Projet 2 : Questions préliminaires

L'objectif du projet est de résoudre

$$-\nabla \cdot (\mathbf{M}\nabla u) = f \text{ dans } \Omega \qquad u|_\Gamma = u_\Gamma$$

avec les données suivantes

- Ω un ouvert borné de $\mathbb{R}^2$ de frontière Γ qu'on approchera par une triangulation générée par `freefem++`.
- $\mathbf{M}$ un tenseur 2×2 dont les éléments $M_{i,j}$ sont dans C (les nombres complexes) et tels que $\bar{w}^T\mathbf{M}w \in \mathbb{R} \ \ \forall w \in \mathrm{C}^2$ où $\bar{w}$ désigne le complexe conjugué de w.
- f, g des applications de Ω dans C.

Le probléme sera résolu par une méthode de gradient conjugué sur la version discrétisée en éléments finis P^1 triangulaires de

$$\min_{v\in H^1(\Omega;C)} \int_\Omega [\frac{1}{2}\vec{\nabla}\bar{v}^T\mathbf{M}\vec{\nabla}v - \mathbb{R}e(fv)] < +\frac{1}{2\epsilon}\int_\Gamma (v-g)(\bar{v}-\bar{u}_\Gamma)$$

où ϵ est un paramètre de pénalisation.

Q1 : Utilisation de FreeFem++

1. Utiliser le script FreeFem++ *cercle.edp* pour générer un maillage du cercle avec différents n. Suivant ce modèle écrire un script FreeFem++ pour générer le maillage du carré $[0, L] \times [0, L]$, avec $L = 1$.
2. Construire une triangulation avec `FreeFem++` d'un carré unité avec un trou elliptique (les trous sont obtenus en changeant le sens de parcours de la frontière).
3. Changer le sens de parcours de l'ellipse et l'utiliser pour contrôler le raffinement du maillage dans cette région.
4. Raffiner le maillage du carré dans sa partie basse en introduisant une frontière interne horizontale de longueur 1 et à une hauteur de 0.05. (Il faut couper les côtés verticaux en deux segments pour assurer la jonction exacte avec la frontière interne).

Q2 : Résolution de l'équation de la chaleur 2D

1. Le modifier le programme `sfemMatPleine.cpp` qui résoud le problème :

$$-\Delta u = -6 \quad \text{sur} \quad \Omega, \quad \text{et} \quad u|_\Gamma = g, \tag{1}$$

avec

$$g(x, y) = x^2 + 2y^2; \tag{2}$$

$$f(x,y) = -6. \tag{3}$$

Afin de supprimer la matrice pleine en construisant la classe suivante qui modéliser la matrice A du "Laplacien" sans la construire ni la stocker.

```
class MatLap2d: VirtualMatrice<R> {
                                //    modelisation de la matrice du Laplacien
  public:
  const  Mesh & Th;
  typedef  VirtualMatrice<R>::plusAx plusAx;
  MatLap2d(const  Mesh & T) : Th(T) {};
   void addMatMul(const  KN_<R>  & x, KN_<R> & y) const;  //    y +=A*x
    plusAx operator*(const KN<R> &  x) const {return plusAx(this,x);}
 };
```

Où la méthode `MatLap2d : :addMatMul` calcule :

pour toutes les triangles K de `Th` et pour $i = 0, 1, 2$ faire :

$$y[i_K] += \int_K \nabla\omega^{i_K} \sum_{j=0}^{3} x[j_K] \nabla\omega^{j_K}$$

où i_K (resp. i_K) est le numéro global du sommet i (resp. j) du triangle K.

Avec le maillage du carré construit à la question Q1, exécuter le programme, et représenter graphiquement la solution (graph 3D et iso-lignes).

Calculer la solution exacte et comparer avec la solution numérique.

2. Modifier le programme précédent afin de résoudre le problème

$$-\Delta u + u = f \quad \text{sur} \quad \Omega, \quad \text{et} \quad u|_\Gamma = g. \tag{4}$$

avec $\Omega = [0, L] \times [0, L]]$, $\Gamma = \partial\Omega$.

Choisir f et g afin d'avoir la même solution exacte que précédemment.