculer :

$$M^e_\sigma(i,j) = \int_\sigma \varphi^\sigma_j \varphi^\sigma_i,$$

avec $\varphi^\sigma_i = \varphi_I|_\sigma$ et $\mathbf{s}^\sigma_i = \mathbf{s}_I$. La matrice M^e_σ est de dimension 2x2.

4.3.4 Calcul de la matrice

Nous introduisons la coordonnée curviligne t

$$\forall \mathbf{x} \in \sigma, t(\mathbf{x}) = \frac{\|\mathbf{x} - \mathbf{s}_1^\sigma\|}{\|\mathbf{s}_1^\sigma - \mathbf{s}_2^\sigma\|} \in [0, 1].$$

Quand $\mathbf{x} = \mathbf{s}_1^\sigma$ alors $t = 0$ et $\mathbf{x} = \mathbf{s}_2^\sigma$ alors $t = 1$.

La trace d'une fonction de forme $\mathbb{P}_1$ sur σ est la « fonction chapeau » 1D classique. Plus précisément :

$$\begin{aligned}\varphi_1^\sigma(\mathbf{x}) &= \varphi_1^\sigma(\mathbf{x}(t)) = \hat{\phi}_1(t) = 1 - t \\ \varphi_2^\sigma(\mathbf{x}) &= \varphi_2^\sigma(\mathbf{x}(t)) = \hat{\phi}_2(t) = t\end{aligned}$$

À l'inverse, connaissant t on peut retrouver le point $\mathbf{x}$:

$$\mathbf{x}(t) = (1 - t)\mathbf{s}_1 + t\mathbf{s}_2.$$

Nous avons une transformation bijective entre σ et le segment $[0, 1]$ dit de référence et noté $\hat{\sigma}$. Nous pouvons opérer un changement de variable $(i, j = 1, 2)$:

$$\int_\sigma \varphi_j^\sigma \varphi_i^\sigma \mathrm{d}\mathbf{x} = |\sigma| \int_0^1 \varphi_j^\sigma(\mathbf{x}(t)) \varphi_i^\sigma(\mathbf{x}(t)) \mathrm{d}t = |\sigma| \int_0^1 \hat{\phi}_j(t) \hat{\phi}_i(t) \mathrm{d}t.$$

Les coefficients de masse de bord se calculent alors aisément et on obtient :

$$M_\sigma^e(i, j) = \frac{|\sigma|}{6} \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

CHAPITRE 5

Avancée

Und der Haifisch der hat Tränen
Und die laufen vom Gesicht
Doch der Haifisch lebt im Wasser
So die Tränen sieht man nicht
In der Tiefe ist es einsam
Und so manche Zähre fließt
Und so kommt es dass das Wasser
In den Meeren salzig ist

---Rammstein - *Haifisch*

5.1 Erreur comise et convergence

5.1.1 Méthode de Galerkin : Erreur commise

Nous considérons ici une formulation variationnelle abstraite :

$$\begin{cases} \text{Trouver } u \in V \text{ tel que} \\ \forall v \in V, \quad a(u,v) = \ell(v). \end{cases} \tag{1}$$

Les formes continues $a(\cdot,\cdot)$ et $\ell(\cdot)$ sont respectivement bilinéaire et linéaire, et $a(\cdot,\cdot)$ est de plus coercive. De cette manière, le Théorème de Lax-Milgram s'applique et le problème (1) admet une unique solution. Nous noterons $(\cdot,\cdot)_V$ et $\|\cdot\|_V$ respectivement le produit scalaire et la norme sur V.

Nous restons dans un cadre abstrait et introduisons V_h, un sous-espace fonctionnel de V, de Hilbert et de dimension finie. Nous appliquons la méthode de Galerkin pour obtenir la formulation faible « approchée » :

$$\begin{cases} \text{Trouver } u_h \in V_h \text{ tel que} \\ \forall v_h \in V_h, \quad a(u_h,v_h) = \ell(v_h). \end{cases} \tag{2}$$

Nous quantifions maintenant l'erreur commise en approchant u par u_h, ou plus exactement, V par V_h. Notons une propriété très intéressante de la solution approchée u_h :

Lemma 5.1

Soit u la solution exacte (*i.e.* solution de eqref{eq3-pbmodel}) et u_h la solution approchée (*i.e.* solution de (2)). Soit $e_h = u - u_h$ est l'erreur d'approximation, alors nous avons l'égalité suivante

$$\forall v_h \in V_h, \qquad a(e_h, v_h) = 0.$$

Proof. Comme $V_h \subset V$, nous pouvons choisir $v = v_h$ dans la formulation variationnelle eqref{eq3-pbmodel} :

$$\begin{aligned} \forall v_h \in V_h, \quad a(u - u_h, v_h) &= a(u, v_h) - a(u_h, v_h) \\ &= \ell(v_h) - \ell(v_h) \\ &= 0 \end{aligned}$$

Remark 5.1

Si $a(\cdot, \cdot)$ est symmétrique, le lemme précédent implique que l'erreur est orthogonale à V_h par rapport au produit scalaire $a(\cdot, \cdot)$.

Nous pouvons maintenant montrer que l'erreur d'approximation u_h de u est uniformément bornée par la distance entre u et l'espace V_h. Ce résultat est connu comme étant le Lemme de Céa, démontré par Jean Céa durant sa thèse, en 1964.

Lemma 5.2 (de Céa)

Soit u la solution exacte (*i.e.* solution de eqref{cv-pbmodel}) et u_h la solution approchée (*i.e.* solution de (2)). Nous avons

$$\|u - u_h\|_V \leq \frac{M}{\alpha} \inf_{v_h \in V_h} \|u - v_h\|_V ,$$

où M et α sont respectivement les constantes de continuité et de coercivité de $a(\cdot, \cdot)$ qui apparaissent dans le Théorème de Lax-Milgram.

Proof. Pour $v_h \in V_h$, la quantité $v_h - u_h$ est aussi un élément de V_h, ce qui implique d'après le lemme précédent que

$$\begin{aligned} a(u - u_h, u - u_h) &= a(u - u_h, u - v_h + v_h - u_h) \\ &= a(u - u_h, u - v_h) + a(u - u_h, v_h - u_h) \\ &= a(u - u_h, u - v_h). \end{aligned}$$

La coercivité et la continuité de $a(\cdot, \cdot)$ impliquent que

$$\begin{aligned} \forall v_h \in V_h, \quad \alpha \|u - u_h\|_V^2 &\leq |a(u - u_h, u - u_h)| \\ &\leq |a(u - u_h, u - v_h)| \\ &\leq M \|u - u_h\|_V \|u - v_h\|_V . \end{aligned}$$

Nous en déduisons le résultat cherché :

$$\forall v_h \in V_h, \qquad \|u - u_h\|_V \leq \frac{M}{\alpha} \|u - v_h\|_V .$$

Remark 5.2

Le point important du Lemme de Céa est de remplacer le problème d'estimation de l'erreur par un problème d'approximation. En effet, il nous suffit de montrer que la solution est « bien approchée » par les fonctions de V_h pour savoir que l'erreur ne sera *qu'une constante fois plus grande* que cette erreur d'approximation.

Nous pouvons maintenant donner une condition pour que la méthode de Galerkin converge.

Lemma 5.3

Soit $\Pi_h : V \to V_h$ un **opérateur d'interpolation** tel que

$$\forall v \in V, \qquad \lim_{h \to 0} \|v - \Pi_h v\|_V = 0,$$

alors la méthode de Galerkin converge, c'est-à-dire :

$$\lim_{h \to 0} \|u - u_h\|_V = 0.$$

Proof. C'est une conséquence directe du lemme de Céa, puisque :

$$0 \leq \|u - u_h\|_V \leq \frac{M}{\alpha} \|u - \Pi_h u\|_V \to 0 \quad (h \to 0).$$

La propriété demandée à l'opérateur de projection $\Phi_h : V \to V_h$ est assez naturel : plus h est petit et plus le projeté d'une fonction $v \in V$ doit être proche de v. Nous pouvons voir cela comme l'espace V_h est « proche » de V.

