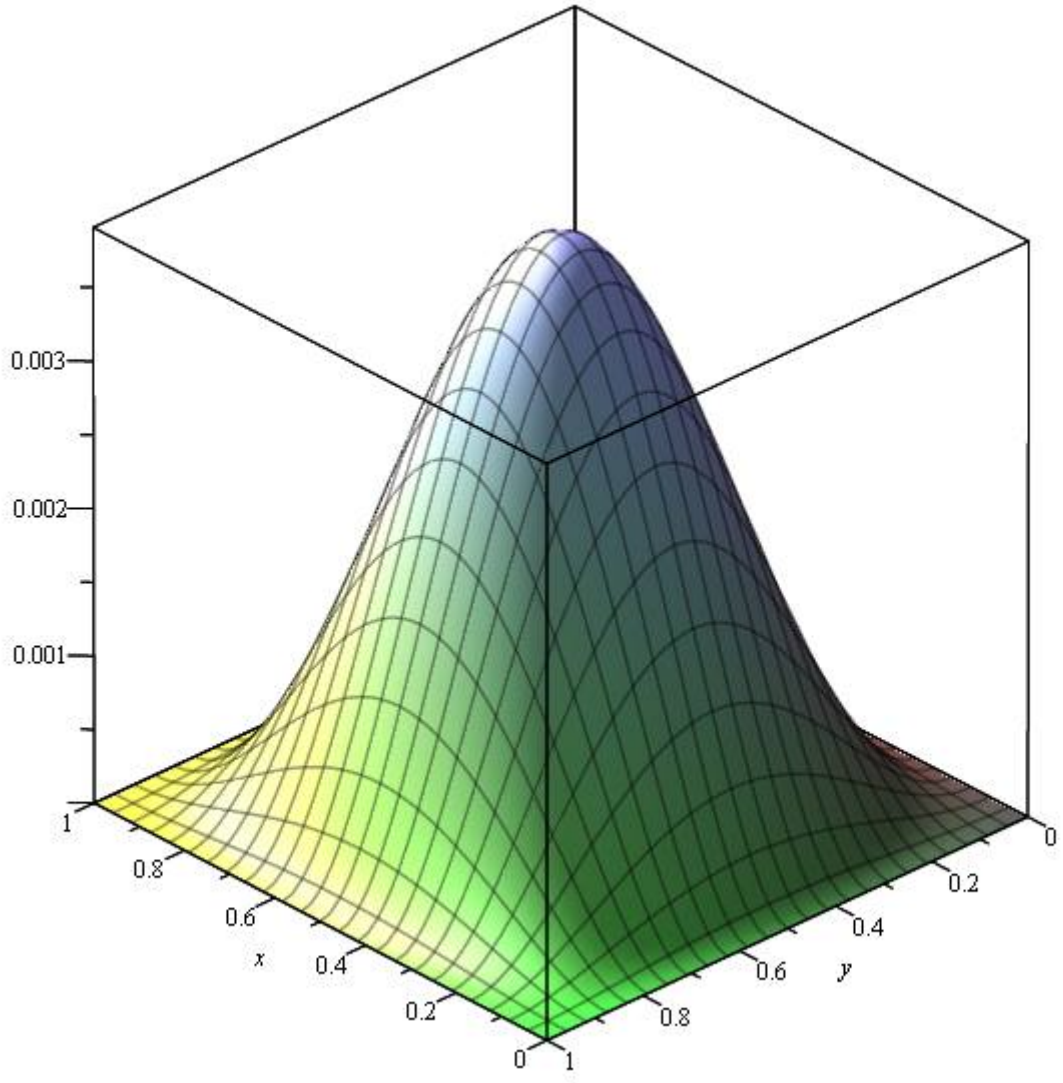
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN

Projet de Physique P6
STPI/P6/2016-12

Déformation d'une membrane



Enseignant responsable
Bernard GLEYSE

Étudiants :
Laura BLASZCZYK
Morgan BUISSON
Ninon CONSTANT
Ou LIU
Mathieu WALES

Date de remise du rapport : 13/06/16

Référence du projet : STPI[1]/P6/2016-12

Intitulé du projet : Déformation d'une membrane

Type de projet : *Bibliographie, modélisation*

Objectifs du projet : L'objectif de ce projet est de modéliser la déformation d'une membrane élastique rectangulaire. Pour cela, notre étude a été décomposée en trois parties : une modélisation physique, une résolution mathématique et une simulation numérique.

