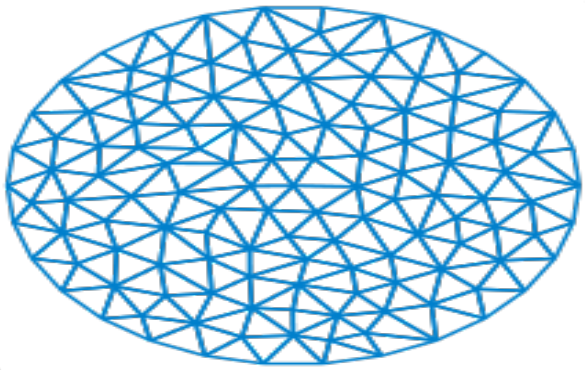
Rapport du projet de Volumes finis :
4A. Mathématiques appliquées et Modélisation 2017-2018

Volumes Finis :
Résolution de l'équation de Laplace 2d : Partie VF4



Groupe de travail : Groupe II

École Polytechnique Lyon
15 Boulevard Latarjet 69622 Villeurbanne

Table des matières

1 Introduction

Définition 1 *Élément fini :*
un triplet (K,P_k, σ_k) où K est l'une des formes géométriques présentées (convexe en général), σ_k est l'ensemble des degrés de liberté, P_k est l'espace des fonctions tel que l'ensemble σ_k soit ***unisolvant****.*

Définition 2 *Unisolvance :*
Soit un ensemble σ_k, cet ensemble est dit unisolvant s'il existe une unique fonction de P_k qui prend les valeurs données aux degrés de liberté de σ_k.

En analyse numérique, la méthode des volumes finis est utilisée pour résoudre numériquement des équations aux dérivées partielles, au même titre que la méthode des éléments finis. Ces deux méthodes exploitent toutes deux des approximations d'intégrales. Toutefois, la méthode des volumes finis se base directement sur la forme dite forte de l'équation à résoudre :

Soit $f \in L^2(\Omega), \Omega$ un ouvert de $\mathbb{R}^n$, on introduit le problème suivant :

$$\begin{cases} u \in H^2(\Omega) \\ -\Delta u = f \in \Omega \\ u = 0 \text{ dans } \partial\Omega \end{cases}$$

La méthode des éléments finis se fonde sur une formulation variationnelle de l'équation [1].

$$\begin{cases} u \in H^1(\Omega) \\ A(u,v) = F(v) \ \ \forall v \in H^1(\Omega) \ \mid \ v = 0 \text{ sur } \partial\Omega \\ u = 0 \text{ dans } \partial\Omega \end{cases}$$

Où $A(u,v) = \int_\Omega \nabla u.\nabla v$ *et* $F(v) = \int_\Omega fv$ Une équation aux dérivées partielles (EDP) peut être résolue de manière approchée à l'aide d'un maillage (discrétisation spatiale d'un milieu continu) constitué de volumes finis qui sont des petits volumes disjoints en 3D, des surfaces en 2D, des segments en 1D. **La réunion de ces volumes finis constitue le domaine d'étude**.
Ces équations aux dérivées partielles contiennent des termes de divergence. En utilisant le théorème de *Green-Ostrogradski*, les intégrales de volume d'un terme de divergence sont transformées en intégrales de surface et ces termes de flux sont ensuite évalués aux frontières de chaque volumes finis. On utilise une fonction de flux numérique pour élaborer une approximation des flux aux frontières. Puisque le flux entrant dans un volume donné est égal au flux sortant du volume adjacent, ces méthodes sont conservatives, donc parfaitement adaptées à la résolution de lois de conservation.
La méthode des volumes finis est facilement utilisable avec des maillages non structurés car, en matière de discrétisation des lois de conservation, sa formulation ne tient aucun compte de la complexité du maillage.
La méthode des volumes finis trouve son utilité dans de multiples domaines comme dans les modélisations de trafic routier, en mécanique des fluides, et plus généralement dans tout problème comportant des équations aux dérivées partielles.

1. on parle aussi de formulation faible

1.1 Sujet du projet :

Le projet suivant s'inscrit dans la partie pratique de la matière Volume Fini est consiste à comprendre et faire la documentation d'un code en Scilab réalisé par Stella Krel et F. Boyer et disponible sur la page web de cette dernière [**?**]. Ce code permet de résoudre le problème elliptique en 2d par le biais appelé "**équation de Laplace**", par une méthode de volumes finis :

$$\begin{cases} -\Delta u(x) = f(x), \forall x \in \Omega \\ u(x) = g(u(x), x), \forall x \in \partial\Omega \end{cases}$$

Avec

- Ω un ouvert de $\mathbb{R}^2$
- $f \in L^2(\omega)$
- $f \in L^2(\partial\omega)$

Les équations de ce type peuvent être résolue par un schéma VF4 qui présente plusieurs propriétés telles que

- Conservation locale de la masse et consistance des flux.
- Préservation des propriétés qualitatives des E.D.P telle que l'existence et l'unicité.
- Préservation des bornes physiques de la solution.
- Bonne précision, même au cas des maillages grossiers [2] et avec de fortes anisotropies et hétérogénéités.
- Implémentation facile" et complexité raisonnable

Durant notre projet nous allons essayer de visualiser cela.

1.2 Organisation des taches :

Abouabdallah Mohamed Anwar Maillage remplissez-la comme vous le souhaiter ! ! !

2 Jeu de données

2.1 Présentation :

Afin de tester le schéma VF4, plusieurs jeux de données ont été implémentés dans le code Scilab. Ces jeux de données correspondent à des problèmes physiques variés, comme par exemple l'écoulement vertical d'un liquide. Un fichier (*donnees.sci*) est uniquement dédié à la définition de ces jeux de données, tandis que les fichiers *.sce* feront office d'exécutables.
Un jeu de données est composé de plusieurs éléments. Un jeu de donnée est en fait un cas, qui va permettre de tester le schéma. Pour chaque jeu de donnée, on aura les informations suivantes :

2. voir partie raffinement

- Un **nom** : chaque jeu de données est caractérisé par un nom. Il s'agira le plus souvent de u la solution exacte de l'équation.
- L'expression du **terme source** : Il s'agit de la fonction f dans la formulation forte du problème.
- La **solution exacte** : Cette fonction permettra de comparer la solution exacte avec les résultats obtenus avec la méthode des volumes finis. Cependant, la solution exacte n'existe pas pour tous les cas tests.
- Un coefficient **k** : Il s'agit d'une fonction qui permet de caractériser le milieu physique du problème. Cette fonction se base sur le coefficient de diffusion du milieu.
- Un **tenseur** : Ce tenseur est défini par 3 fonctions.
- Les **conditions de Dirichlet** : Cette fonction permet de spécifier les conditions aux limites de Dirichlet.
- La **méthode** : Ce paramètre permet de choisir une méthode qui pour le calcul du coefficient k ou du tenseur.
- L'ensemble des **maillages** qui peuvent être utilisés pour ce cas test. Il s'agit d'une liste, définie dans le fichier *maillage.sci*.

Tous les jeux de données ne contiennent pas l'ensemble des éléments. Cela va dépendre des cas. Chaque cas de test est une structure de donnée personnalisée. Tous les cas tests sont stockés dans une liste appelée "cas_test". Cette liste est donc une liste de structure de donnée.

2.2 Données utilisées dans le projet :

Dans le code on crée une structure de donnée qui permet de caractériser un cas test de manière unique, à l'aide des éléments suivants :
Une première partie est consacrée à la définition de la fonction, ici on donne comme exemple la définition de celle du premier cas test :

$$u(x,y) = sin(pi*x)sin(pi*y)$$

```
function [z]=f_nulle(x,y)
z=0;
endfunction

function [z]=coeff_cte(x,y)
z=1;
endfunction

cas_test=list();

function [z]=f1(x,y)
z=2*%pi^2*sin(%pi*x)*sin(%pi*y);
endfunction

function [z]=ue1(x,y)
z=sin(%pi*x)*sin(%pi*y);
endfunction
```

Ensuite on enregistre notre fonction dans une structure [3] afin de pouvoir les réutiliser par la suite. Cette structure est défininie de manière unique pour chaque cas test.

```
cas_test($+1)=struct("nom","u(x,y)=sin(pi*x)sin(pi*y)",...
"coeff_k",coeff_cte,...
"uexacte",ue1,...
"source",f1,...
"bordD",ue1,...
"maillages",maillages_carre_Dir);
```

2.3 Résultats obtenus :

Dans les sous sections qui vont suivre nous nous focaliserons sur les différents cas test que nous allons traité durant ce projet. Pour cela, on va choisir arbitrairement des maillages triangulaires et un niveau de raffinement de 4. Ensuite, nous allons nous concentrer sur la différence entre la solution exacte et la solution rapprochée.