5.1.2 Converge des éléments finis $\mathbb{P}^1$

Afin de montrer que la méthode des éléments finis $\mathbb{P}^1$ converge, nous devons obtenir un opérateur d'interpolation et montrer qu'il vérifie l'hypothèse nécessaire du lemme 5.1.1.

TODO : !

5.2 Éléments Finis $\mathbb{P}^2$

Deuxième partie

Implémentation

CHAPITRE 6

Maillage avec GMSH

6.1 Prise en main de GMSH

Suivez ce tutoriel GMSH pour apprendre à :
- Générer de géométries simples (*e.g.* stade)
- Générer de géométries plus compliquées avec OpenCascade (*e.g.* tasse)
- Gerer des labels *Physical*
- Débuter avec l'API Python

6.2 API GMSH

Déjà abordé dans le tutoriel GMSH, nous nous intéressons à l'API Python de GMSH.

6.2.1 Fonction de Forme

Le but est d'afficher une fonction de forme $\mathbb{P}^1-$ Lagrange, c'est à dire une fonction φ_{IJ} de V_h qui vaut 0 sur tous les sommets $J \neq I$ du maillage sauf sur le sommet I, pour lequelle la fonction prend la valeur 1 :

$$\forall I, J = 0, \dots, N_s - 1, \qquad \varphi_I(\mathbf{s}_J) = \delta_{IJ}$$

Exercise 1.1

À l'aide de l'API Python de GMSH :
- Générez un carré unitaire avec un pas de maillage de 0.25
- Appliquez des labels `Physical` : un pour la surface et un pour son bord
- Générez le maillage 2D
- Construisez 3 numpy array `Phi`, `X` et `Y` unidimensionnel et de taille le nombre de sommets N_s du maillage tels que :

  - `Phi` est le vecteur nul sauf en un coefficient où `Phi[I] = 1` (choisissez le `I`)
  - `X` et `Y` sont respectivement les coordonnées x et y des points du maillage (voir ci-dessous)
- Affichez le tout à l'aide de Matplotlib et de la projection 3D

Pour obtenir les coordonnées des points d'un groupe `Physical` donné, vous pouvez utilisez `gmsh.model.mesh.getNodesForPhysicalGroup(dim, tag)` (voir gmsh.py) :

```
def getNodesForPhysicalGroup(dim, tag):
  """
  Get the nodes from all the elements belonging to the physical group of
              dimension `dim' and tag `tag'. `nodeTags' contains the node tags; `coord
↪'
              is a vector of length 3 times the length of `nodeTags' that contains␣
↪the x,
              y, z coordinates of the nodes, concatenated: [n1x, n1y, n1z, n2x, ...].

              Return `nodeTags', `coord'.
  """
```

Remark 1.1

- GMSH commence la numérotation des sommets à 1
- La liste retournée par `getNodesForPhysicalGroup` n'est pas triée

6.2.2 Interpolation $\mathbb{P}^1$

Prenons une fonction f définie sur Ω. Une interpolation possible de f sur l'espace $\mathbb{P}^1$ est la fonction $\Pi_h f$ telle que $\Pi_h f(\mathbf{s}) = f(\mathbf{s})$ pour chaque sommet $\mathbf{s}$ du maillage.

Exercise 1.2

Construisez l'interpollée $\Pi_h f$ de la fonction $f(x, y) = \sin(\pi x)\sin(\pi y)$

CHAPITRE 7

Solveur FEM Python

7.1 Matrices Creuses

Le but maintenant est d'implémenter la méthode des éléments finis P1 en 2D, autrement dit, à un maillage donné, de calculer :

- Les matrices de masse 2D (union de triangles) et 1D (union de segments)
- Les matrices de rigité (union de triangles) et 1D (union de segments)
- Appliquer les conditions de Dirichlet éventuelles
- Approcher des intégrales par des quadratures adaptées

La matrice du système finale sera stockée sous le format COOrdinate. L'algorithme d'assemblage se prête particulièrement bien au stockage COO puisque la redondance des coefficients est autorisée par Scipy. Autrement dit, les fonctions calculant les matrices de masse et de rigidité retourneront des triplets de type `[I, J, Valeur]` où `I` est l'indice ligne, `J` l'indice colonne et `Valeur` le coefficient (potentiellement partiel).

Avant de résoudre le système, la matrice sera transformée au format Compressed Storage Raw (CSR).

7.1.1 Matrice COO

Rappels

Le format COO (*COOrdinates*) propose de ne stocker que les coefficients non nuls d'une matrice `A` sous la forme de 2 listes d'entiers `row` et `col` et une liste de réels `val`, toutes trois de même taille, telles que

```
A[row[i], col[i]] = val[i]
```

Les trois listes peuvent aussi être combinées en une seule liste `data` de triplets de type `[int, int, double]` et telle que :

```
data[i] = [row[i], col[i], val[i]]
```

Le format COO est assez permissif :

— Redondance : si deux triplets possèdent les mêmes indices ligne et colonne alors le coefficient de la matrice, associé à ces indices ligne et colonne, sera obtenu en sommant les valeurs de ces triplets.
— Pas d'ordre : aucune nécessité d'ordonner les triplets selon les lignes ou les colonnes

Par exemple, les deux jeux de listes ci-dessous décrivent la même maitrce, la seule différence est que le coefficient en (2,2) est scindé en deux et le premier et le dernier triplet ont été permutés :

i	0	1	2	3	4	5	6
row[i]	0	0	1	2	3	3	3
col[i]	0	3	1	2	0	1	3
val[i]	1.1	2	1	2.3	0.5	2	2

i	0	1	2	3	4	5	6	7
row[i]	3	0	1	**2**	**2**	3	3	0
col[i]	3	3	1	**2**	**2**	0	1	0
val[i]	2	2	1	**1.3**	**1**	0.5	2	1.1

Scipy

Pour construire une matrice COO dans Scipy à partir d'un jeu de données `data`, ce dernier doit être de type `([], ([],[]))` : un `Tuple` contenant une `List` (`val`) ainsi qu'un `Tuple` contenant deux `List` (`row` et `col`) (list vs. tuple ?) :

```
data = (val, (row, col))
```

Une matrice COO se construit alors ainsi

```
A = coo_matrix(data)              # si data dans le format ci-dessus
A = coo_matrix((val, (row, col))) # si row, col et val sont séparées
```

Remark 2.15

Une matrice COO peut être visualisée en se transformant en `array` avec la méthode toarray() :

```
print(A.toarray())
```

Triplets

Nous proposons de construire notre future matrice par concaténation de triplets de type (I, J, valeur). une classe `Triplets` qui encapsule cette structure de données. Nous lui adjoignons une méthode `append` permettant d'ajouter un triplet au bout des autres :

```
Triplets t;           print(t.data) # ([], ([], []))
t.append(0, 1 ,2.);  print(t.data) # ([2.], ([0], [1]))
t.append(3, 4 ,5.2); print(t.data) # ([2., 5.2], ([0, 3], [1, 4]))
```

La classe ressemble alors à cela :

```
def class Triplet:
  def __init__():
    self.data = ([], ([], []))
  def __str__():
    return str(self.data)
  def append(self, I, J, val):
    # Ajoute le triplet [I, J, val] dans self.data
    # ...
```

Exercise 2.1

Construisez la classe `Triplet` et implémentez la méthode `append`. N'oubliez pas de tester votre classe.

Exercise 2.2

Testez votre classe `Triplet` en construisant la matrice suivante (au format COO évidemment) :

$$A = \begin{pmatrix} 1.1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2.3 & 0 \\ 0.5 & 2 & 0 & 2 \end{pmatrix}$$

7.1.2 Format CSR

Une fois la matrice au format COO construite, nous la transformerons au format CSR par la méthode tocsr() :

```
A = coo_matrix((val, (row, col))).tocsr()
```

7.2 Gestion du maillage

7.2.1 Une classe par élément

Nous proposons de construire 3 classes : `Point`, `Segment` et `Triangle` représentant un élément de type point, segment et triangle. La première étape consiste à transcrire les informations du maillage dans notre structure de données. Nous pouvons aussi ne pas passer par cette étape et n'utiliser que GMSH, cependant, cela nous donnera un meilleur contrôle par la suite.

