Projet 2 - première partie
Utilisation de `FreeFem++` et des classes `sfem` pour la méthode des éléments finis P^1

Q1 : Construction d'une triangulation avec FreeFem++

1. Le script *cercle.edp* contient des instructions qui permettent au logiciel `FreeFem++` de générer le maillage d'un cercle, avec n points de discrétisation distribués sur la frontière. Tester le script[1] pour différents n.
   Analyser la structure du fichier *cercle.msh* qui contient la triangulation générée avec ce script.

2. Suivant le modèle précédent, écrire un script FreeFem++ pour générer le maillage du rectangle $[0, L_1] \times [0, L_2]$, en optimisant la forme des triangles du maillage (le rapport L/n doit être le même pour chaque morceau de frontière de longueur L sur lequel on distribue n points de discrétisation).

3. 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).

4. Changer le sens de parcours de l'ellipse et l'utiliser pour contrôler le raffinement du maillage dans cette région.

Q2 : Résolution d'une EDP avec FreeFem++

Le script *cercle_lap.edp* résout l'EDP suivante

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

avec le choix particulier :

$$g(x, y) = x^2 + 2y^2, \qquad f(x, y) = -6, \tag{2}$$

et le domaine Ω le cercle de rayon R, centré en origine.

1. Tester le script et analyser la syntaxe utilisée pour résoudre la formulation variationnelle. Rajouter dans le script la visualisation de la solution exacte pour ce problème.

2. Modifier le script précédent afin de résoudre le problème

   $$-\Delta u + u = f \quad \text{sur} \quad \Omega, \quad \text{et} \quad u|_\Gamma = g. \tag{3}$$

   avec $\Omega = [0, L] \times [0, L]$, $\Gamma = \partial\Omega$. Choisir f et g afin d'avoir la même solution exacte que précédemment.

Q3 : Utilisation des classes `sfem`

Le programme `sfemMatPleine.cpp` résout le problème (1) en utilisant les classes `sfem`, spécialement conçues pour la méthode des éléments finis. Il utilise également les classes `RNM` qui facilitent l'utilisation des tableaux (matrices).

1. Analyser (voir plus bas quelques indications sur la formulation mathématique du problème) et tester le programme. Comparer avec la solution trouvée avec `FreeFem++` (Inclure dans le programme la fonction `gnuplot-iso` qui permet de tracer les iso-lignes avec Gnuplot.)

[1] Pour télécharger et installer `FreeFem++` sur votre machine (sous Windows ou Linux), consulter la page www.freefem.org.

2. Pour utiliser de manière efficace l'algorithme du gradient conjugué, nous éviterons de calculer la matrice du système linéaire en évaluant directement les produits matrice-vecteur qui interviennent dans l'algorithme.

Pour cela, nous aurons besoin de matrices *virtuelles* (qui ne sont pas connues explicitement mais pour lesquelles on sait calculer le produit avec un vecteur donné). L'exemple suivant montre comment définir une matrice virtuelle – il s'agit de la matrice qui est utilisée comme préconditionneur dans la méthode du gradient conjugué.

```
template <class R>
class MatDiag: VirtualMatrice<R> { public:
  const KN_<R> d; // Adr du Vecteur diagonal
  typedef typename  VirtualMatrice<R>::plusAx plusAx;

  MatDiag(KN_<R> dd) :d(dd) {}

  void addMatMul(const  KN_<R>  & x, KN_<R> & Ax) const {
    assert(x.N()==Ax.N() && x.N() == d.N() );
    Ax+=DotStar_KN_<R>(x,d);   // <=>  for(int i=0;i<x.N;i++) Ax[i] += x[i]*d[i];
  }

  plusAx operator*(const KN<R> &  x) const {
    return plusAx(this,x);}
};
```

Il faut retenir que la partie la plus importante est la définition de l'opérateur `addMatMul` qui va calculer en fait le produit matrice-virtuelle.

3. D'après le modèle précédent, définir une matrice virtuelle `Matrice_lap` correspondant au problème (1). Seulement le produit matrice-vecteur (correspondant à la formulation variationnelle du problème) doit être programmé dans l'opérateur `addMatMul`.

Création d'une matrice virtuelle `Mat_lap` qui va calculer le produit matrice vecteur sans construire ni stocker la matrice du laplacien.

L'idée est de décrire un un tous les triangles de notre maillage afin de rajouter à chaque sommet, sa contribution au gradient de temprature.