2.3.1 Produit de sinus pour le Laplacien

Dans les sous sections qui suivront on reprendra les éléments dans le commentaire code de Stella Krell : La fonction décrite dans ce cas test est la fonction

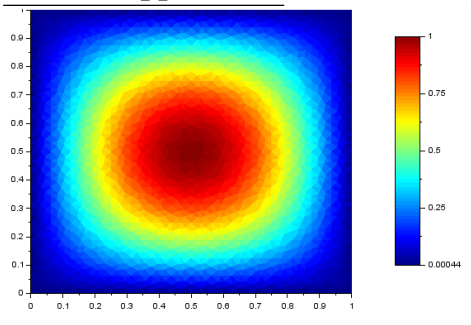
$$u(x,y) = sin(pi*x)sin(pi*y)$$

Analyse des sorties :

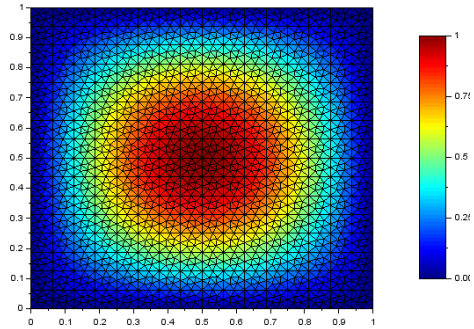
Utilité :	Variable
Solution exacte :	$ue1(x,y) = sin(\pi x)sin(\pi y)$
Fonction qui décrit le terme source :	$f_1(x,y) = 2\pi^2 \sin(\pi x)\sin(\pi y)$
Coefficient de diffusion isotrope :	1
Dirichlet au bord :	$ue1(x,y) = sin(\pi x)sin(\pi y)$

Après la présentation des fichiers nous allons maintenant comparer entre les solutions exacte et approchée :

Solution approchée :



Solution exacte :



3. Ici on rentre pas dans les détails car la fonction struct sera reprise dans la section maillage.

Comme nous l'avons annoncé dans l'introduction, le schéma V.F.4 permet d'avoir une bonne précision, ce qui se manifeste dans notre cas.

2.3.2 Polynôme pour le laplacien

La fonction décrite dans ce cas test est la fonction

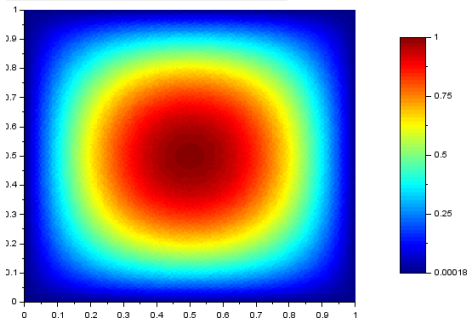
$$u(x,y) = 16x(1-x)y(1-y)$$

Analyse des sorties :

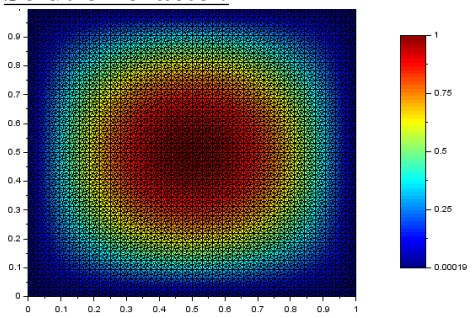
Utilité :	Variable
Solution exacte :	$ue2(x,y) = 16x(1-x)y(1-y)$
Fonction qui décrit le terme source :	$f_2(x,y) = 32y(1-y) + 32x(1-x)$
Coefficient de diffusion isotrope :	1
Dirichlet au bord :	$ue2(x,y) = 16x(1-x)y(1-y)$

Après la présentation des fichiers nous allons maintenant comparer entre les solutions exacte et approchée :

Solution approchée :



Solution exacte :



2.3.3 Solution du laplacien non nulle au bord

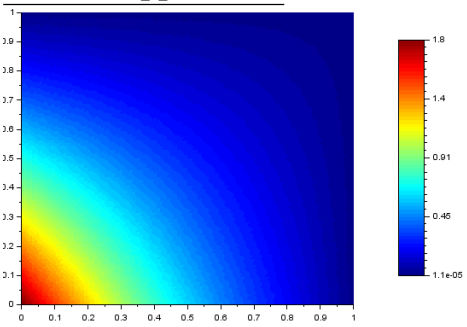
La fonction décrite dans ce cas test est la fonction

Analyse des sorties :

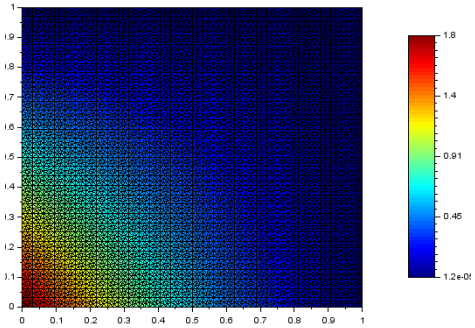
Utilité :	Variable
Solution exacte :	$ue3(x,y) = \sin((1-x)(1-y)) + (1-x)^3(1-y)^2$
Fonction qui décrit le terme source :	$f_3(x,y) = (1-y)^2\sin((1-x)(1-y)) - 6(1-x)(1-y)^2 + (1-x)^2\sin((1-x$
Coefficient de diffusion isotrope :	1
Dirichlet au bord :	$ue3(x,y) = \sin((1-x)(1-y)) + (1-x)^3(1-y)^2$

Après la présentation des fichiers nous allons maintenant comparer entre les solutions exacte et approchée :

Solution approchée :



Solution exacte :

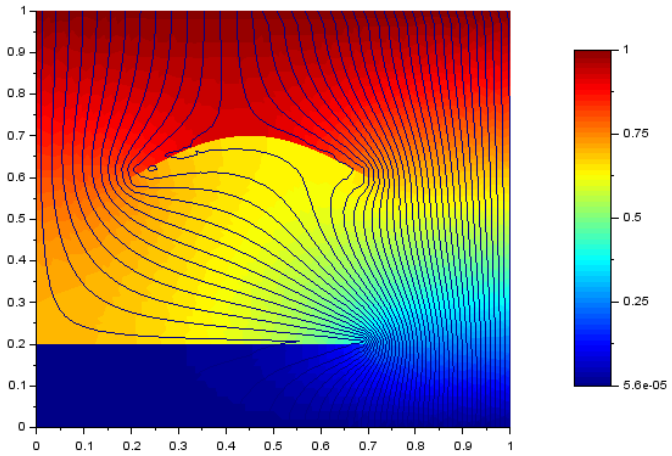


2.3.4 Écoulement vertical dans un milieu poreux fracture

Analyse des sorties :

Notation	Utilité
Nom :	
Fonction qui décrit le terme source :	0
Coefficient de diffusion isotrope :	Ici on a pas de sol exacte.
Dirichlet au bord :	$ubord2(x,y) = y$
Neumann au bord :	$\frac{\partial u}{\partial n} = 0$
Coefficient qui intervient dans le modèle de fractures :	0
Nombre de fonctions de courant à afficher :	40

Solution approchée :



Ici on a choisit un Maillage de triangle du domaine avec deux fractures.

2.3.5 Polynomiale isotrope discontinue

La fonction décrite dans ce cas test est la fonction

Analyse des sorties :

Fonction qui décrit le terme source :

$$f_5(x,y) = \frac{-10^{-3}2x-(10^{-3}x^2+y^2)}{x^2+y^2+10^{-16}} + \frac{2(10^{-3}x^2+y^2)x}{(x^2+y^2+10^{-16})^2} - \frac{(10^{-3}-1)x+(10^{-3}-1)xy}{x^2+y^2+10^{-16}} + \frac{2(10^{-3}-1)xyy}{(x^2+y^2+10^{-16})^2}$$

Utilité :	Variable
Solution exacte :	$ue1(x,y) = sin(\pi x)sin(\pi y)$
Coefficient de diffusion isotrope :	1
Dirichlet au bord :	$ue1(x,y) = sin(\pi x)sin(\pi y)$

Tenseur de diffusion : le coefficient de diffusion est ici matriciel

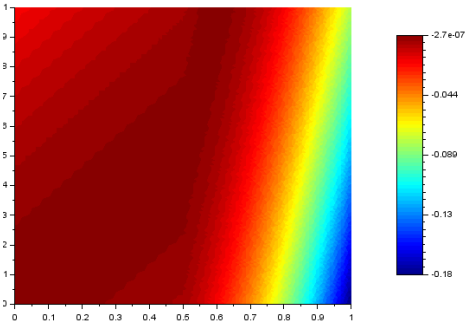
$$mat\_xx\_cont(x,y) = \frac{10^{-3}x^2+y^2}{x^2+y^2}$$

$$mat\_yy\_cont(x,y) = \frac{x^2+10^{-3}y^2}{x^2+y^2}$$

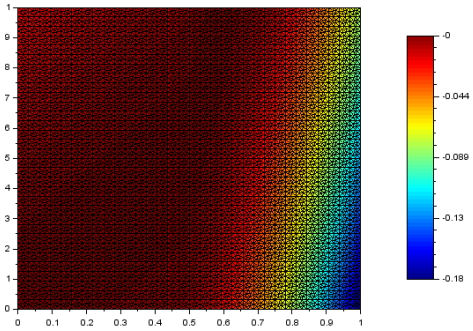
$$mat\_xy\_cont(x,y) = \frac{(10^{-3}-1)xy}{x^2+y^2}$$

Après la présentation des fichiers nous allons maintenant comparer entre les solutions exacte et approchée :

Solution approchée :



Solution exacte :



Les deux cas tests restants mobilisent des schémas DDFV, donc on ne vas pas les traité dans cette section.

3 Les Maillages

Au court de cette partie, nous allons nous focaliser sur les fichiers et fonctions relatives aux maillages : Ainsi nous allons étudier les maillages proposés, ce qui nous conduit à se pencher sur leurs structures et le niveau de raffinement .

3.1 Définitions :

Pour cela, nous allons définir les différents maillages avec lesquels on va travailler dans ce projet ainsi que leurs propriétés. Pour cela on se basera sur la partie théorique faite par Stella Krell.