Mots-clefs du projet : Membrane, déformation, laplacien, bilaplacien, équation de Poisson, méthode de Cholesky.

1. INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE ROUEN
DÉPARTEMENT SCIENCES ET TECHNIQUES POUR L'INGÉNIEUR
685 AVENUE DE L'UNIVERSITÉ BP 08- 76801 SAINT-ETIENNE-DU-ROUVRAY
TÉL : 33 2 32 95 66 21 - FAX : 33 2 32 95 66 31

Table des matières

Introduction

Le but de ce projet est de modéliser la déformation d'une membrane rectangulaire. Pour cela nous avons repris le travail entrepris par le groupe de l'an dernier et nous l'avons approfondi.

L'objectif, dans un premier temps, était de valider le modèle de l'an passé. Nous nous sommes ensuite tournés vers une nouvelle représentation, qui permet une définition différente de la déformation de membrane. Pour ce nouveau modèle, nous avons divisé notre travail en différentes parties avec : une modélisation mathématique, numérique et physique. La modélisation mathématique nous a donné une solution approchée de la relation entre la force qui agit sur la membrane et le comportement de cette dernière. La solution numérique nous a permis de rendre possible l'application de la solution mathématique et d'obtenir des valeurs exploitables. La modélisation physique, elle, nous a rendu possible l'observation de la validité du modèle trouvé. Pour ce faire, nous avons pu modifier différents paramètres et étudié le problème dans différents contextes.

Chapitre 1

Résolution avec le Laplacien

1.1 Résolution mathématique

Dans cette partie, nous nous intéressons à la solution apportée l'année dernière. Nous définissons une membrane rectangulaire, horizontale et élastique. On la met sous tension en la fixant par les bords et en exerçant une force transversale. L'équation de poisson que l'on utilise est comme suit :

$$-c_1 \Delta u = f$$

où u est le terme source et c_1 est la tension appliquée à la membrane.

Comme décrit précédemment, une condition est donnée sur les bords de la membrane : ils sont fixés, c'est-à-dire $|_{\partial\Omega} = 0$.

Pour caractériser le comportement de la membrane, il est nécessaire de déterminer les positions de nombreux points. Pour ce faire, on va procéder à une discrétisation de la membrane, qui consiste à quadriller la surface. On pose $u_{i;j} = u(x_i, y_j)$ comme la hauteur du point considéré aux coordonnées i,j. Désignons par L_x sa longueur et L_y sa largeur. La membrane sera désignée par $\Omega = [0, L_x][0, L_y]$. N et M sont le nombre de divisions respectives des côtés de la membrane. On obtient alors deux pas de discrétisation h et k s'exprimant tel que :

$$h = \frac{L_x}{N+1} \text{ et } k = \frac{L_y}{M+1}$$

On a : $x_i = ih$ et $y_j = jk$ où $i, j = 0, .., 1$.

Le groupe de l'an passé a utilisé la méthode des différences finies afin de faire une résolution locale pour un couple $(x_i; y_j)$. La solution a été approchée par des dérivées partielles secondes de u dans des développements de Taylor.

Après avoir considéré un accroissement local en x et additionné les équations ainsi trouvées. On néglige les restes et on obtient :

$$\frac{1}{h^2}[u(x_{i+1}; y_j) - 2u(x_i; y_j) + u(x_{i-1}; y_j)] \simeq \frac{\partial^2 u}{\partial x^2}(x_i; y_j) \tag{1.1}$$

Par analogie, on a considéré un accroissement local en y, additionné les équations obtenues et négligé les restes :

$$\frac{1}{k^2}[u(x_i; y_{j+1}) - 2u(x_i; y_j) + u(x_i; y_{j-1})] \simeq \frac{\partial^2 u}{\partial y^2}(x_i; y_j) \tag{1.2}$$