La classe `Point` contient les coordonnées du point et un identifiant (0,1,2,3, …), son indice globale, qui correspondra en P1 à la ligne dans la matrice du degré de liberté (DOF). Les classes `Segment` et `Triangle` ont pour paramètres :

- Un identifiant (0,1,2,3…)
- Une liste orientée de `Point` qui le définissent
- Un tag `Physical` (ou -1 sinon)

L'identifiant n'a pas besoin d'être le même que dans GMSH, il est même plus logique que les identifiants soient consécutifs et incrémenté à chaque nouvelle instance (0, 1, 2, …). Le tag `Physical` doit en revanche être le même que GMSH pour éviter toute confusion. Vous êtes évidemment libre d'ajouter des paramètres et des méthodes à ces classes !

Par ailleurs, les 2 classes d'élément `Segment` et `Triangle`, disposeront au moins des méthodes suivantes :

— `area()` : calcule l'aire de l'élément (pour un segment, sa taille ; peut être stockée dans un paramètre et être calculé une fois pour toute)
— `jac()` : calcule son jacobien (son aire pour un segment, 2 fois l'aire pour un triangle)

7.2.2 Une classe pour les gouverner toutes...

Il peut être assez malin de construire une classe `Mesh` qui représente le maillage et permettra d'effectuer des recherches d'éléments. Cette classe aura pour paramètre trois listes : une liste de `Point`, une de `Segment` et une de `Triangle`. De plus, nous lui ajoutons deux méthodes :

— `Mesh.getElements(dim, physical_tag)` : retourne une liste de tous les éléments ayant la dimension `dim` (=1 pour segment, 2 pour triangle) et le tag physique `physical_tag`
— `Mesh.getPoints(dim, physical_tag)` : retourne uniquement les points du domaine de dimension `dim` et de label physique `physical_tag`

7.2.3 ...Et dans les ténébres les lier (à GMSH)

Le maillage étant construit avec GMSH, il s'agit maintenant de convertir les données issues de GMSH dans notre structure. Nous supposons ici que le maillage est déjà chargé en mémoire, soit parce qu'il vient d'être construit par GMSH (via `gmsh.model.mesh`) soit parce qu'il a été lu sur disque (via `gmsh.merge("fichier.msh")`). Nous proposons que le constructeur de `Mesh` soit vide (ne construit rien) et que l'instance de `Mesh` soit construite à l'aide d'une méthode `GmshToMesh(string filename)`, qui lit les données GMSH et construit les `Point`, `Segment` et `Triangle` et les ajoute dans les listes correspondent à `Mesh`. L'argument `filename` peut être optionnel si le maillage est déjà construit.

Remark 2.16

La méthode la plus délicate à construire est `GmshToMesh`. Pour vous aider un petit peu, n'hésitez pas à fouiller dans l'API de GMSH) :

— `gmsh.model.mesh.getNodes()` : retourne tous les noeuds
— `gmsh.model.getPhysicalGroups()` : retourne tous les groupes physiques avec leurs dimension et tag
— `gmsh.model.getEntitiesForPhysicalGroup(dim, physical_tag)` : retourne toutes les entités d'un groupe physique
— `gmsh.model.mesh.getElements(dim, tag)` : retourne tous les éléments de dimension `dim` (segments (`dim=1`), triangles (`dim=2`), ...) appartenant à l'entity de label `tag`.

L'algorithme ressemble surement à ceci :

```
// Création des points
For every Nodes
  Point(...)
End
//Création des éléments
For every Physical Entity
  For every Entity
    For every Element
```

7.3 Matrices de Masse et de Rigidité

7.3.1 Rappel : Algorithme d'assemblage

Plutôt que de calculer les coefficients un à un de la matrice A, l'algorithme d'assemblage propose de parcours chaque élément et d'ajouter leur contribution élémentaire à la grande matrice. Avec notre notation en `Triplets` cela donne :

```
Triplets t;
For p = 1, ... Nt  // Parcours des Triangles
  Mp = MatElem(p); // Matrice Elementaire du triangle p
  For i = 1,2,3
    I = Loc2Glob(p, i);
    For j = 1,2,3
      J = Loc2Glob(p,j);
      t.append(I, J, Mp(i,j)); // contribution élémentaire
    End
End
```

Si les élements parcourus sont ne sont pas des triangles à 3 points (segments, tétrahèdres, …), il suffit d'adapter le pseudo-code ci-desssus. Nous devons donc implémenter les calculs des matrices de masse et de rigidité élémentaire pour chaque élément.

7.3.2 Matrices de masse élémentaires

Construisez une fonction `mass_elem` prenant en argument un `Segment` ou un `Triangle`, un `Triplets` et un scalaire optionnel :

```python
# element = Segment ou Triangle ; triplets = Triplets ; alpha un scalaire optionnel
def mass_elem(element, triplets, alpha =1.):
    # ...
    # return triplets
```

Cette fonction calcule [les coefficients de la matrice élémentaire](http://bthierry.pages.math.cnrs.fr/course/fem/implementation_matrices_elementaires/) de l'élément (selon son type) et les ajoute à `triplets`.

Remark 2.17

Pour un élément donné, son type (`Segment` ou `Triangle`) est donné par son paramètre `name`.

7.3.3 Matrice de masse globale

Nous proposons de construire une fonction qui calcule toutes les contributions élémentaires de la matrice de masse d'un domaine de tag `Physical` et de dimension `dim` issue d'un maillage `msh`. Les coefficients partiels seront ajoutés sous forme de `Triplet` dans une liste envoyée en argument. Nous séparons pour le moment les calculs de la matrice de masse de ceux de la matrice de rigidité :

```python
# msh = Mesh, dim = int, physical_tag = int, triplets = Triplets
def Mass(msh, dim, physical_tag, triplets):
    #...
```

Exercise 2.3

Au boulot ! Assurez vous que la matrice de masse globale M associée au domaine Ω vérifie la relation suivante

$$U^T M.U = |\Omega|, \qquad U = [1, 1, 1, \ldots, 1]^T.$$

7.3.4 Matrice de Rigidité

Pour les matrices de rigidité, il faut calculer des quantités supplémentaires, comme la matrice B_p ou les gradients des fonctions de forme, par exemple :

```
def gradPhi(element, i:int):
    # ...
```

Exercise 2.4

Ajoutez les fonctionnalités dans votre code permettant de calculer les contributions élémentaires des matrices de rigidité puis la matrice globale.

Vérifiez que votre matrice de rigidité D satisfait bien la relation suivante :

$$DU = 0, \qquad U = [1, 1, 1, \ldots, 1]^T.$$

7.4 Quadratures

7.4.1 Rappel

Certaines intégrales ne peuvent être calculées analytiquement et devront être approchées numériquement via des règles de quadrature. Prenons pour exemple d'une fonction f quelconque mais connue, le second membre B (un vecteur) sera alors de la forme suivante, où $\mathbf{x} = (x, y)$,

$$\begin{aligned} B[I] &= \int_\Omega f(\mathbf{x})\varphi_I(\mathbf{x})\ \mathrm{d}\mathbf{x} = \sum_p \sum_i \int_{K_p} f(\mathbf{x})\varphi_i^p(\mathbf{x})\ \mathrm{d}\mathbf{x} \\ &= \sum_p |\det(J_p)| \sum_i \int_{\hat{K}} f(\mathbf{x}(\xi,\eta))\widehat{\varphi}_i(\xi,\eta)\ \mathrm{d}(\xi,\eta) \\ &\approx \sum_{K_p} |\det(J_p)| \sum_i \sum_m \omega_m f(\mathbf{x}(\xi_m,\eta_m))\widehat{\varphi}_i(\xi_m,\eta_m) \end{aligned} \tag{1}$$

Les poids ω_m et les points de quadrature (ξ_m, η_m) dépendent de la précision recherchée *comme expliqué dans le cours*. Rappelons aussi que $\mathbf{x}(\xi_m, \eta_m)$ s'obtient par les fonctions d'interpolation géométrique, qui dans le cas d'éléments finis isoparamétriques, sont les mêmes que les fonctions éléments finis $\mathbb{P}^1$, c'est à dire que pour $\mathbf{x}$ appartenant à un élément de sommets $(\mathbf{s}_i)_i$:

$$\mathbf{x}(\xi_m, \eta_m) = \sum_{i=1} \hat{\psi}_i(\xi_m, \eta_m)\mathbf{s}_i,$$

Les fonctions $(\hat{\psi}_i)_i$ sont égales au fonctions de forme $\mathbb{P}^1$ – Lagrange $(\widehat{\varphi}_i)_i$. Cependant, les fonction $(\hat{\psi}_i)_i$ sont des fonctions d'interpolation *géométriques* tandis que les fonctions de forme $(\hat{\varphi}_i)_i$ sont des fonctions d'interpolation de la solution.