Définition 3 *Maillage admissible : Soit Ω, un fermé borné de $\mathbb{R}^2$. Un maillage τ admissible de ω au sens des volumes finis est donné par :*

- *un ensemble M d'ouverts polygonaux convexes disjoints 2 à 2, appelés volumes de contrôle T_i, tels que . On note l'ensemble des bords de volume de contrôle de M inclus dans $\partial\omega$ qui sont considérés comme des volumes de contrôles dégénérés.*
- *Pour tous les volumes de contrôle voisins Ti et L, on suppose que $\partial T_i \cup \partial L$ est un côté de chaque volume de contrôle, et il est appelé arête σ du maillage τ, notée $\sigma = K|L$. On note E l'ensemble de ces arêtes.*
- *À chaque volume de contrôle $T_i \in M \cup \partial\Omega$, on associe un point $x_k \in K\tau$ On impose pour $\sigma = T_i|L$ que la ligne joignant xK à xL est orthogonale à l'arête $\sigma = K|L$ et pour $\sigma \in \partial T_i \cap T_i$ que la ligne joignant x_k à x_σ est orthogonale à l'arête σ.*

Définition 4 *Maillage primate : On choisit $N+1$ points $(x_{i+\frac{1}{2}})i \in [0;N]$tels que*

$$0 = x_{\frac{1}{2}} < x_{\frac{3}{2}} < ... < x_{N-\frac{1}{2}} < x_{N+\frac{1}{2}}$$

et qui ne sont pas nécessairement équidistants.
Les segments $T_i = [x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}]$ sont appelés cellules primales et sont de longueur $|T_i| = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$
Le pas du maillage primale est noté $h = \max_{i\in[1,N]} |Ti|$

Définition 5 *Maillage dual : A chaque T_i , on associe un point $x_i \in T_i$ qui n'est pas nécessairement le milieu de T_i. Suite à cela on pose $x_0 = 0$ et $x_{N+1} = 1$, et on appelle cellules duales les segments $D_{i+\frac{1}{2}} = [x_i, x_{i+1}]$ sont appelés cellules primales et sont de longueur $|D_{i+\frac{1}{2}}| = x_{i+1} - x_i$*

Définition 6 *Maillage diamant : Grâce aux mailles primales et duales, on définit les diamants D du maillage tels que leurs diagonales principales soient une arête primale et une duale*

3.2 Analyse de la structure :

La méthodes des volumes finis il faut décomposer l'ensemble Ω sur lequel on travaille en plusieurs éléments, c'est ce qu'on appelle faire un maillage du domaine.
Un maillage est défini par :

- Son repère
- Les points le constituant
- Les cellules, constituant des polygones reliant n de ces points
- **Les arrêtes des polygones :** qui sont intérieures quand elles se trouvent entre deux éléments et extérieures quand elles sont au bord (donc ne concernent qu'un seul élément dans ce cas).
- **Sa dimension :** $1D$, $2D$ ou $3D$
- **Son volume :** C'est à dire la dimension totale couverte.
- **Son raffinement :** surface ou volume moyen des cellules composant le maillage
- **la géométrie des cellules en** $2D$ **:** triangles, quadrilatères, polygones... En $3D$: tétraèdres, prismes, hexaèdres (parallélépipèdes, cubes), polyèdres...
- **Les diamants :** Un dament est un quadrilatère formé par les deux sommets de l'arrête joignant deux cellules et leurs deux centres de gravité.
- **Le degré de l'élément :** c'est le degré du polynôme servant à décrire les côtés ou arêtes des éléments. Un élément de degré 1 aura des côtés ou arêtes rectilignes.

Dans l'idéal, le maillage doit aussi épouser les contours $\partial\Omega$ du domaine.

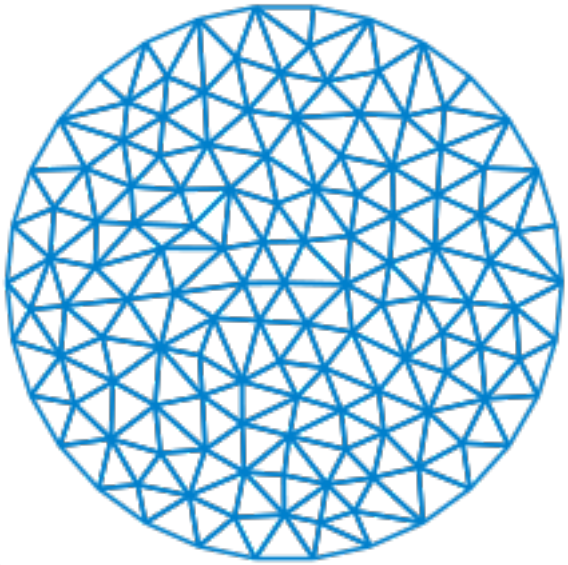
Alors, comment se fait la gestion des maillages dans notre code ?

3.3 Analyse Code

Dans le code on crée une structure de donnée qui permet de caractériser un maillage de manière unique, à l'aide des éléments suivants :

- *maillage.nom* : Le nom de la famille de maillages
- *indice_min* et *indice_max* : Les niveaux de raffinement extremums disponibles.
- *maillage.nbvol* : Le nombre de volumes de contrôle.
- *maillage.nbare* : Le nombre d'arêtes.
- *maillage.nbsom* : Le nombre de sommets.
- *maillage.centres* : Matrice qui aura le nombre de volumes en guise de nombre de lignes et 2 colonnes (une pour chaque dimension de l'espace). Cette matrice contient les coordonnées des centres des volumes de contrôle.
- *maillage.sommets* : Matrice qui aura le nombre de sommets en guise de nombre de lignes, et 2 colonnes. Cette matrice contient les coordonnées des centres des sommets.
- *maillage.aretes* : Matrice de taille nbsomx17 contenant des informations sur les arêtes diamants.

L'ensemble des maillages seront stockés dans une liste. Cette liste sera donc une liste de structures de données. Chaque élément de cette liste sera un maillage.
Dans un répertoire annexe, on trouvera des fichiers qui contiennent les tables de position des nœuds. Chacun de ces fichiers a été généré par un mailleur. La table de positions des nœuds est en deux parties. La première ligne indique le nombre de nœuds total. La suite du fichier contient simplement les coordonnées des nœuds, dans un repère orthonormé (à 2 dimensions ici).
Ces fichiers de maillage seront lus par une fonction appelées *lecture_maillage*. Cette fonction permettra, à partir d'un fichier généré par le mailleur, de le mettre sous la forme d'une structure de donnée définie ci-dessus.



3.3.1 Implémentation

L'implémentation des maillages se fait tel cela : Le code commence par charger un nouveau répertoire courant :"./maillage/" puis distingue deux types de données : " maillages d'un domaine fracture avec un écoulement vertical" (cas 4) et " maillages du carre unité avec CL de Dirichlet" (les données restantes). Ensuite, selon chaque cas, il va définir les maillages par rapport leur noms et leur indices max et min de raffinement grâce à la fonction struct [4]. Ensuite il va lire les maillages par la fonction suivante.

```
function [maillage]=lecture_maillage(nom)
    [sommets,text_som]=fscanfMat(nom+'_sommets');
    [centres,text_cen]=fscanfMat(nom+'_centres');
    [aretes,text_aretes]=fscanfMat(nom+'_aretes'); maillage=struct("nom",
    text_aretes(1),...
                          "sommets",sommets,...
            "centres",centres,...
            "aretes",aretes,...
            "nb_vol",size(centres,1),...
            "nb_are",size(aretes,1),...
            "nb_som",size(sommets,1));
endfunction
```

4. **Struct :** Retourne une struct avec des noms de champs field1, field2, .., et dont les valeurs respectives sont value1, value2, ...

Enfin, le code va évaluer les maillages dans les différents points par le biais de cette fonction :

```
function [v]=eval_fonction(points,f)
    v=zeros(size(points,1),1);
    for i=1:size(points,1)
            v(i)=f(points(i,_X),points(i,_Y));
     end;
endfunction
```

Cette fonction prend entrée points qui est une matrice $M_n(R)$ ainsi que la fonction qu'on souhaite approximer et retourne un vecteur de taille N+2 rempli de valeurs issues de l'évaluation de la fonction f sur ces points.
Jusqu'à la fin de cette section on va faire notre analyse sur le cas 3 : **Solution du laplacien non nulle au bord** définit par la fonction :

$$u(x,y) = sin((1-x)(1-y)) + (1-x)^3(1-y)^2$$

3.3.2 Analyse des maillages :

Pour ce paragraphe, on ne s'intéresse qu'à la forme des maillages ce qui nous conduit à opter pour un niveau de raffinement de quatre. Nous allons maintenant analyser les solutions exactes obtenues.

Maillage rectangle

Définition 7 *N-rectangle Est une génération d'un rectangle à une grande dimension, il est défini comme le produit cartésien d'intervalles.*

Définition 8 *Maillage rectangle Un maillage rectangulaire de $\overline{\Omega}$ est un ensemble τ_h de N rectangles T_i qui vérifient :*

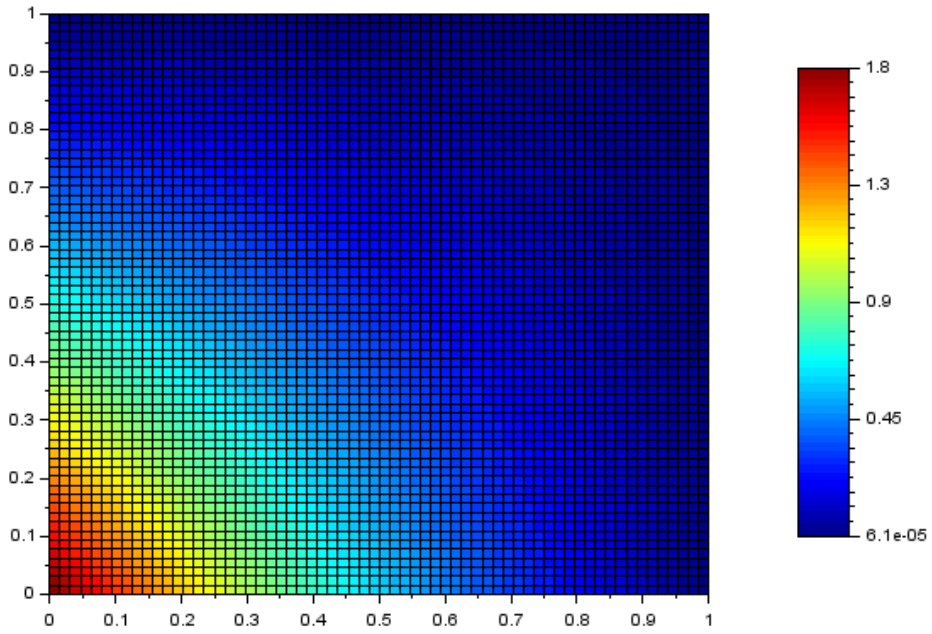
- *$T_i \in \overline{\Omega}$ et $\cup_{i=1}^{N} T_i = \overline{\Omega}$*
- *L'intersection $K_i \cap K_j$ de deux N-rectangles distincts est un m-rectangle, dont tous les sommets sont aussi des sommets de ces deux rectangle.*

Dans notre cas, on est en dimension 2 donc l'intersection sera une face commune.
Dans ce code on s'intéresse à un type particulier de maillages rectangles, les maillages rectangles uniforme, pour cela, on commencera par expliquer ce que veut dire maillages uniforme ensuite on verra les résultats obtenu par ce type de maillages.