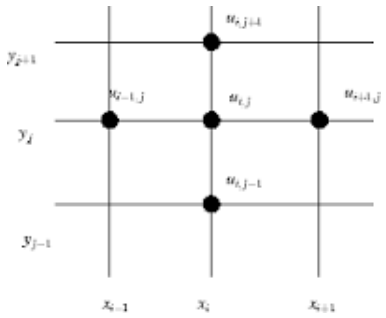
Or on sait que $-c_1 \triangle u = -c_1 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) = f$

$\Leftrightarrow -c_1 \left(\frac{1}{h^2} (u(x_{i+1}; y_j) - 2u(x_i; y_j) + u(x_{i-1}; y_j) + \frac{1}{k^2} (u(x_i; y_{j+1}) - 2u(x_i; y_j) + u(x_i; y_{j-1})) \right) = f_{i,j}$

Pour la suite de la résolution, nous allons prendre $M = N$, $h = k$ et $c_1 = 1$. On obtient des équations linéaires sous la forme :

$$-u_{i+1,j} - u_{i,j+1} + 4u_{i,j} - u_{i-1,j} - u_{i,j-1} = h^2 f_{i,j}$$

Cette équation est valable pour les points de coordonnées $i, j = 1..N-1$ car on a précisé que les points aux bords de la membrane avaient une hauteur nulle (ils sont fixés).
On obtient alors un calcul de la force en un point, qui est en fonction des positions des 4 points les plus proches du point considéré. La figure ci-après représente la situation pour un point (i, j) donné.



Représentons ces équations linéaires sous forme matricielle, c'est-à-dire $Au = b$ où A est la matrice des coefficients des équations linéaires, u est le vecteur hauteur des points de la grille et $b = h^2 f$.

On obtient alors une matrice carrée tridiagonale par blocs de dimension $(N-1)^2$ de la forme :

$$\begin{pmatrix} G & -I & 0 & \dots & 0 \\ -I & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -I \\ 0 & \dots & 0 & -I & G \end{pmatrix}$$

Le bloc I est la matrice identité de dimension $N-1$. Le bloc G est explicité ci-dessous :

$$\begin{pmatrix} 4 & -1 & 0 & \dots & 0 \\ -1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 4 \end{pmatrix}$$

Pour mieux visualiser la représentation matricielle, donnons des valeurs numériques aux paramètres du problème. Prenons $N = 4$, $h = \frac{1}{4}$ et $f = 16$.
Les équations linéaires seront alors de la forme :

$$-u_{i+1,j} - u_{i,j+1} + 4u_{i,j} - u_{i-1,j} - u_{i,j-1} = 1$$

Construisons d'abord la matrice A en donnant des valeurs à i et j dans l'équation ci-dessus.

$$\begin{array}{ll}
i,j = 1,1 & -u_{2,1} - u_{1,2} + 4u_{1,1} - u_{0,1} - u_{1,0} = 1 \\
i,j = 2,1 & -u_{3,1} - u_{2,2} + 4u_{2,1} - u_{1,1} - u_{2,0} = 1 \\
i,j = 3,1 & -u_{4,1} - u_{3,2} + 4u_{3,1} - u_{2,1} - u_{3,0} = 1 \\
i,j = 1,2 & -u_{2,2} - u_{0,2} - u_{1,1} - u_{1,3} + 4u_{1,2} = 1 \\
i,j = 2,2 & -u_{3,2} - u_{1,2} - u_{2,1} - u_{2,3} + 4u_{2,2} = 1 \\
i,j = 3,2 & -u_{4,2} - u_{2,2} - u_{3,1} - u_{3,3} + 4u_{3,2} = 1 \\
i,j = 1,3 & -u_{2,3} - u_{0,3} - u_{1,2} - u_{1,4} + 4u_{1,3} = 1 \\
i,j = 2,3 & -u_{3,3} - u_{1,3} - u_{2,2} - u_{2,4} + 4u_{2,3} = 1 \\
i,j = 3,3 & -u_{4,3} - u_{2,3} - u_{3,2} - u_{3,4} + 4u_{3,2} = 1
\end{array}$$

Remplissons maintenant la matrice.

$$\begin{pmatrix}
4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\
-1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\
0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\
0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\
0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4
\end{pmatrix}$$

On voit bien l'illustration de la matrice tridiagonale par blocs de dimension $9 = (N-1)^2$. Cette matrice sera utilisée dans la modélisation numérique. Cette dernière ayant pour but de servir la modélisation physique.