Indications

La formulation variationnelle du problème (1) est : trouver $u \in H_0^1(\Omega)$ tel que

$$\int_\Omega \nabla u \nabla v = \int_\Omega fv; \quad \forall v \in H_0^1(\Omega), \tag{4}$$

La solution approchée du problème sera (on considère tous les sommets de la triangulation $\mathcal{T}_h$)

$$u = \sum_{i \in \mathcal{S}_{\mathcal{T}_h}} u_i w^i. \tag{5}$$

En utilisant cette décomposition dans la formulation variationnelle on obtient le système linéaire suivant :

$$A[u_i] = M[f_i], \tag{6}$$

avec les vecteurs $[u_i]$ et $[f_i]$ $(i = 1, \dots n_v)$ et les matrices de taille $(n_v \times n_v)$ $M = m_{ij}$, $A = a_{ij}$ définies par :

$$m_{ij} = \int_{\Omega_h} w^i w^j, \qquad a_{ij} = \int_{\Omega_h} \nabla w^i . \nabla w^j. \tag{7}$$

Les conditions aux limites de Dirichlet seront renforcées en imposant :

$$\text{si } i \in \Gamma \Longrightarrow \quad a_{ii} = tgv, \quad f_i = tgv * g(i). \tag{8}$$

Cette technique nous assure qu'après la résolution du système linéiaire, nous obtenons bien $u_i = g(i)$ pour tout point i appartenant à la frontière Γ.

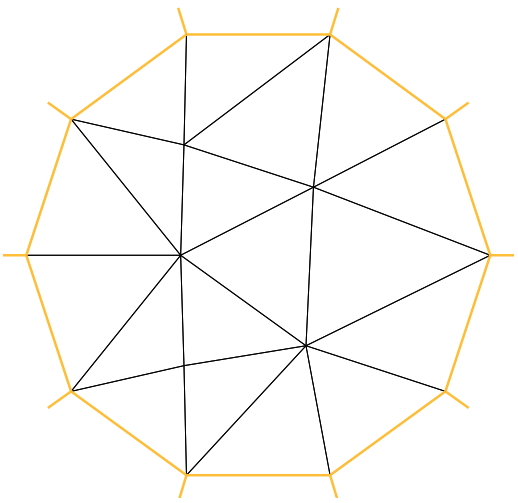

FIG. 1 – Utilisation de freefem++ pour construire le maillage d'un cercle pour lequel le nomrbre n de points du maillage situs sur la frontire est 10.

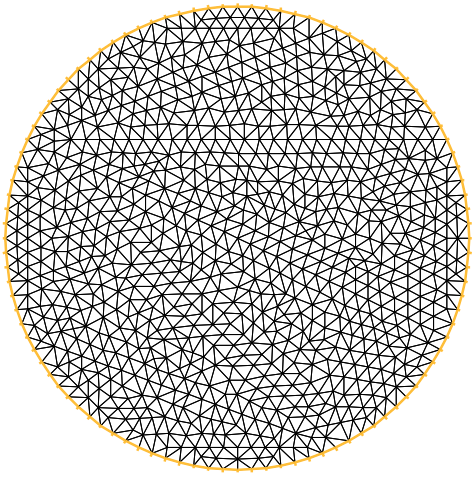

FIG. 2 – idem avec n=100

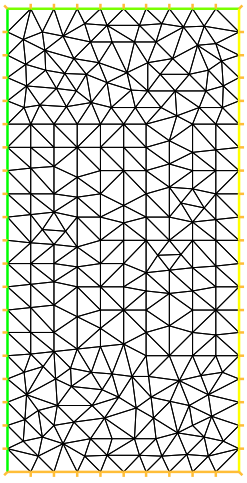

FIG. 3 – maillage uniforme d’un rectangle dont la largeur vaut 1 et la longueur vaut 2