Maillage rectangle uniforme :

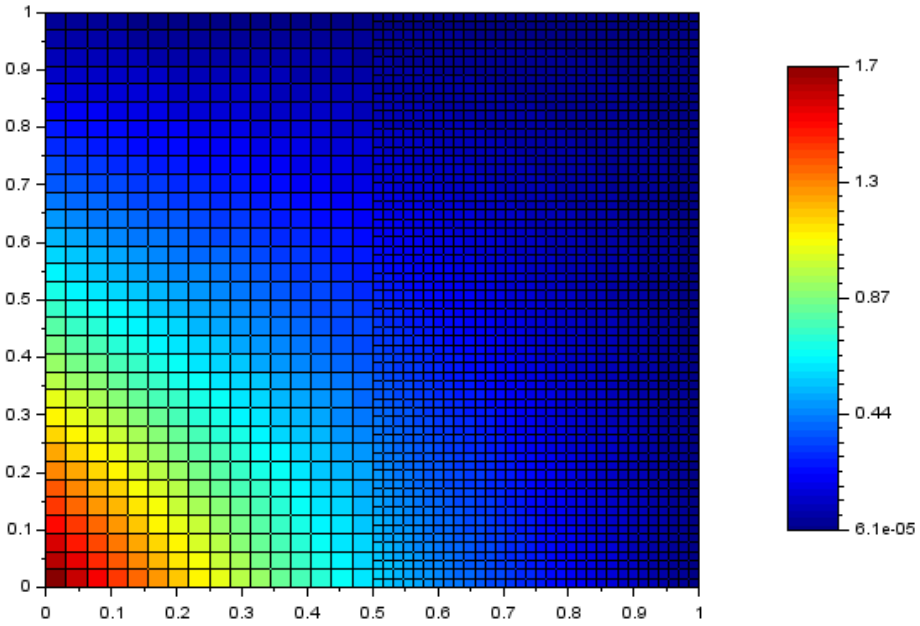
Définition 9 *Maillage uniforme Le maillage sera dit uniforme si les points x_i sont équidistants.*

Solution exacte obtenues :



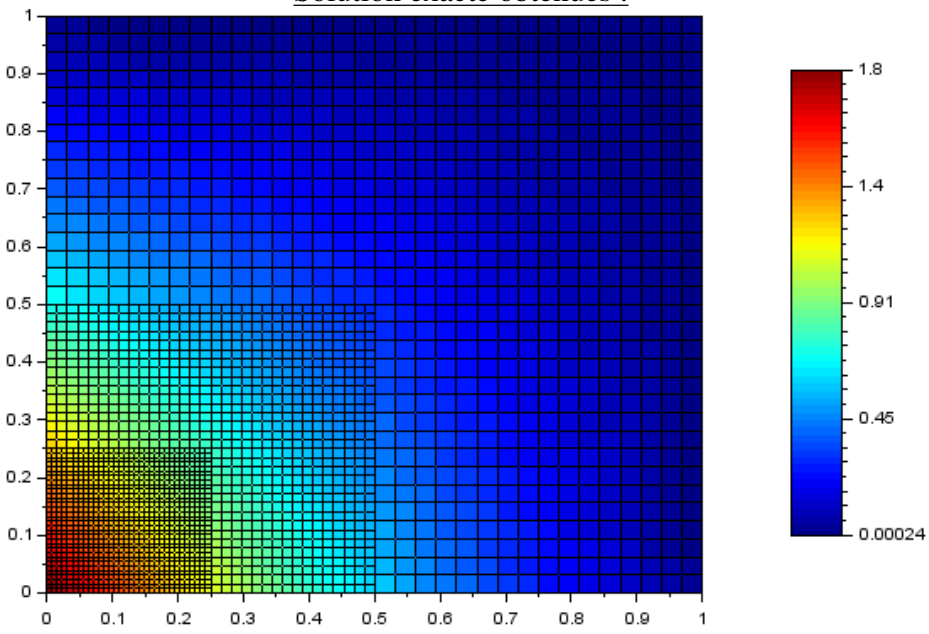
Maillage rectangle bi-domaine : Il s'agit d'un maillage ou on divise le domaine en deux, la seconde partie contient deux fois plus de volumes de contrôles.

Solution exacte obtenues :



Maillage local raffine : Il s'agit d'un domaine où le raffinement double en approche d'un bord.

Solution exacte obtenues :



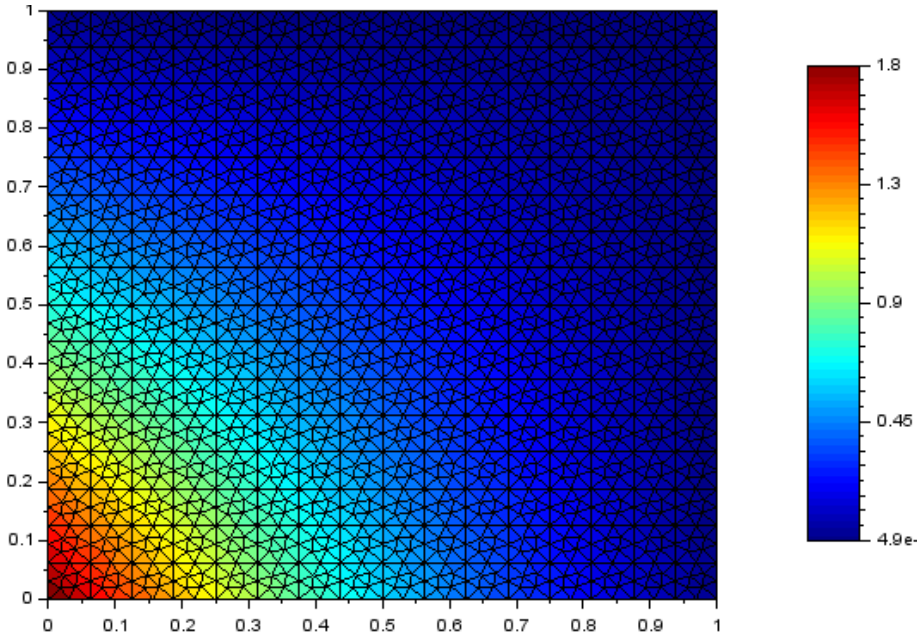
Maillage triangle :

Définition 10 *Maillage Un maillage triangulaire de* $\overline{\Omega}$ *est un ensemble* τ_h *de N-simplexes* T_i *qui vérifient :*

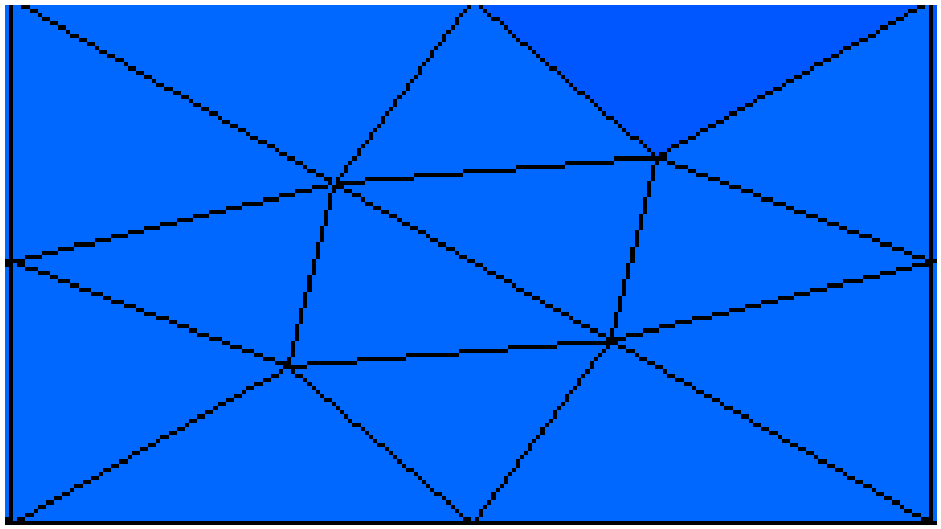
- $T_i \in \overline{\Omega}$ *et* $\cup_{i=1}^{N} T_i = \overline{\Omega}$
- *L'intersection* $K_i \cap K_j$ *de deux N-simplexes distincts est un m-simplexe avec* $1 < n < N$*, dont tous les sommets sont aussi des sommets de ces deux simplexes.*

En dimension N = 2, l'intersection de deux triangles est soit vide, soit réduite à un sommet commun, soit une arête commune entière ; en dimension N = 3, l'intersection de deux tétraèdres est soit vide, soit un sommet commun, soit une arête commune entière, soit une face commune entière.