1.2 Simulation numérique

1.2.1 Méthode de Cholesky - Programme Pascal

Algorithme

Notre traitement numérique du problème nous amenait à résoudre une équation du type $AX = B$, nous avons donc opté pour la méthode de Cholesky (facilement programmable sous Matlab à l'aide de la commande $X = B/A$). Soit A notre matrice symétrique définie positive et X vecteur colonne :

$$A = A^T A$$
$$X^T AX > 0 \qquad \forall x \neq 0$$

D'après le théorème de la décomposition de Cholesky, pour une matrice A telle que la nôtre nous avons :

$$A = RR^T$$

On calcule donc R par identification en exprimant les r_{ij} en fonction des a_{ij} :

$$a_{ij} = \sum_{k=1}^{n} R_{ik} R_{kj}^T \qquad i \leqslant j$$

$$R = \begin{pmatrix} r_{11} & 0 & \dots & 0 & \dots & 0 \\ r_{21} & r_{22} & \dots & 0 & \dots & 0 \\ \vdots & \vdots & & \vdots & & \vdots \\ r_{i1} & r_{i2} & \dots & r_{ii} & \dots & 0 \\ \vdots & \vdots & & \vdots & & \vdots \\ r_{n1} & r_{n2} & \dots & r_{ni} & \dots & r_{nn} \end{pmatrix}$$

On identifie d'abord la 1ère ligne de A (i= 1) :

$$a_{11} = r_{11} r_{11} = r_{11}^2 \rightarrow r_{11} = \sqrt{a_{11}}$$
$$a_{1j} = r_{11} r_{j1} \rightarrow r_{j1} = \frac{a_{1j}}{r_{11}} \qquad j = 2, ...n$$

Et de même pour i = 2,. . . ,n :

$$a_{ii} = \sum_{k=1}^{i-1} r_{ik}^2 + r_{ii}^2 \rightarrow r_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} r_{ik}^2}$$
$$a_{ij} = \sum_{k=1}^{i-1} r_{ik} r_{jk} + r_{ii} r_{ji} \rightarrow r_{ji} = \frac{1}{r_{ii}} \left[\sum_{k=1}^{i-1} r_{ik} r_{kj} \right]$$

Avec $A = RR^T$, on arrive à :