7.4.2 Méthodes et fonctions utiles

Pour chaque type d'élément, `Triangle` ou `Segment`, nous avons besoin de méthodes permettant d'obtenir les quantités suivantes : :

1. Les poids des points de quadrature
2. Les coordonnées paramétriques des points de quadrature
3. Les coordonnées physiques des points de quadrature
4. Les valeurs des fonctions de forme de référence $\hat{\varphi}$ sur des coordonnées paramétriques

Remarquez que les points 1, 2 ne dépendent que du type de l'élément considéré.

Exercise 2.5

Afin de bien compartimenter chaque fonctionnalité, nous proposons :

— Ajouter aux classes `Triangle` et `Segment` la méthode `def gaussPoint(self,order=2):` qui retourne, dans le format de votre choix, les poids, les coordonnées paramétriques et les coordonnées physiques des points de Gauss de l'élement considéré et pour une précision `order`. Vous aurez sans doute besoin de méthodes intermédiaires pour calculer, par exemple les $\hat{\psi}_i(\xi, \eta)$.

— Ajouter une fonction `def phiRef(element, i:int, param:[float]):` qui calcule $\hat{\varphi}_i(\xi, \eta)$ sur un élément `Segment` ou `Triangle`. L'argument `param` est une liste des coordonnées paramétriques ((ξ, η) pour un triangle, s pour un segment))

7.4.3 Intégrale

Construisez maintenant une fonction de prototype suivant

```
def Integrale(msh:Mesh, dim:int, physical_tag:int, f, B:np.array, order=2):
```

Cette fonction calcul l'intégrale $\int f\varphi_I$ sur le domaine de tag physique `physical_tag` et de dimension `dim`. Le résultat est alors **ajouté** dans `B[I]` (voir (1)). L'argument `f` sera une fonction décrite par l'utilisatrice/utilisateur, elle prendra 2 arguments, `x` et `y` et retournera un scalaire correspondant à $f(x, y)$.

7.5 Conditions de Dirichlet

Pour prendre en compte les éventuelles condition de Dirichlet, nous avons besoin d'une fonction de prototype suivant

```
def Dirichlet(msh, dim, physical_tag, g, triplets, B):
```

Cette fonction prend comme argument le `Triplets triplets` et le vecteur `B` du système linéaire et les modifie pour prendre en compte la condition de dirichlet $u =$`g` sur le domaine de dimension `dim` et de tag physique `physical_tag`. La technique utilisée pour forcer cette condition est celle vue en cours.

Pour cela, nous parcourons les noeuds `I` du domaine de Dirichlet. Puis, dans la liste des indices ligne de `triplets`, dès qu'un occurence à `I` est obtenu, la valeur de ce triplet est mise à 0. Il ne faut pas oublier, à la fin, d'ajouter un triplet `(I,I,1)` correspondant au terme diagonal et de modifier le coefficient `B[I] = g(x,y)`.

Remark 2.18

Cette technique n'est peut être pas la plus optimale ! Mais elle a le mérite de fonctionner…

7.6 Résolution et Analyse

7.6.1 Problème de référence

Résumons ici l'utilisation de notre programme éléments finis sur le problème suivant :

$$\left\{ \begin{array}{rcll} -\Delta u + u & = & f & (\Omega) \\ u & = & 0 & (\partial\Omega) \end{array} \right. \tag{2}$$

La formulation variationnelle est donnée par

$$\left\{ \begin{array}{l} \text{Trouver } u \in H_0^1(\Omega) \text{ tel que} \\ \forall v \in H_0^1(\Omega), \qquad \displaystyle\int_\Omega \nabla u \cdot \nabla v + \int_\Omega uv = \int_\Omega fv \end{array} \right.$$

Pour simplifier nous prenons $\Omega =]0,1[\times]0,1[$ le carré unitaire et $f(x,y) = (1+2\pi^2)\sin(\pi x)\sin(\pi y)$ de sorte que la solution exacte est connue et vaut

$$u(x,y) = g(x,y) = \sin(\pi x)\sin(\pi y).$$

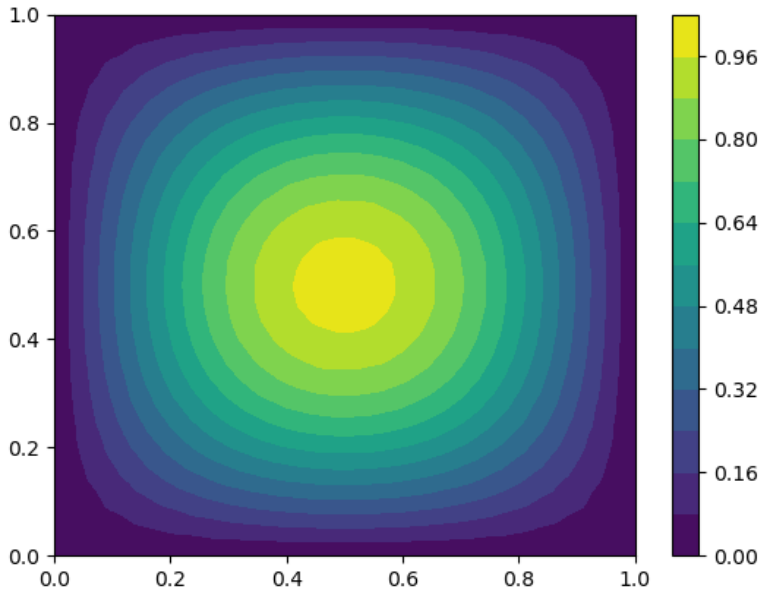

FIG. 1 – Solution

7.6.2 Résolution

Dans notre programme, cela reviendra à écrire quelque chose comme

```
#import ...

#Données
def g(x,y):
  return np.sin(np.pi*x)*np.sin(np.pi*y)
def f(x,y):
  return g(x,y)*(2*np.pi*np.pi +1 )
def diri(x,y):
  return 0.
#Maillage
msh = geo.mesher("mesh.msh")
# Triplets
t = common.Triplets()
fem_p1.Mass(msh, 2,10, t)
fem_p1.Stiffness(msh, 2,10, t)
b = np.zeros((msh.Npts,))
fem_p1.Integral(msh, 2, 10, f, b, 2)
fem_p1.Dirichlet(msh, t, b, 1, 1, diri)
# Résolution
A = (sparse.coo_matrix(t.data)).tocsr()
U = sparse.linalg.spsolve(A, b)

# Visualisation
x= [pt.x for pt in msh.points]
y= [pt.y for pt in msh.points]
connectivity=[]
for tri in msh.triangles:
  connectivity.append([ p.id for p in tri.p])

plt.tricontourf(x, y, connectivity, U, 12)
plt.colorbar()
plt.show()

### U de référence
Uref = np.zeros((msh.Npts,))
for pt in msh.points:
  I = int(pt.id)
  Uref[I] = g(pt.x, pt.y)
plt.tricontourf(x, y, connectivity, Uref, 12)
plt.colorbar()
plt.show()
```

7.6.3 Convergence

Exercise 2.6

1. **Pour différents pas de maillage**, calculez l'erreur en norme L^2 entre la solution exacte et la solution approchée pour le problème (2).
2. **Affichez** la courbe de l'erreur en fonction de h en échelle log-log.
3. **Calculez** la pente de la courbe et déduisez-en la vitesse de convergence par rapport au pas de maillage (h). Sauvegardez par ailleurs une copie de la courbe en format données (JSON ou autre) ou image (`PNG` par exemple,

pas de `JPG` nous ne sommes pas des sauvages !).