Solution exacte obtenues :

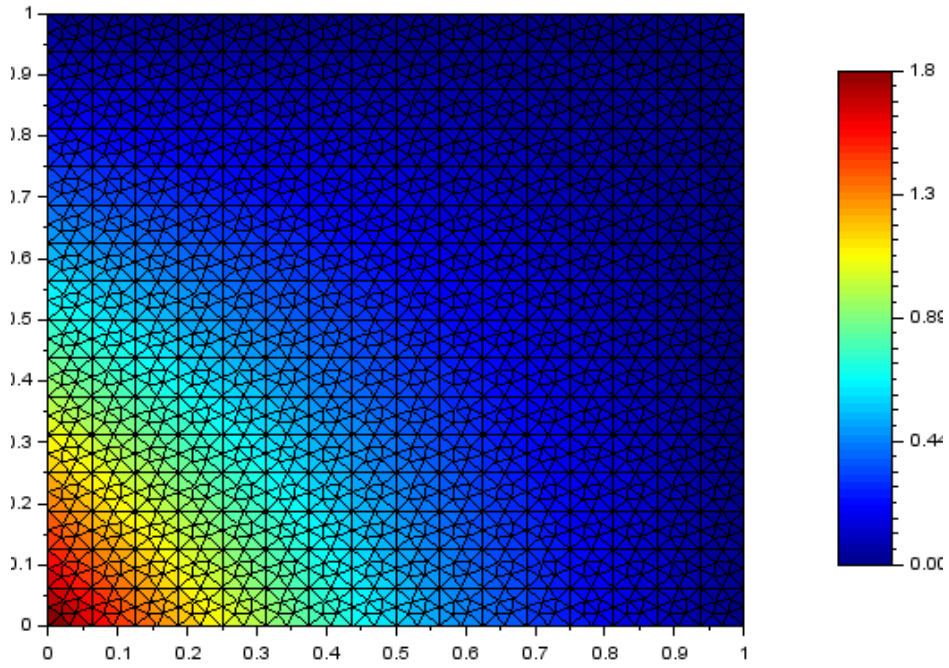


Maillage triangle centre gravite : Les maillages centre de gravité sont construit de cette manière, on discrétise notre espace en carré qu'on divise en deux triangle isocèles, ensuite on discrétise chaque triangle en sept triangles dont 5 partage comme arête commune le centre de gravité de ce triangle. Voici une image à grande échelle représentante ses maillages.



Ensuite présentera le résultat obtenus :

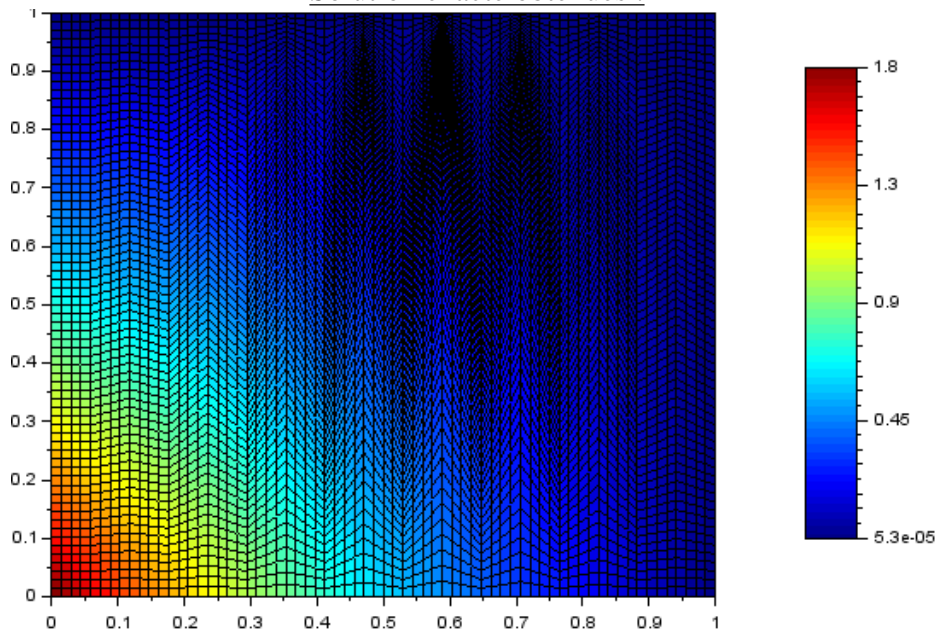
Solution exacte obtenues :



Maillage déformé :

Définition 11 *Maillage déformé : La déformation de maillages 2D est une technique utile pour l'animation d'objets, ils sont utilisé afin d'éviter à l'utilisateur de bouger tous les points du maillage à la main, on peut lui demander de placer les points de l'objet qui ne bougent pas et ensuite, de bouger un unique point du maillage.*

Solution exacte obtenues :



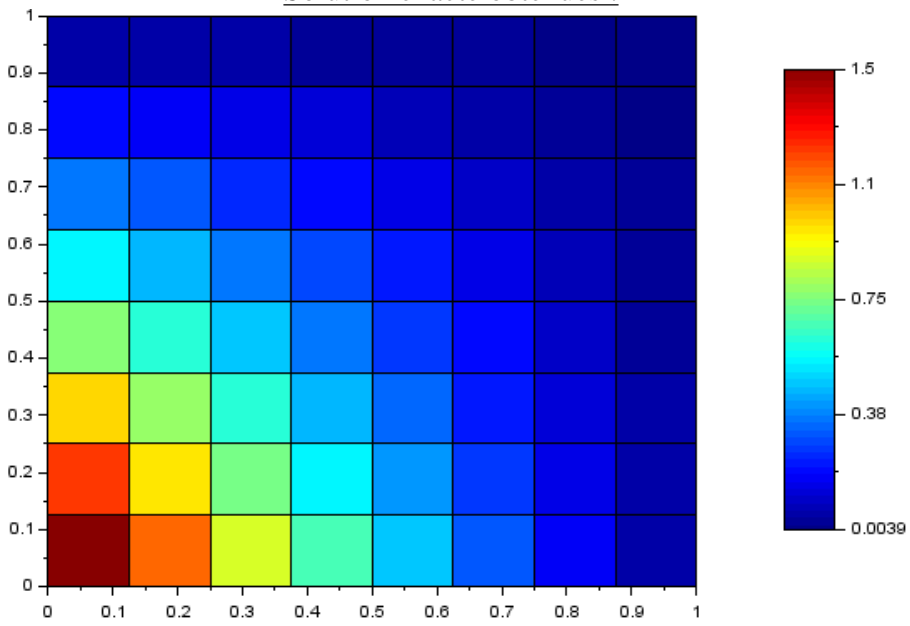
3.3.3 Analyse du niveau de raffinement :

Définition 12 *Le raffinement : Le raffinement de maillage adaptatif est une technique utilisée en analyse numérique pour résoudre des problèmes d'équations aux dérivées partielles sur des grilles adaptées à la solution du problème. Il permet de définir le nombre de volumes de contrôles.*

Par cela, plus le raffinement augmente (on parlera de maillage grossier dans ce cas) et plus le nombre de volumes par lesquels il est divisé est grand. Le code nous donne entre 5 et 6 choix possibles [5], allant de 1 à 6 et donnant un nombre de volumes élémentaires allant de 2^3 à 2^7.
À dessein d'illustrer l'effet du raffinement, nous allons choisir un maillage simple : "Le maillage rectangle uniforme".

Raffinement de niveau 1 : Ce maillage a 64 volumes de contrôle.

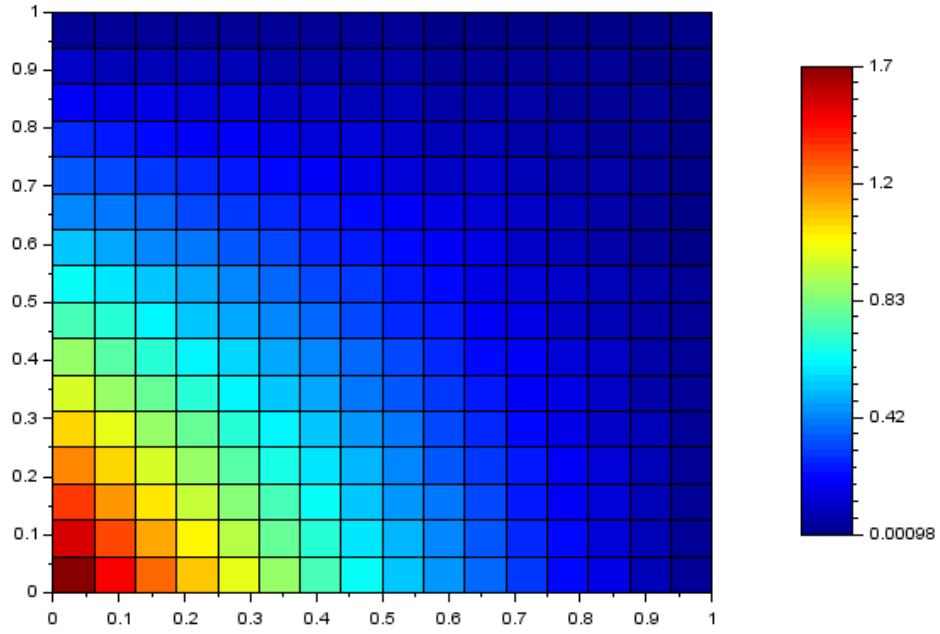
Solution exacte obtenues :



Raffinement de niveau 2 : Ce maillage a 256 volumes de contrôle.