$$RR^T X = B$$

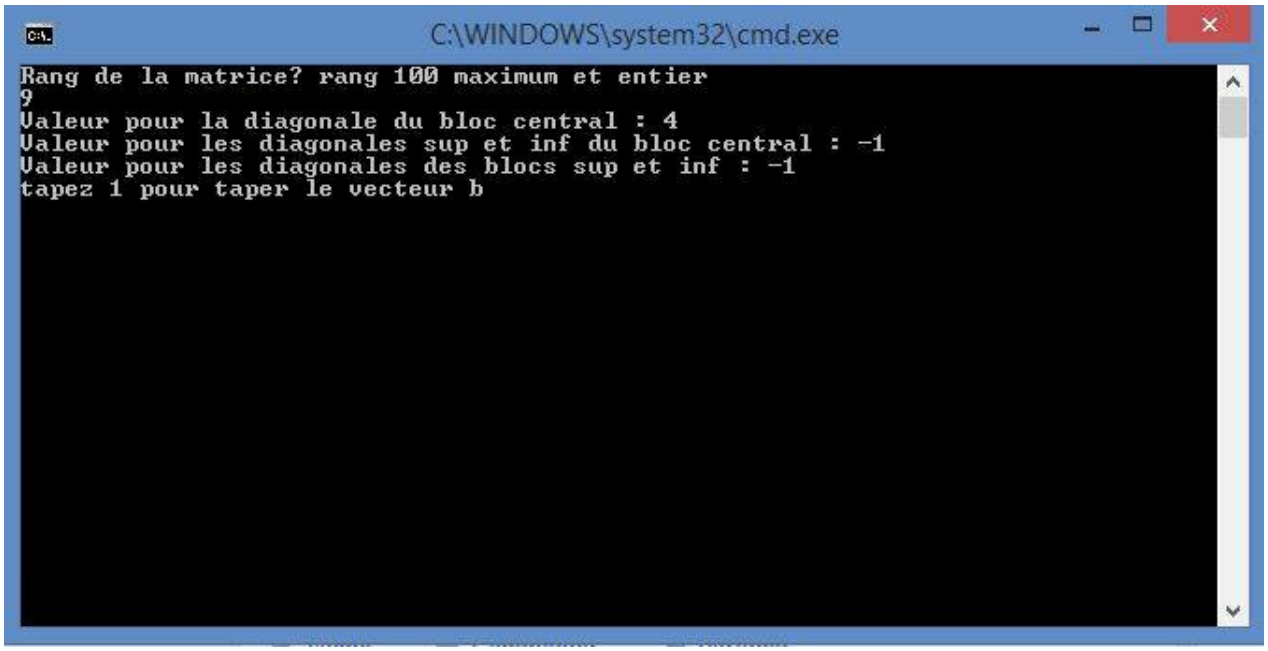
On décompose ensuite le système tel que :

$$RY = B \qquad systeme\ triangulaire\ inferieur$$
$$R^T X = Y \qquad systeme\ triangulaire\ superieur$$

On peut ainsi résoudre $AX = B$ en résolvant les deux équations précédentes. Pour cela, nous procéderons par ordre décroissant des indices, comme dans la méthode de Gauss, pour la première puis par ordre croissant pour la première.

Nous avons récupéré le programme de l'an dernier qui permettait de déterminer les forces appliquées à la membrane sur 9 points différents. Le programme était capable de générer une matrice en saisissant tout simplement sa dimension et les valeurs de chacune de ses diagonales (différentes de 0).
Nous avons simplement fait en sorte de pouvoir rentrer n'importe quel vecteur force B dans ce programme, ceci étant nécessaire pour le calcul des positions pour une matrice théorique de hauteur particulière. Ayant un tel système matriciel, il est aussi très facile, à partir des positions des points et de la matrice A, de déterminer le vecteur force B par simple multiplication de matrice. Mr Gleyse nous a cependant indiqué que l'on ne s'intéressera pas à cette résolution, ne permettant pas de vérifier la validité du modèle utilisé. Un tel calcul est néanmoins possible dès lors que le modèle est considéré juste.

Le programme final, qui se nomme $deformationMembrane.pas$ a donc pour but de résoudre un système du type $Ax = b$ en utilisant la méthode de Cholesky.



1.3 Simulation physique

Dans cette partie nous allons vérifier la validité du modèle du Laplacien. On utilise un vecteur position qui respecte les conditions limites aux bords, c'est-à-dire $|_{\partial\Omega} = 0$.

On va calculer la force exercée sur la membrane. Pour ce faire, on va calculer le Laplacien à l'aide des dérivées secondes.

On pose l'hypothèse suivante : $-\Delta p = \frac{h^2 f}{c_1}$

A l'aide de cette hypothèse, on calcule le vecteur force B. Le vecteur position est calculé à l'aide de la méthode numérique de résolution de $AX = B$. Par la suite, nous regardons la

différence des résultats entre le vecteur position obtenu avec le vecteur position exact. Nous avons ensuite étudié l'erreur correspondante à l'utilisation du modèle du laplacien. Cette procédure a été répétée plusieurs fois pour différents maillages.

Exemple :
Choisissons tout d'abord une fonction des hauteurs des points de la membrane qui vérifie la condition initiale.