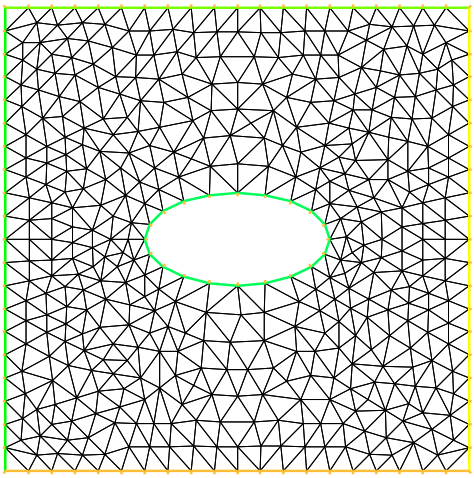

FIG. 4 – maillage d’un carre possédant un trou elliptique

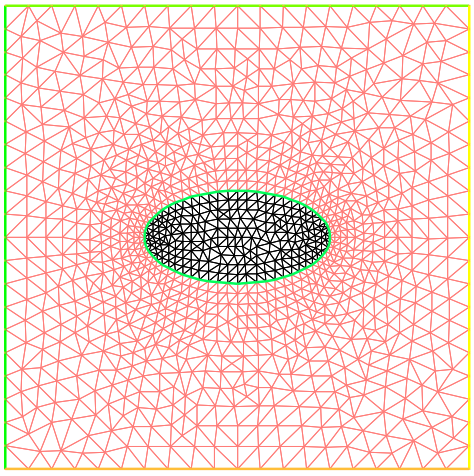

FIG. 5 – maillage d'un carre possédant un trou elliptique l'intérieur duquel le maillage a été raffiné

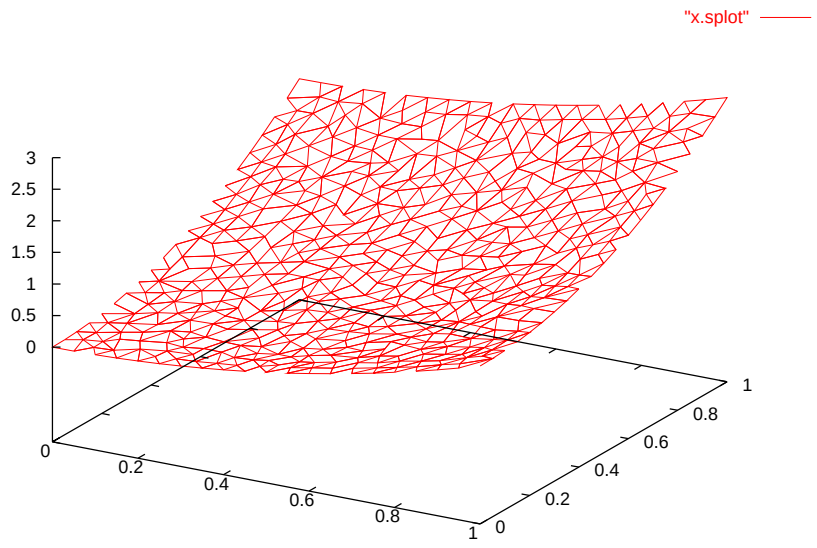

FIG. 6 – solution de l'equation $-\Delta u = f$ et $u|_\Gamma = g$ avec le choix particulier : $g(x,y) = x^2 + 2y^2$, $f(x,y) = -6$

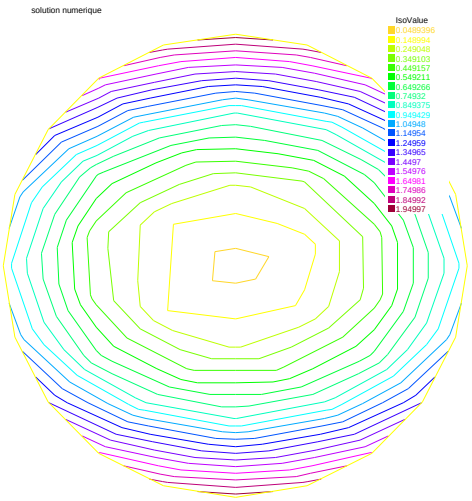

FIG. 7 – solution numerique de l'equation $-\Delta u = f$ et $u|_\Gamma = g$ avec le choix particulier : $g(x,y) = x^2 + 2y^2$, $f(x,y) = -6$ affichage des isolignes

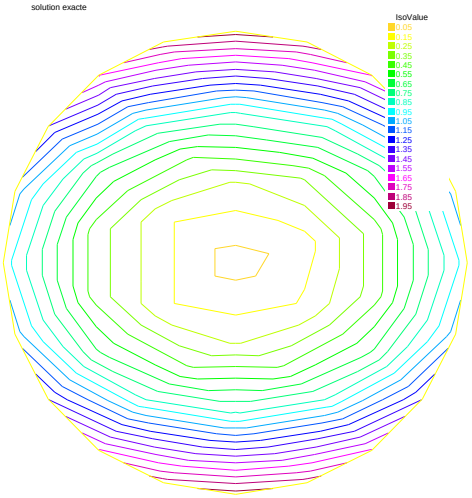

FIG. 8 – solution exacte de l'equation $-\Delta u = f$ et $u|_\Gamma = g$ avec le choix particulier : $g(x, y) = x^2 + 2y^2$, $f(x, y) = -6$ affichage des isolignes