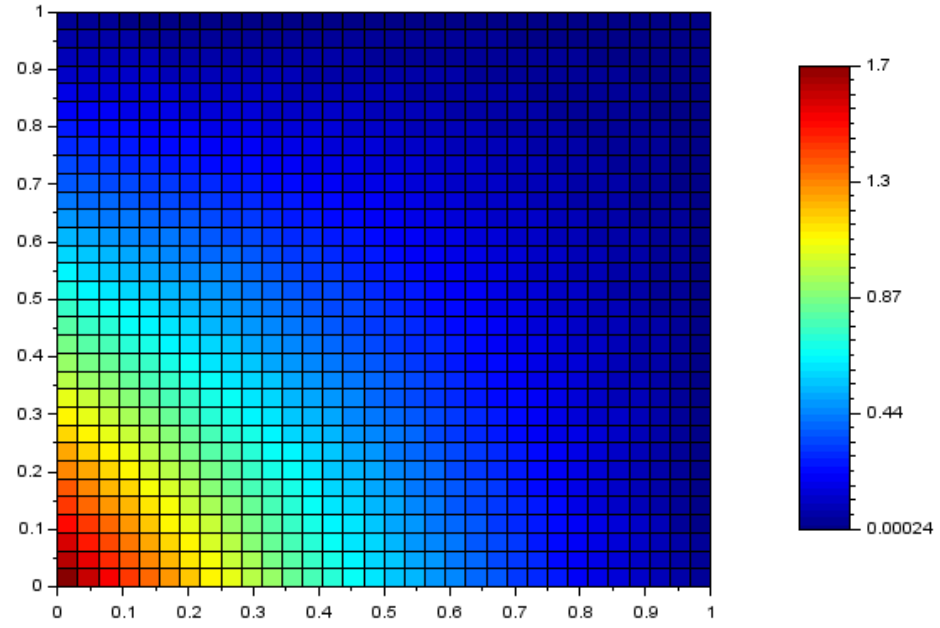
Solution exacte obtenues :

5. Dans notre exemple 5.



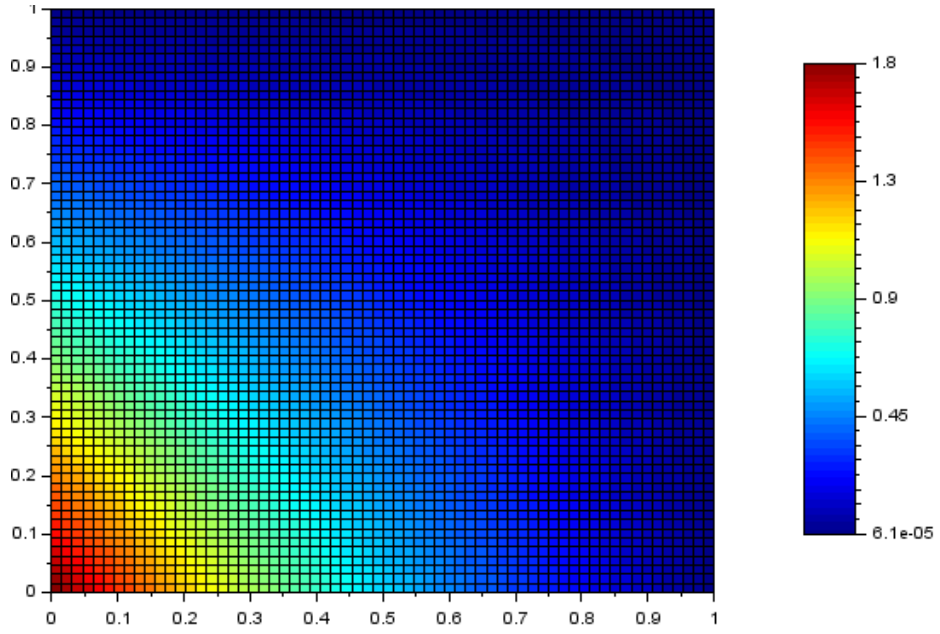
Raffinement de niveau 3 : Ce maillage a 1024 volumes de contrôle.

Solution exacte obtenues :



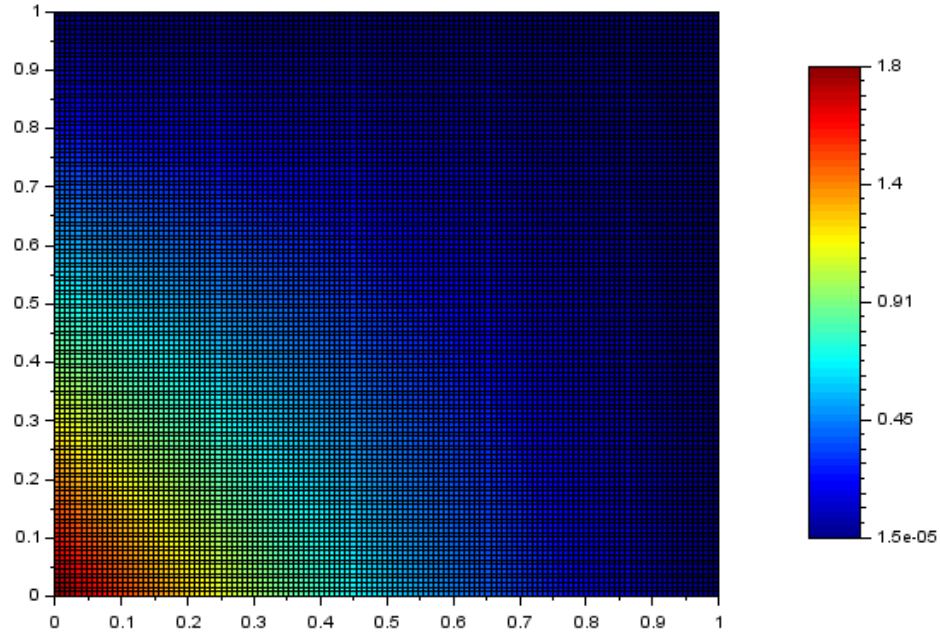
Raffinement de niveau 4 : Ce maillage a 4096 volumes de contrôle.

Solution exacte obtenues :



Raffinement de niveau 5 : Ce maillage a 16384 volumes de contrôle.

Solution exacte obtenues :



Conclusion : Nous remarquons que les courbes obtenus avec un grand raffinement c'est à dire supérieur à 4, sont plus proches de

4 Principe du schéma

Comme préciser avant nous supposons que les trois propriétés sont vérifiées :

- Ω un ouvert de $\mathbb{R}^2$
- $f \in L^2(\omega)$
- $f \in L^2(\partial\omega)$

4.1 Partie théorique :

4.1.1 Notation :

Nous définissons quelques notations qui nous seraient utiles pour la suite.

- **Pas du maillage :** size(T_i)
- **Ensembles d'arêtes :** E, E_{ext}, E_{int}, E_k
- **Normales :** ν_k, $\nu_{k\sigma}$, ν_{Kl}
- **Volumes et Mesures :** |K|, |σ|
- **Distances :** $d_{k\sigma}$, $d_{L\sigma}$, d_{KL}, d_σ

4.1.2 Construction du schéma :

Définition 13 *Le schéma vf4 est un Stencil de 4 points en 2D sur maillages. Son explication nécessite une nouvelle définition, celle des maillages orthogonaux.*

Méthode des volumes finis : Le but de ce code est d'approximer l'équation de Laplace 2d par l'intermédiaire d'un schéma de Volumes finis du type V.F.4 :
Ainsi,

$$\begin{cases} -\Delta u(x) = f(x), \forall x \in \Omega \\ u(x) = g(u(x), x), \forall x \in \partial\Omega \end{cases}$$

Sur un maillage orthogonal admissible, on intègre notre équation sur les volumes de contrôles $K_i = [x_{i-1/2}, x_{i+1/2}]$, ce qui nous donne :

$$\int_{K_i} \frac{f}{[K_i]} = \int_{K_i} \frac{-\Delta u(x)}{[K_i]}$$

$$\Rightarrow \int_{K_i} \frac{f}{[K_i]} = \sum_{\sigma \in E_k} - \int_\sigma \Delta u(x) \nu_{k\sigma}$$

Approximation des flux : Ensuite, sur les pas du td3 "Equation de la place 2d", on va approximer les flux internes et frontière selon le type d'arête :

- **Cas d'une arête intérieure :**
$$F_{k\sigma}(U^T) = -|\sigma|\frac{U_L - U_K}{d_{[kL]}}$$
- **Cas d'une arête extérieur :**
$$F_{k\sigma}(U^T) = |\sigma|\frac{U_K}{d_{[kL]}}$$

Schéma VF4 : Enfin on peut écrire le schéma Vf4 qui approxime cette équation de cette manière :

$$\begin{cases} f_i = \sum_{\sigma \in E_k} - \int_\sigma \Delta u(x) \nu_{k\sigma} \\ F_{k\sigma}(U^T) = -|\sigma|\frac{U_L - U_K}{d_{[kL]}}, \sigma \in E_{int} \\ F_{k\sigma}(U^T) = |\sigma|\frac{U_K}{d_{[kL]}}, \sigma \in E_{ext} \end{cases}$$

Ainsi l'intégration de l'équation de base sur les volumes de contrôle nous donne [6] :

$$|T_i| f_i = -\sum_{i|k \not\subset \Gamma} \frac{l_{ik}}{d_{ik}}(u_k - u_i) - \sum_{i|k \subset \Gamma_D} \frac{l_{ik}}{d_{ik}}(u_d(x_k) - u_i) - \sum_{i|k \subset \Gamma_N} l_{ik} G_{ik}$$

4.2 Implémentation informatique :

L'implémentation du schéma se fait par parcours des maillages par arêtes selon trois phases distinctes. Ces phases seraient présentées dans la suite de cette sous-section.
Ainsi pour calculer notre solution, le code du schéma fonctionne de cette manière :

- Dans un premier temps, on évalue le terme source aux centres des mailles.
- Ensuite on calcul du coefficient de diffusion sur chaque arête.
- Puis on évalue les données au niveau des arêtes du bord et des arêtes intérieures.
- Enfin on fait le calcul des termes sources.