$p(x, y) = xy(x-1)(y-1)$

Calculons les dérivées secondes par rapport à x et y.

$$\begin{cases} \frac{\partial^2 p}{\partial x^2} = 2y^2 - 2y \\ \frac{\partial^2 p}{\partial y^2} = 2x^2 - 2x \end{cases}$$

Avec l'approximation $-\Delta p = \frac{h^2 f}{c_1}$, on obtient :

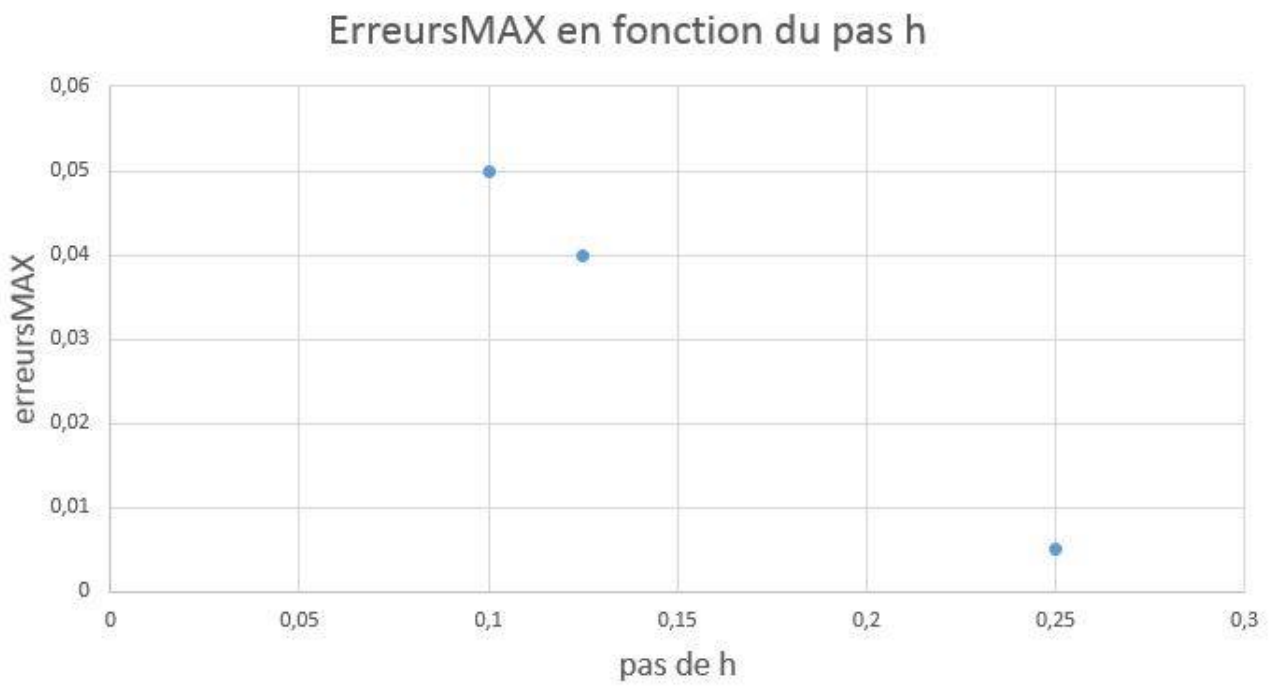
$-2y^2 + 2y - 2x^2 + 2x = fh^2$

Où $fh^2 = B$

Le vecteur B a été calculé en tous points avec cette formule à l'aide d'excel. À l'initialisation du programme $deformationMembrane.pas$ nous avons rentré le vecteur B, composé de la force exercée en chaque point.

Le vecteur X, composé de l'ensemble des vecteurs hauteurs des points de la membrane, peut alors être calculé.
On calcule dès lors X pour différentes valeurs de h (le pas entre chaque ligne du maillage).
Les erreurs maximum observées après calcul pour trois maillages sont listées ici :

erreurMAX $(h = 0.1) = 0.05$
erreurMAX $(h = 0.125) = 0.04$
erreurMAX $(h = 0.25) = 0.005$



Etudions ces résultats. On observe que les plus grandes valeurs d'erreur tendent à être obtenues au centre de la membrane, soit pour des valeurs plus grandes de hauteur.
Tout d'abord, il faut énoncer quelques valeurs de hauteur des point considérés pour mesurer l'importance de ces erreurs. Au centre de la membrane, la hauteur théorique vaut 1/16, soit 0,0625. Les erreurs calculées semblent alors assez conséquentes, mais elles restent tout de même du même ordre de grandeur que la hauteur. Ensuite, on semble voir que cette erreur augmente de manière inversement proportionnelle au pas h. Cela signifie que pour une maille plus fine, dont on pourrait attendre une précision meilleure, l'erreur maximale est plus grande.