Troisième partie

Projet

CHAPITRE 8

2017 - 2018 : Wi-Fi

Ce projet propose d'étudier la propagation des ondes Wi-Fi dans un appartement de 35m^2. Nous ferons appels aux logiciels GMSH, pour la gestion du maillage et la visualisation, et FreeFem++ pour la résolution par éléments finis. Le sujet est décomposé en trois parties : la théorie (le modèle), l'implémentation et l'étude des résultats.

8.1 Équation de Helmholtz

La définition Wikipédia d'une onde est la suivante :

> Une onde est la propagation d'une perturbation produisant sur son passage une variation réversible des propriétés physiques locales du milieu. Elle se déplace avec une vitesse déterminée qui dépend des caractéristiques du milieu de propagation. Une onde transporte de l'énergie sans transporter de matière.

Mathématiquement, une onde $\mathscr{E}(\mathbf{x}, t)$ dépend du temps t et de l'espace $\mathbf{x}$, et vérifie l'équation des ondes :

$$\Delta\mathscr{E}(\mathbf{x}, t) = \frac{1}{c^2}\frac{\partial^2\mathscr{E}}{\partial t^2}(\mathbf{x}, t),$$

où $\Delta = \sum_{j=1}^{3}\frac{\partial^2}{\partial x_j^2}$ est l'opérateur Laplacien (spatiale) et c est la célérité de l'onde dans le milieu (peut dépendre de $\mathbf{x}$!). Par exemple, dans le cas d'une onde électromagnétique et dans le vide, c est la célérité de la lumière, soit 299792458m.s^{-1}. La quantité $\mathbf{x}$ est un vecteur de dimension $d = 2$ ou $d = 3$ selon le problème considéré : dans notre cas $d = 2$.

Lors d'une excitation périodique, c'est-à-dire lorsque la pulsation ω (en rad.s^{-1}) de l'onde est fixée, l'onde s'écrit alors $\mathscr{E}(\mathbf{x}, t) = \Re\left(u(\mathbf{x})e^{-\imath\omega t}\right)$ où $\imath = \sqrt{-1}$ et E est une **onde spatiale** satisfaisant l'équation de Helmholtz :

$$\Delta E + \frac{\omega^2}{c^2}E = f.$$

Cette équation s'obtient en remplaçant $\mathscr{E}(\mathbf{x}, t)$ par $E(\mathbf{x})e^{-\imath\omega t}$ dans l'équation des ondes. Nous notons en général $k = \frac{\omega}{c}$ (en rad.m^{-1}) le nombre d'onde et $\lambda = \frac{2\pi}{k}$ (en m) la longueur d'onde, autrement dit, la distance entre deux amplitudes, de sorte que l'équation de Helmholtz s'écrit

$$\Delta E + k^2 E = f.$$

Les ondes Wi-Fi qui suivent la norme IEEE 802.11g sont émises à une fréquence variant de 2.4GHz à 2.5GHz. Dans notre projet, nous nous limiterons à des ondes de fréquence 2.4GHz textbf{si votre machine vous le permet !} En effet, les simulations de propagation d'ondes sont parmis les plus coûteuses en terme de CPU. Si votre machine n'est pas assez puissante, nous prendrons une onde de fréquence plus faible, comme 1GHz voire moins.

Exercise 1.3

Sachant que $\omega = 2\pi F$ où F est la fréquence, en Hertz (Hz), calculez le nombre d'onde k et la longueur d'onde λ dans le vide, pour une onde électromagnétique et pour $F = 2.4$GHz et $F = 1$GHz.

8.2 Modèle

8.2.1 Entre les murs

Nous notons Ω l'appartement tout entier dans lequel est situé notre routeur. Les murs sont supposés être du même matériau : du placo-plâtre. Le domaine Ω est décomposé en deux domaines :

$$\Omega = \Omega_{\text{air}} \cup \Omega_{\text{mur}},$$

où Ω_{air} est l'intérieur de l'appartement, composé d'air (nous le supposerons vide de meubles), et Ω_{mur} contient les murs en placo-plâtre. Nous ne prendrons pas en compte les appartements voisins au nôtre. De plus, nous nous limiterons à la dimension 2 de l'espace.

L'air est modélisé comme étant le vide : $c_{\text{air}} = c$, tandis que, pour les murs en placo-plâtre, nous avons $c_{\text{placo}} = \frac{c}{2.4}$. Plutôt que de rendre le nombre d'onde dépendant de l'espace, nous introduisons la **fonction de contraste** n définie par :

$$n(\mathbf{x}) = \begin{cases} 1 & \text{si } \mathbf{x} \in \Omega_{\text{air}} \quad (\text{i.e. } \mathbf{x} \text{ est dans l'air}), \\ 2.4 & \text{si } \mathbf{x} \in \Omega_{\text{mur}} \quad (\text{i.e. } \mathbf{x} \text{ est dans le mur}). \end{cases}$$

Nous modifions alors l'équation de Helholtz ainsi :

$$\Delta E(\mathbf{x}) + k^2 n(\mathbf{x})^2 E(\mathbf{x}) = f(\mathbf{x}) \qquad \forall \mathbf{x} \in \Omega.$$

Le nombre d'onde k est donc ici celui de l'air (ou du vide).

8.2.2 Le routeur : la fonction f

Idéalement, le routeur devrait être modélisé comme un point source et donc mathématiquement par la distribution de Dirac $\delta_{\mathbf{x}_s}$ centré sur la position $\mathbf{x}_s$ du routeur. L'équation à résoudre serait alors :

$$\Delta E(\mathbf{x}) + k^2 n(\mathbf{x})^2 E(\mathbf{x}) = -\delta_{\mathbf{x}_s}.$$

Cependant nous ne pouvons pas utiliser de distribution de Dirac avec la méthode des éléments finis (car $\delta_{\mathbf{x}_s} \notin L^2(\Omega)$). Nous modélisons alors le routeur par un disque Ω_s de rayon $\varepsilon_s = 0.1$ et de centre $\mathbf{x}_s = (x_s, y_s)$ et la distribution $\delta_{\mathbf{x}_s}$ est approchée par la **fonction chapeau** f_s définie par

$$\forall \mathbf{x} \in \Omega, \qquad f_s(\mathbf{x}) = \begin{cases} \dfrac{3}{\pi \varepsilon_s^2} \left(1 - \dfrac{\|\mathbf{x}_s - \mathbf{x}\|}{\varepsilon_s} \right) & \text{si } \mathbf{x} \in \Omega_s, \\ 0 & \text{sinon.} \end{cases}$$

La fonction f_s est d'intégrale totale égale à 1 et quand ε_s tend vers 0, la suite de fonction $(f_s)_{\varepsilon_s}$ tend vers la distribution de Dirac $\delta_{\mathbf{x}_s}$.

Exercise 1.4

Montrez que :

$$\int_\Omega f_s(\mathbf{x})\mathrm{d}\mathbf{x} = 1.$$

Pour simplifier, on pourra se placer dans le cas où Ω_s est centré en $(0,0)$ et utiliser un changement de coordonnées adéquat.

8.2.3 Réflexions parasites

Sur le bord extérieur de notre appartement, nous souhaitons que l'onde ne se réfléchisse pas, pour cela nous imposons la condition aux limites suivantes :

$$\partial_{\mathbf{n}} E(\mathbf{x}) - \imath k n(\mathbf{x}) E(\mathbf{x}) = 0, \qquad \text{sur } \partial\Omega.$$

Cette condition est une approximation de la condition non réfléchissante **exacte** : l'onde sera légèrement réfléchie, toutefois la mise en oe{}uvre d'une telle condition reste très simple et peu coûteuse, ce qui explique que nous l'utilisions. Notez que $\mathbf{n}$ est le vecteur normale unitaire sortant à Ω (ne pas confondre avec n la fonction de constraste).