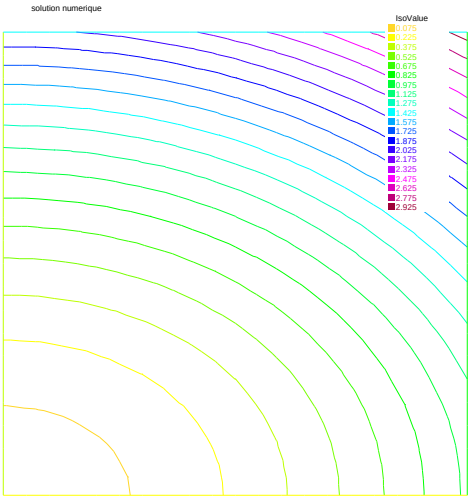

FIG. 9 – solution exacte de l'equation $-\Delta u = f$ et $u|_\Gamma = g$ avec le choix particulier : $g(x, y) = x^2 + 2y^2$, $f(x, y) = -6$ affichage des isolignes

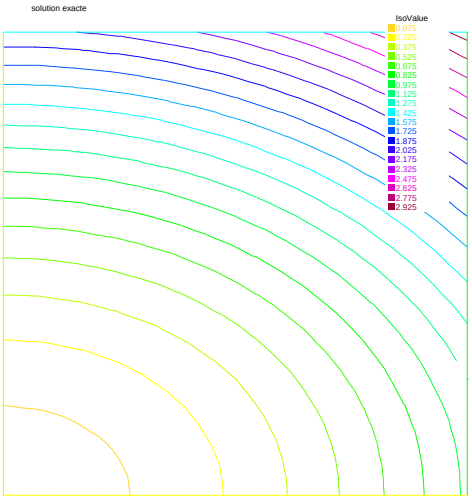

FIG. 10 – solution exacte de l'equation $-\Delta u = f$ et $u|_\Gamma = g$ avec le choix particulier : $g(x, y) = x^2 + 2y^2$, $f(x, y) = -6$ affichage des isolignes

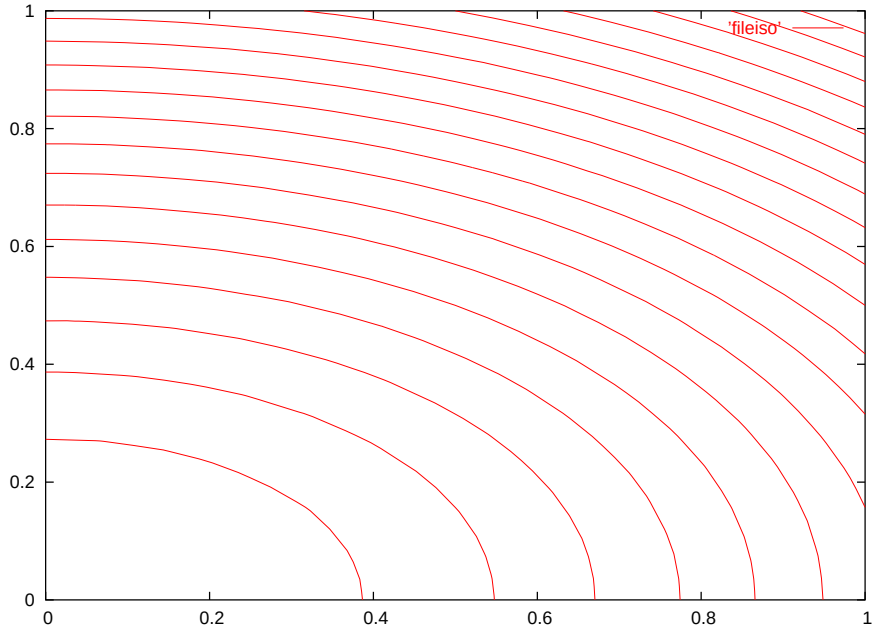

FIG. 11 – solution numérique de l'equation calcule via un programme c++ $-\Delta u = f$ et $u|_\Gamma = g$ avec le choix particulier : $g(x, y) = x^2 + 2y^2$, $f(x, y) = -6$ affichage des isolignes