Pourquoi une telle tendance est-elle observée ?
L'erreur calculée est composée de l'erreur d'approximation du Laplacian (on a négligé les restes lors des développements de Taylor), ainsi que d'une erreur qui n'était pas prise en compte précédemment : l'erreur machine. L'erreur due au laplacien est une erreur constante pour un même point considéré. Ainsi, l'erreur machine augmente avec la complexité des calculs.

La source de cette erreur est la représentation des nombres par la machine. En effet, il n'est pas possible de représenter des nombres réels sur machine. Ces nombres sont approximés et l'ordinateur utilise des nombres dits « flottants ». Cette partie d'estimation d'erreurs est à la source même du cours de MAO de Cyril Bredel et nous vous invitons alors à le consulter pour de plus amples détails.

Pour une maille fine, de nombreuses positions sont à calculer et les matrices deviennent vite très grandes car elles évoluent de manière inversement quadratique avec le pas de la membrane.

Par exemple, pour h=0.1, la matrice A est de taille 81 et il est facile de voir combien le nombre d'opérations est important. L'erreur devient alors vite trop grande et il est important de bien choisir la largeur des mailles avant de lancer les calculs.

On voit bien qu'il est primordial de limiter le nombre de calculs au maximum. Dans ce sens, on voit des études de déformation de membrane sur internet commencant leur étude

par un calcul global simple des positions de chaque point. Cela leur permet de voir l'allure générale de la déformation. Il leur est ensuite possible de considérer une partie plus restreinte de la membrane et de lancer des calculs sur cette zone. Cela est un peu compliqué à mettre en oeuvre car il faut toujours calculer certains points en dehors de la zone d'étude. En effet, si le calcul n'est fait que sur les bords de l'étude, ce calcul interprétera que les bords de la zone considérée sont fixés, ce qui n'est pas le cas.
Une telle méthode implique une discrétisation différente en différents points de la membrane, ce qui est difficile à mettre en oeuvre.

Mais pour l'instant, on peut dire que notre modèle de résolution est fiable pour des maillages relativement larges et que des erreurs interfèrent avec notre modèle pour des maillages fins.

L'ensemble des valeurs calculées permettant d'arriver aux résultats d'erreurs donnés plus haut sont disponibles en annexe.

Chapitre 2

Résolution avec le Bilaplacien

2.1 Résolution mathématiques

Nous tentons dans cette partie une nouvelle approche de cette déformation avec le bilaplacien, que l'on définit comme : $\Delta^2 u = \Delta(\Delta u)$.

En Bilaplacien, l'équation à résoudre devient :

$$c_2 \Delta^2 u = f$$

où u est le terme source et c_2 est le coefficient de résistance à la flexion. c_2 peut s'écrire sous la forme $\frac{Ee^3}{12(1-\nu)}$ où E est le module de Young du matériau de la membrane, e l'épaisseur de la membrane et ν le coefficient de poisson du matériau.

De la même manière que pour le laplacien, la membrane est discrétisée.

On travaille maintenant avec une nouvelle condition initiale qui est : $\frac{\partial u}{\partial n}|_\Omega = 0$.
On considère donc que la membrane est fixée à ses bords et que la membrane n'accepte pas de mouvement de torsion à ses bords (la direction initiale est fixée par un vecteur de coordonnée verticale nulle).