8.2.4 Le système à résoudre

Au final, le système que nous devons résoudre est le suivant. Nous cherchons l'onde spatiale textbf{complexe} E, telle que, pour $k = \omega/c$ donné, elle vérifie :

$$\left\{ \begin{array}{rcll} \Delta E(\mathbf{x}) + k^2 n(\mathbf{x})^2 E(\mathbf{x}) & = & -f_s(\mathbf{x}) & \text{dans } \Omega, \\ \partial_{\mathbf{n}} E(\mathbf{x}) - \imath k n(\mathbf{x}) E(\mathbf{x}) & = & 0 & \text{sur } \partial\Omega. \end{array} \right. \tag{1}$$

8.2.5 Formulation variationnelle

Exercise 1.5

Montrez que la formulation variationnelle du problème (ref{eq:helmholtz}) s'écrit

$$\left\{ \begin{array}{l} \text{Trouver } u \in H^1(\Omega) \text{ tel que} \\ \forall v \in H^1(\Omega), \quad a(u,v) = \ell(v), \end{array} \right. \tag{2}$$

avec

$$\begin{aligned} a(u,v) &= -\int_\Omega \nabla E(\mathbf{x}) \cdot \overline{\nabla v(\mathbf{x})}\mathrm{d}\mathbf{x} + \int_\Omega k^2 n(\mathbf{x})^2 E(\mathbf{x}) \overline{v(\mathbf{x})}\mathrm{d}\mathbf{x} + \int_{\partial\Omega} \imath k n(\mathbf{x}) E(\mathbf{x}) \overline{v(\mathbf{x})}\mathrm{d}\mathbf{x}, \\ \ell(v) &= -\int_\Omega f_s(\mathbf{x}) \overline{v(\mathbf{x})}\mathrm{d}\mathbf{x}. \end{aligned}$$

Exercise 1.6

Montrez que les applications $a(\cdot,\cdot)$ et $\ell(\cdot)$ sont continues sur respectivement $H^1(\Omega) \times H^1(\Omega)$ et $H^1(\Omega)$.

Malheureusement pour nous, $a(\cdot,\cdot)$ n'est pas coercive : nous ne pouvons pas appliquer le Théorème de Lax-Milgram et de ses corollaires. Nous admettrons le Théorème suivant

Theorem 1.1

Les problèmes (1) et (2) admettent une unique solution.

8.3 Implémentation

8.3.1 Le pas de maillage

Pour résoudre des problèmes de propagation d'ondes, le pas de maillage (ou la finesse de maillage), noté h (diamètre du plus grand élément), dépend du nombre d'onde k, ou plutôt, de la longueur d'onde $\lambda = 2\pi/k$. En effet, si nous ne prenons pas assez de points de discrétisation, l'onde ne sera pas suffisamment approchée comme le montre la Figure 1. Cela rejoint le Théorème de Shanon d'échantillonage.

En pratique, nous prenons un nombre de points $10 \leq n_\lambda \leq 20$ par longueur d'onde. Autrement dit, nous avons, si h est la taille caractéristique d'un élement :

$$h = \frac{\lambda}{n_\lambda} = \frac{2\pi}{k n_\lambda}.$$

Il faudra veiller, lors de nos simulations, à ce que $n_\lambda = 10$ textbf{au minimum}. Pour les simulations finales à haute fréquence, il vaut mieux privilégier $n_\lambda = 15$ ou $n_\lambda = 20$, si la machine le permet.

Remark 1.2

Avant de lancer un calcul et/ou un maillage pour tester (*i.e* mon code plante-il ?) : choisissez un nombre d'onde faible ! Par exemple $k = 10$ ou $k = 5$ (mais pas $k = 0$!).

8.3.2 L'appartement : GMSH

Le plan de l'appartement que nous considérons est donné par la Figure 2 où chaque dimension est exprimée en mètre. Nous proposons les contraintes suivantes sur l'appartement : sa longueur L est fixée à 7, sa largeur ℓ à 5 et l'épaisseur d des murs est égale à 0.2.

Exercise 1.7

Implémentez un code GMSH qui reproduit le plan de la figure 2, en respectant las consignes suivantes :

1. Nous devons être en mesure de pouvoir modifier via l'interface graphique les quantités suivantes :

— Le nombre de points n_λ de discrétisation par longueur d'onde λ. N'oubliez pas que n_λ est un entier.
— La position du routeur $\mathbf{x}_s$ (**Il n'est pas nécessaire de contraindre/vérifier sa position par rapport aux murs (mais par rapport aux dimensions de l'appartement, oui !), nous supposerons l'utilisateur suffisamment malin pour cela**)
— Remarque : dans la version originale du projet, il était demandé de pouvoir modifier les dimensions des pièces et donc de pouvoir déplacer les murs.

2. La largeur des portes est fixées à 80cm.
3. Le rayon de Ω_s est fixé à 0.1cm.

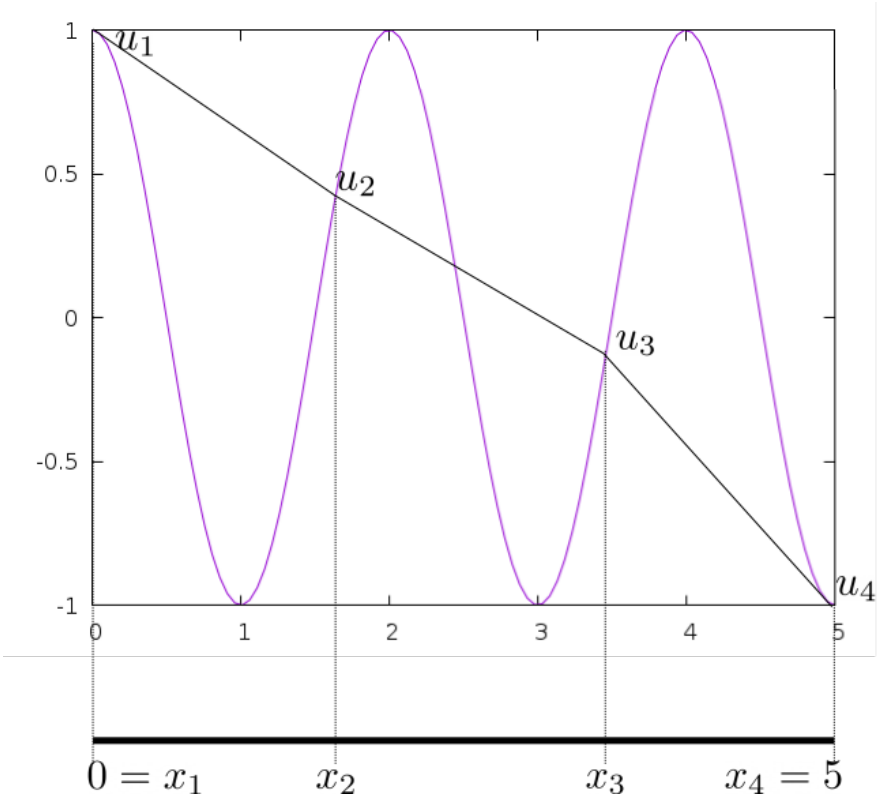

FIG. 1 – Illustration d'un échantillonage trop faible en 1D : la solution discrète est linéaire et non oscillante ! Elle est trop éloignée de la solution exacte. Nous avons besoin de plus de points de discrétisations.