4.2.1 Calcul du coefficient de diffusion :

Le calcul du coefficient de diffusion aux arêtes selon se fait par le biais de cette routine qui prend en paramètres les données choisies par l'utilisateur et un type de maillage. Cette routine effectue le calcul du coefficient de diffusion (*cette fonction n'est utilisé que pour les trois derniers cas test.*).

6. Les calculs ont été fortement inspiré du TD 3

Description du code : Les coefficient de diffusion sur les arêtes sont calculées par le biais de la fonction **calcul coef k interf** qui effectue le calcul ainsi que la fonction **fonction eval** qui permet de faire l'évaluation .
Cette fonction permet de faire l'évaluation des coefficients de diffusion selon trois méthodes :

- **Méthode exacte :** Le calcul exact du coefficient se fait aux centres des arêtes.
- **Calcul par moyenne arithmétique :** On fait le calcul des valeurs du coefficient aux centres des mailles voisines.
- **Calcul par moyenne harmonique :** On fait la moyenne harmonique pondérée des valeurs du coefficient aux centres des mailles voisines.
  La moyenne harmonique est calculée par cette formule :

$$\bar{m} = \frac{\sum_{i=1}^{n} \alpha_i m_i}{\sum_{i=1}^{n} \alpha_i}$$

```
function [ck]=calcul_coeff_k_interf(m,donnees);
    if isfield(donnees,"methode") then
donnees.methode="exacte";
end;

centres_aretes=(m.sommets(m.aretes(:,_DEB),:)+m.sommets(m.aretes(:,_FIN),:))/2;
select donnees.methode
case "exacte"
// On fait le calcul exact du coefficient aux centres des aretes

ck=eval_fonction(centres_aretes,donnees.coeff_k);

case "arithmetique"
ck=eval_fonction(m.centres(m.aretes(:,_K),[_X _Y]),donnees.coeff_k);

temp=find(m.aretes(:,_L)>0);
ck(temp)=0.5*(ck(temp) ...
+ eval_fonction(m.centres(m.aretes(temp,_L),[_X _Y]),donnees.coeff_k));

case "harmonique"
ck=eval_fonction(m.centres(m.aretes(:,_K),[_X _Y]),donnees.coeff_k);

temp=find(m.aretes(:,_L)>0);
ckL=eval_fonction(m.centres(m.aretes(temp,_L),[_X _Y]),donnees.coeff_k);

ck(temp)=  m.aretes(temp,_DKL).*ck(temp).*ckL ...
./(m.aretes(temp,_DKsigma).*ckL+m.aretes(temp,_DLsigma).*ck(temp));
else
disp("Vous n''avez pas choisi une methode de calcul du coeff de"+...
" diffusion valable!");
halt;
end;
endfunction
```

4.2.2 Schéma VF4 :

Après avoir approximé les coefficients de diffusion, la seconde phase de ce programme consiste à utiliser un schéma VF4 programmé à l'aide d'opérations vectorielles afin de prendre en compte les CL de Dirichlet non-homogène, les CL de Neumann homogène et les fractures à l'intérieur du domaine.
Ce schéma opéré tel le schéma théorique, à une seule différence près, on ne s'intéresserait pas aux arêtes extérieures le schéma implémenté sera de cette forme :

$$\begin{cases} f_i = \sum_{\sigma \in E_k} - \int_\sigma \Delta u(x) v_{k\sigma} \\ F_{k\sigma}(U^T) = -|\sigma| \frac{U_L - U_K}{d_{[kL]}}, \sigma \in E_{int} \end{cases}$$

Les arêtes sont reconnues grâce à l'opérateur LABEL. Elles sont évalué de cette manière :

- label < -1 si l'arête est sur le bord avec condition aux limites de type Neumann homogène
- label = -1 si l'arête est sur le bord avec condition aux limites de type Dirichlet non homogène
- label = 0 s'il s'agit d'une arête intérieure
- label = 1 si l'arête subit une fracture à l'intérieur du domaine

```
function [A,b]=const_schema_VF4(m,donnees)
source=eval_fonction(m.centres,donnees.source);
coeff_diff=calcul_coeff_k_interf(m,donnees);
indiA=[];
indjA=[];
valA=[];
indib=[];
indjb=[];
valb=[];

tauKL=coeff_diff(:).*m.aretes(:,_MES)./m.aretes(:,_DKL);

temp=find(m.aretes(:,_LABEL)==1);
tauKL(temp)=donnees.beta*tauKL(temp).*m.aretes(temp,_DKL)...
./(1+donnees.beta*m.aretes(temp,_DKL));

end;

temp=find(m.aretes(:,_LABEL)==-1);

ubord=eval_fonction(...
0.5*(m.sommets(m.aretes(temp,_DEB),[_X _Y])+m.sommets(m.aretes(temp,_FIN),[_X _Y]))
    ...
,donnees.bordD);
indiA=m.aretes(temp,_K);
indjA=indiA;
valA=tauKL(temp);
indib=[indib; m.aretes(temp,_K)];
valb=[valb; tauKL(temp).*ubord];
```

Description du code :

4.2.3 Calcul du terme source

La troisième phase du code consiste à effectuer l'évaluation du terme source qui concerne tout le monde. Cette évaluation se fait par des formules de quadratures.

```
indib=[indib; m.aretes(:,_K)];
valb=[valb; (m.aretes(:,_MES_K_DEB)+m.aretes(:,_MES_K_FIN)).*source(m.aretes(:,_K))
    ];
temp=find(m.aretes(:,_L)>0);
indiA=[indiA; m.aretes(temp,_K)];
indjA=[indjA; m.aretes(temp,_K)];
valA=[valA; tauKL(temp)];
indiA=[indiA; m.aretes(temp,_K)];
indjA=[indjA; m.aretes(temp,_L)];
valA=[valA; -tauKL(temp)];
indiA=[indiA; m.aretes(temp,_L)];
indjA=[indjA; m.aretes(temp,_K)];
valA=[valA; -tauKL(temp)];
indiA=[indiA; m.aretes(temp,_L)];
indjA=[indjA; m.aretes(temp,_L)];
valA=[valA; tauKL(temp)];
indib=[indib; m.aretes(temp,_L)];
valb=[valb; (m.aretes(temp,_MES_L_DEB)+m.aretes(temp,_MES_L_FIN)).*source(m.aretes(
    temp,_L))];

/// Construction proprement dite

indjb=ones(indib);
A=sparse([indiA indjA], valA);
b=sparse([indib indjb], valb);
b=full(b);

endfunction
```

4.3 Application :

Dans cette sous-section on va décrire comment la solution approchée de notre cas test est calculée à partir du schéma. Les bout de codes qu'on va montrer dans cette section se trouvent dans **VF2D.sce**.

4.3.1 Calcul de la solution exacte et approchée :

Pour chaque cas test on essaye de trouver la solution exact et approchée de notre fonction.
Premièrement le calcul de la solution exacte se fait avec la fonction "eval fonction" qui permet de calculer la valeur exacte de la solution en chaque point de maillage.
Code utilisé :

```
if isfield(donnees,"methode") then
donnees.methode="exacte";
end;
```

La solution rapprochée quant à elle, son calcule repose sur la construction de notre matrice A et b le second membre par l'intermédiaire de la routine schéma VF4. Dans notre cas, le calcul de cette solution approchée se fait via ce bout de code :

```
if isfield(donnees,"uexacte") then
solexacte=eval_fonction(m.centres,donnees.uexacte);
end;

select choix_schema
case 1
[A,b]=const_schema_VF4(m,donnees);
```

4.3.2 Résolution

Enfin la résolution de notre système se fait par le biais d'un solveur scilab ou via Umfpack, un ensemble de routine créées pour la résolution des systèmes linéaires creux et non symétriques de la forme Ax=b où A n'est pas nécessairement symétrique. Cette résolution se fait par le biais d'une méthode non symétrique multi-frontale.
Voici le code qui permet cette résolution :

```
printf('Resolution\n');
try
// On essaie d'abord UMFPACK
sol=umfpack(A,'\',b);
catch // Sinon on resoud par le solveur Scilab
sol=A\b;
end,
```

Maintenant que la résolution a été effectué on va approximer les erreurs. Cette approximation fera l'objet de notre dernière section.

5 Courbes d'Erreurs :

5.1 Implémentation :

5.2 Application sur les différents cas test :

Pour la courbe d'erreurs, on va faire la comparaison de notre ordre d'erreur par rapport à trois normes :

- Norme ∞
- Norme $L^2(\Omega)$
- Norme $H^1(\Omega)$

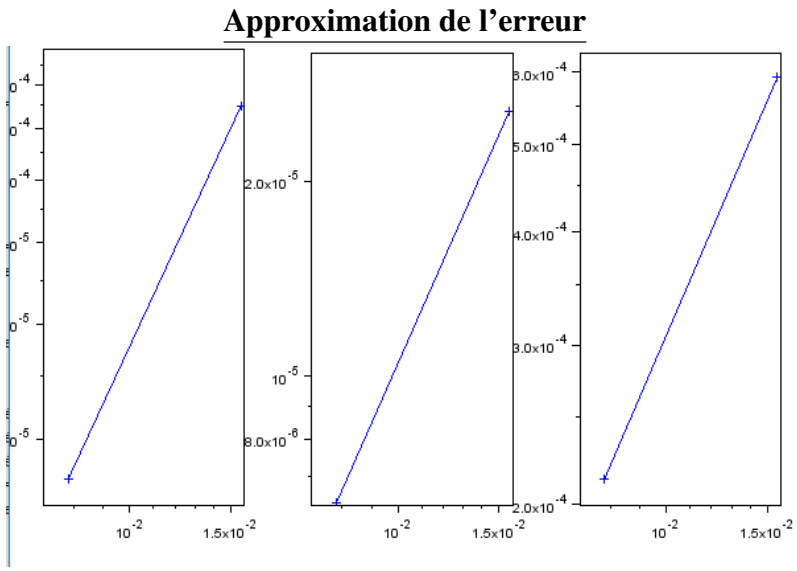
Malheureusement, on ne peut pas trop parler des erreurs car pour les trois normes on obtient un ordre de convergence de 0.