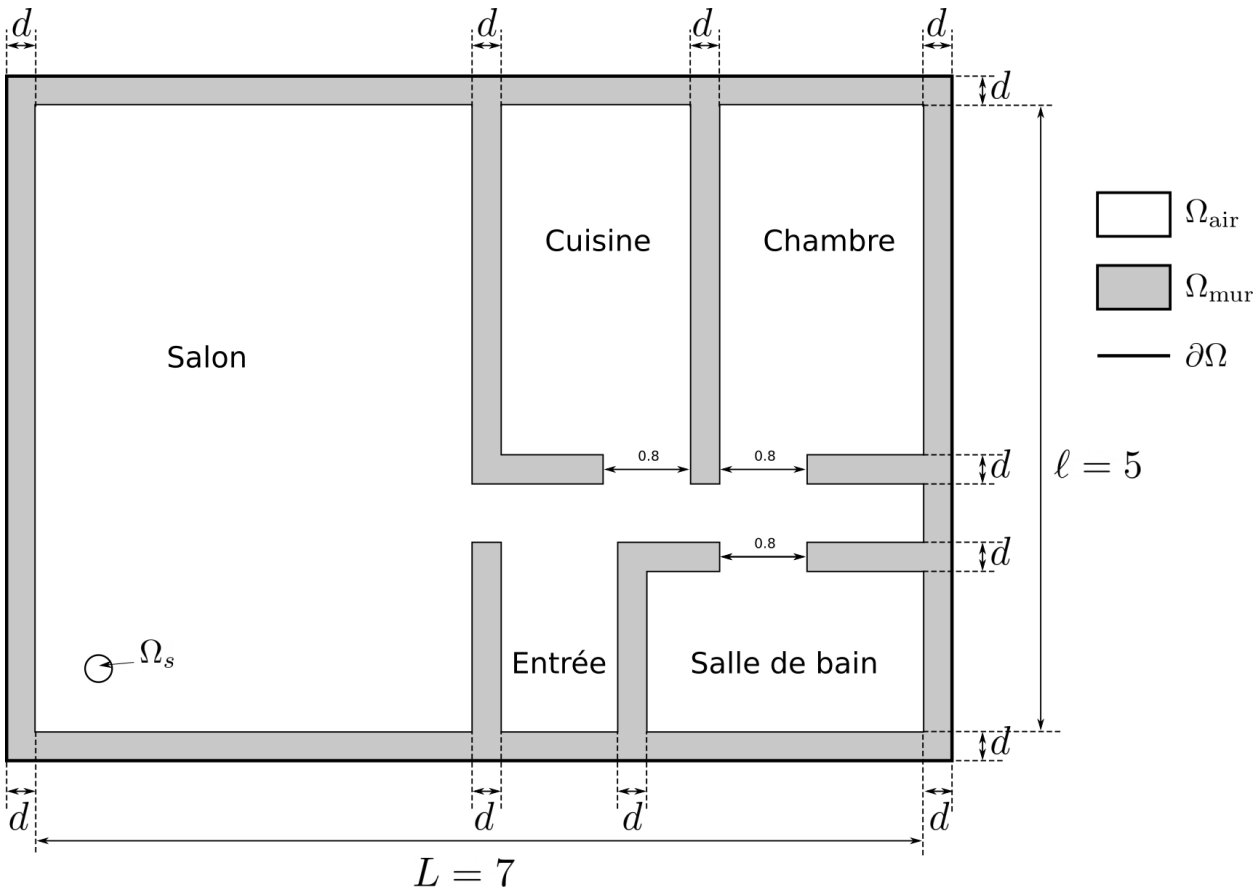

FIG. 2 – Plan de l'appartement (en mètre). Rappelons que $\Omega = \Omega_{\text{air}} \cup \Omega_{\text{mur}}$ et $\partial\Omega$ est le bord « externe » de l'appartement.

4. Tous les murs sont de même épaisseur d.

Remark 1.3

À vous de définir les entités `Physical` qui vous semblent d'importance ainsi que les dimensions des pièces.

8.3.3 La formulation variationnelle : FreeFem++

Exercise 1.8

Rédigez le code freefem permettant de résoudre le problème (2) dans le maillage réalisé par gmsh. Les quantités en sorties qui nous intéressent sont :

— Le champ u : partie réelle et partie imaginaire, dans tout l'appartement.
— La valeur absolue du champ : $|u|$, dans tout l'appartement.

Remark 1.4

Plusieurs remarques :

— Vous pouvez rédiger un script qui modifie le nombre d'onde k dans le(s) fichier(s) freefem et gmsh, puis lance gmsh et freefem… En bref : vous êtes libre de geeker comme bon vous semble.
— En phase de test, rappelez-vous de textbf{ne pas lancer} de résolution du problème pour un k élevé.
— Mettez la solution à l'échelle : faites en sorte que le maximum de la solution soit égale à 1.
— La solution que l'on calcule présente très certainement un pic au niveau du routeur, qui empêche de visionner correctement la solution en dehors de celui-ci. Il peut, dès lors, être intéressant de regarder la solution obtenue **partout en dehors** du routeur et/ou de modifier la *range* de couleur.

8.4 Étude et examen oral

Maintenant que nous disposons d'un code qui permet de résoudre le problème de propagation d'ondes Wi-Fi dans un appartement, nous pouvons le tester.

Il n'y a ici plus d'exercice à proprement parler : à vous d'effectuer les simulations qui vous paraissent intéressantes et de sauvegarder les résultats issus de ces résolutions en vue de les présenter à l'oral. La qualité des réponses et le nombre de questions traitées seront pris en compte. Soyez rigoureux/rigoureuses et précis(es) : notez scrupuleusement les paramètres utilisés, ne vous embrouillez pas entre les différentes simulations, sauvegardez les fichiers images et les fichiers de données, …

Nous donnons ici quelques idées de questions à se poser :

— Si le routeur est situé obligatoirement dans le salon et le long du mur « gauche », peut-on lui trouver une place pour pouvoir toujours être connecté au Wi-Fi tout en étant aux toilettes ? (nous pouvons par exemple supposer qu'en deça de 25% de la puissance max, la réception n'est pas suffisante pour glander sur youtube).
— À défaut de trouver l'optimal, quel serait un bon emplacement pour le routeur, pour obtenir du réseau Wi-Fi partout ?
— Sachant que l'indice n du béton est de l'ordre de 7. Que se passe-t-il si un des murs de l'appartement est en béton (mur porteur) ? Pouvons-nous obtenir une cartographie des différences entre la solution avec mur en placo et avec un mur porteur ?
— Influence du pas de maillage sur la solution ?

— Que se passe-t-il si les murs deviennent comme de l'air ($n(\Omega_{\text{mur}}) = 1$)) ? Pouvons nous obtenir une cartographie des différences entre la solution avec mur en placo et mur « en air » ?
— Et si les murs étaient en béton ($n = 7$) ?
— Et si c'était **votre** appartement ?
— …

Réfléchissez également à d'autres questions plus théoriques et essayez d'apporter des réponses, comme par exemple

— Quelle sont les limites du modèle ?
— Que et comment pourrions nous améliorer notre modèle et notre code si nous disposions d'une machine de puissance infinie ?
— Quelles améliorations, du point de vue géométrique, pouvons-nous apporter à notre appartement ?

Plus que l'obtention de réponse « parfaite », ce sera votre capacité et votre volonté de recherche qui seront pris en compte.

8.5 Résultat

8.5.1 Exemple

La figure 3 illustre un exemple de ce que l'on peut obtenir.