Définition 14 *Ordre de consistance d'un schéma : Une méthode consistante est dite d'ordre p si* $\forall$ *compact K il existe C > 0, tel que pour toute solution z(t), de graphe (t, z(t)) contenu dans K, l'erreur de consistance* e_n *satisfait la condition :*

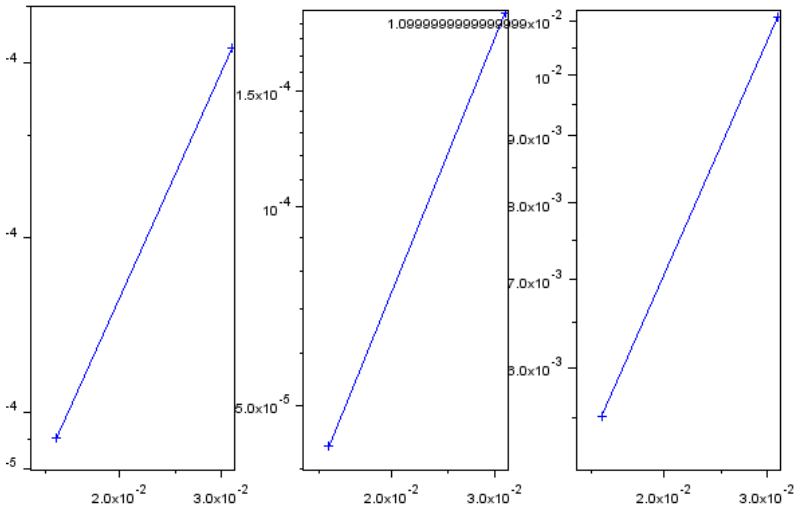
$$|e_n| < Ch^{p+1}$$

L'erreur de consistance est aussi appelée l'erreur de convergence, l'ordre du schéma ou encore l'ordre de consistance. Pour la suite de cette section nous allons montrer les graphiques de l'erreur par rapport au temps.

5.2.1 Produit de sinus pour le Laplacien

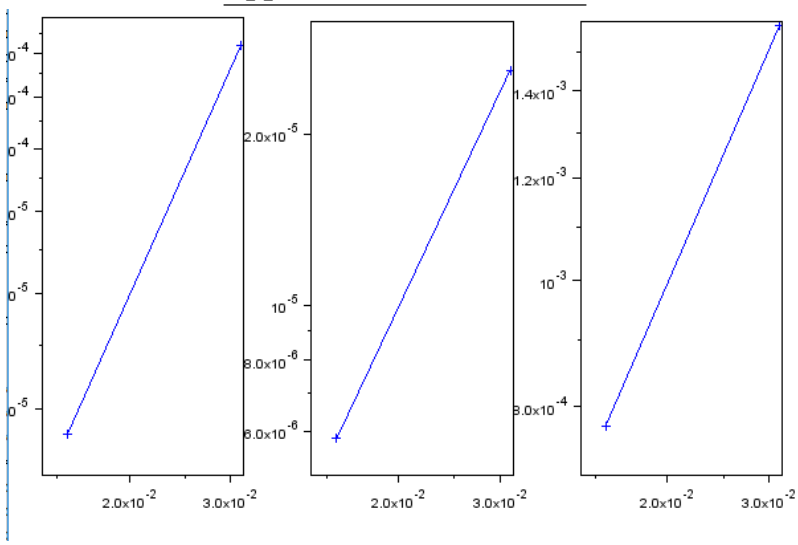


5.2.2 Polynôme pour le laplacien



5.2.3 Solution du laplacien non nulle au bord

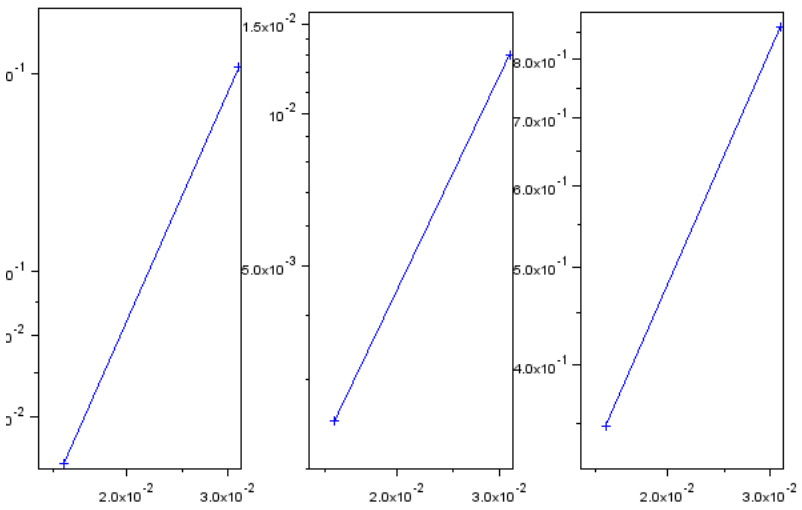
Approximation de l'erreur



Pour les sections suivantes on va utiliser une méthode de calcul des coefficients de diffusion exacte :

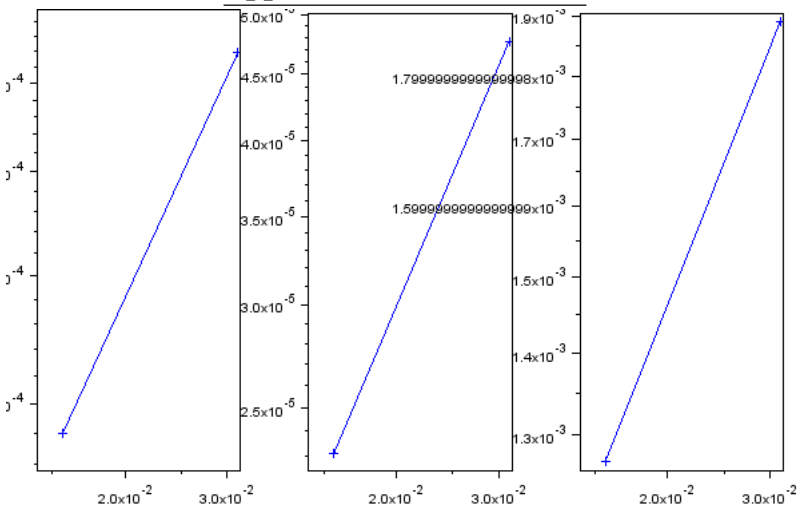
5.2.4 Produit de sinus anisotrope

Approximation de l'erreur



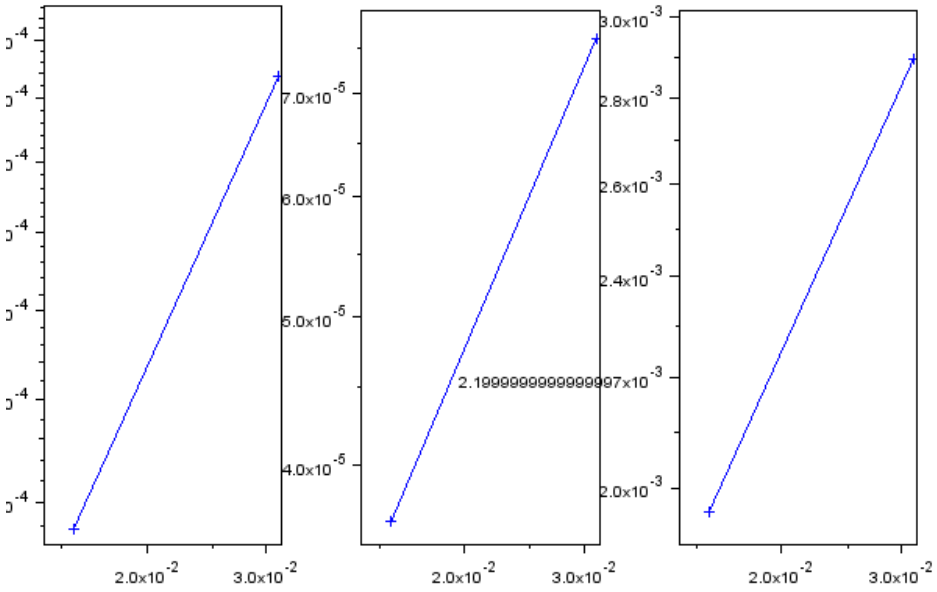
5.2.5 Polynomiale isotrope discontinue

Approximation de l'erreur



5.2.6 Polynomiale anisotrope discontinue

Approximation de l'erreur


7.0x10^-5
6.0x10^-5
5.0x10^-5
4.0x10^-5
2.0x10^-2
3.0x10^-2
3.0x10^-3
2.8x10^-3
2.6x10^-3
2.4x10^-3
2.199999999999997x10^-3
2.0x10^-3
2.0x10^-2
3.0x10^-2
2.0x10^-2
3.0x10^-2