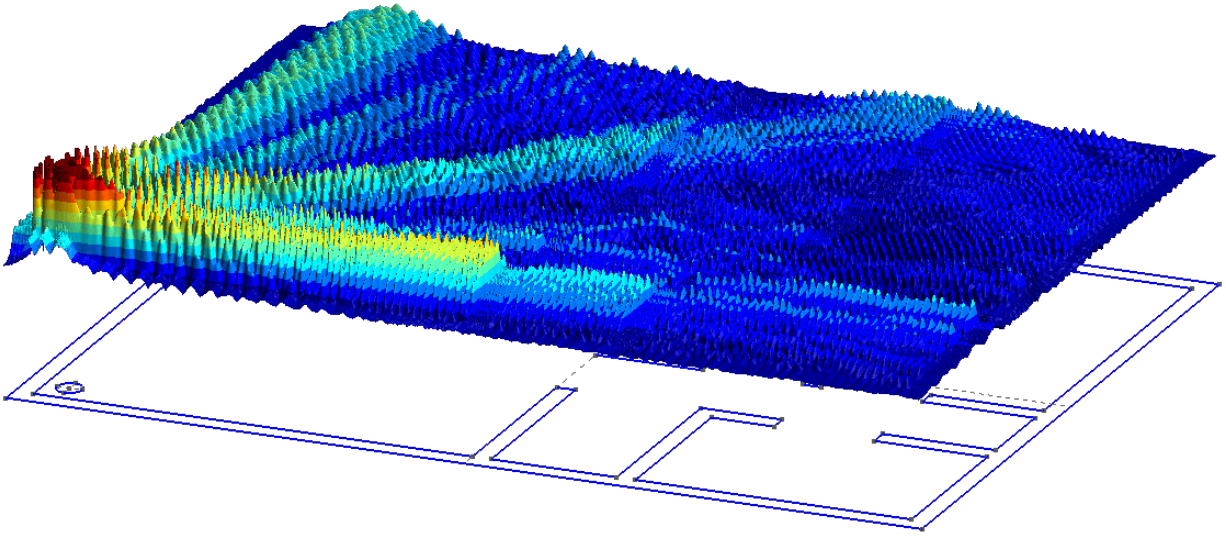

Fig. 3 – Propagation d'une onde Wi-Fi dans un appartement. Après avoir traversé 2 murs, l'onde Wi-Fi semble très amortie. Sous le résultat est affiché le plan de l'appartement et la position du routeur (petit disque à gauche)

8.5.2 Vous voulez tester ?

— Téléchargez le bundle Onelab. Il contient GMSH et GetDP (un solveur éléments finis)
— Téléchargez le code, soit directement soit via *Git* :

```
git clone https://github.com/Bertbk/wifi.git wifi
```

— Dans le dossier et dans un terminal, lancer

```
gmsh wifi.pro
```

— Vous pouvez modifier un peu la géométrie et la fréquence de l'onde, mise à 1GHZ. Attention, cette simulation est très gourmande : testez d'aborg avec 1GHz avant de lancer la simulation pour 2.5GHz (au risque de faire crasher votre ordinateur) !

CHAPITRE 9

2020 - 2021

9.1 Problème

Soit le domaine suivant modélisant un appartement :

Les murs sont supposés parfaitement isolant (nous pouvons toujours rêver !) , ce qui explique la condition de Neumann homogène que nous imposons. Nous remarquons que le bord de Ω est séparé en plusieurs parties : les radiateurs (Γ_{Rad}), les fenêtres (Γ_{Fen}) et les murs (Γ_{Mur}). Nous cherchons à calculer la température u dans la pièce, qui vérifie le système suivant

$$\left\{\begin{array}{rcll} -\Delta u & = & 0 & (\Omega) \\ u & = & T_c & (\Gamma_{\text{Rad}}) \\ u & = & T_f & (\Gamma_{\text{Fen}}) \\ \partial_{\mathbf{n}} u & = & 0 & (\Gamma_{\text{Mur}}) \end{array}\right. \tag{1}$$

Les paramètres sont les suivants :

- La longueur $L \simeq 10$
- la largeur $\ell \simeq 10$
- l'épaisseur des murs $d \simeq 0.5$
- la longueur d'une fenêtre est d'environ 1
- la longueur d'un radiateur est d'environ 1
- Les températures $T_c = 25$ et $T_f = -10$ sont les températures respectivement des radiateurs (ça chauffe) et de dehors (ça caille)

Tous ces paramètres peuvent être librement modifiés par vous même et les paramètres de la géométrie doivent même être choisis par vous !

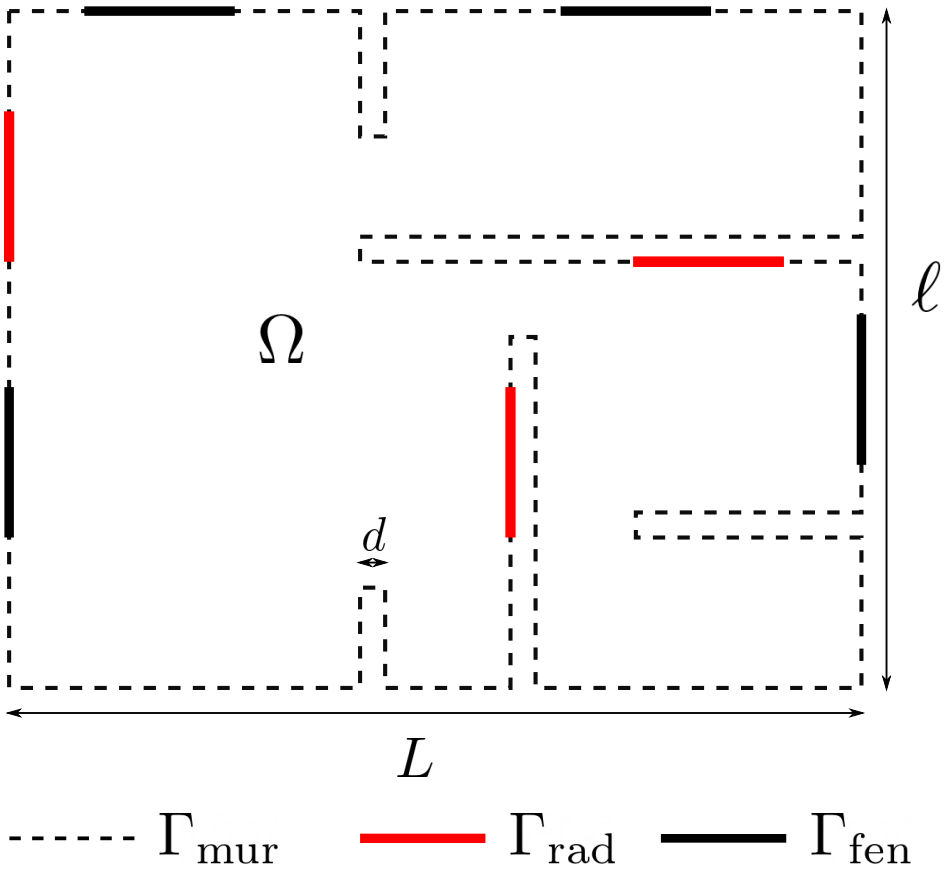

FIG. 1 – Domaine Ω et ses bords : un petit appartement tout mignon

9.2 Travail demandé

Nous souhaitons résoudre ce problème à l'aide de la méthode des éléments finis $\mathbb{P}^1-$ Lagrange.

1. **Construisez** la géométrie. Il ne s'agit pas de reproduire exactement l'appartement décrit plus haut mais de construire un appartement : libre à vous d'ajouter des pièces, fenêtres, des radiateurs ou un poster de Justin Bieber.
2. **Programmez** un code éléments finis P1 qui résolve le problème (1)

9.3 Consignes

1. Vous **pouvez** réaliser ce **projet en binôme**
2. Vous **devez** rendre ce projet sous la forme d'un dépôt `git` : envoyez moi **l'URL du dépôt uniquement**, **pas de fichier zip** !
3. Date limite de rendu : **28 février à 23h59**. Pas de blague, **tout projet rendu après le 28 février ne sera pas noté**.

En outre, **votre dépôt doit contenir** :

1. Un **script** qui résout le problème (1) et **affiche** la solution.
2. Un court fichier **README.md** facilitant sa compréhension, répondant notamment aux questions : « comment lance-t-on vos programmes ? » et « que doit-on obtenir ? » (exemple : « *Exécutez "main.py" et vous devez obtenir la même image que "solution.png" qui résout le problème méga compliqué* »)
3. Ajoutez à votre dépôt une **image** de la solution que vous avez obtenue (en PNG (pas très grosse svp !) et surtout pas en JPG). Vous pouvez même afficher l'image dans le fichier `README.md`.

Remark 2.19

Pour faire simple, je dois pouvoir télécharger votre dépôt, lancer un fichier, et voir la solution, le tout sans avoir à réfléchir de mon côté :-)

Quatrième partie

Download

Proof Index

proposition-5

proposition-6

remark-0

remark-1

remark-10

remark-2

remark-3

remark-4

remark-5

remark-6

remark-7

remark-8

remark-9

theorem-0

theorem-1

theorem-10

theorem-4

theorem-7