Afin de trouver une solution approchée de f, nous allons reprendre le travail effectué pour le laplacien simple. L'idée générale est de calculer $-\Delta U$, puis de remplacer U par $-\Delta u$ en se basant sur le fait que $-\Delta(-\Delta u)) = \Delta^2 u$.

$$-\Delta U = -(\tfrac{1}{h^2}(U_{x_{i+1},y_j} - 2U_{x_i,y_j} + U_{x_{i-1},y_j}) + \tfrac{1}{k^2}(U_{x_i,y_{j+1}} - 2U_{x_i,y_j} + U_{x_i,y_{j-1}}) = f$$

Maintenant, nous devons calculer un laplacien aux différents points U.

$$(\Delta^2 u)_{i,j} = \tfrac{1}{k^4} u_{i,j-2}$$
$$+\tfrac{2}{h^2k^2} u_{i-1,j-1} - (\tfrac{4}{h^2k^2} + \tfrac{4}{k^4}) u_{i,j-1} + \tfrac{2}{h^2k^2} u_{i+1,j-1} + \tfrac{1}{h^4} u_{i-2,j} - (\tfrac{4}{h^2k^2} + \tfrac{4}{k^4}) u_{i-1,j}$$
$$+(\tfrac{8}{h^2k^2} + \tfrac{6}{h^4} + \tfrac{6}{k^4}) u_{i,j}$$
$$-(\tfrac{4}{h^2k^2} + \tfrac{4}{k^4}) u_{i+1,j} + \tfrac{1}{h^4} u_{i+2,j}$$

$$+\frac{2}{h^2k^2}u_{i-1,j+1} - (\frac{4}{h^2k^2} + \frac{4}{k^4})u_{i,j+1} + \frac{2}{h^2k^2}u_{i+1,j+1}$$
$$+\frac{1}{k^4}u_{i,j+2}$$

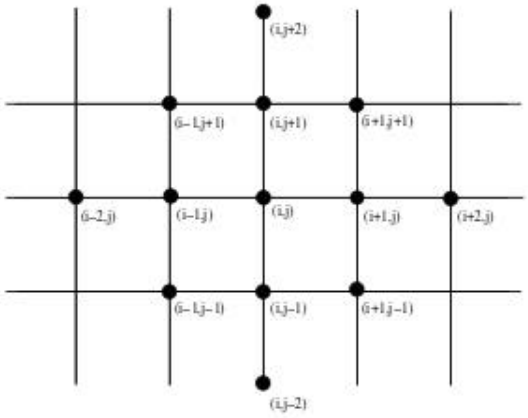
Nous posons $k = h$, dès lors, on obtient :

$$(\Delta^2 u)_{i,j} = \frac{1}{h^4}u_{i,j-2}$$
$$+\frac{2}{h^4}u_{i-1,j-1} - \frac{8}{h^4}u_{i,j-1} + \frac{2}{h^4}u_{i+1,j-1}$$
$$+\frac{1}{h^4}u_{i-2,j} - \frac{8}{h^4}u_{i-1,j}$$
$$+\frac{20}{h^4}u_{i,j}$$
$$-\frac{8}{h^4}u_{i+1,j} + \frac{1}{h^4}u_{i+2,j}$$
$$+\frac{2}{h^4}u_{i-1,j+1} - \frac{8}{h^4}u_{i,j+1} + \frac{2}{h^4}u_{i+1,j+1}$$
$$+\frac{1}{h^4}u_{i,j+2}$$

En factorisant, on obtient donc :

$$h^4(\Delta^2 u)_{i,j} = \frac{h^4 f}{c_2} = u_{i+2,j} + u_{i-2,j} + u_{i,j+2} + u_{i,j-2}$$
$$+2(u_{i+1,j+1} + u_{i+1,j-1} + u_{i-1,j+1} + u_{i-1,j-1})$$
$$-8(u_{i-1,j} + u_{i+1,j} + u_{i,j+1} + u_{i,j-1})$$
$$+20u_{i,j}$$

La forme obtenue comprend 13 termes, qui correspondent au point source et aux 12 points proches de la position à laquelle la force est calculée. Ceci est illustré par l'image ci-dessous.



De la même manière que pour le laplacien simple, représentons ces équations linéaires sous forme matricielle. A est la matrice des coefficients des équations linéaires, u est le vecteur des points de la grille et $b = \frac{h^4 f}{c_2}$.

Pour mieux visualiser la représentation matricielle, donnons des valeurs numériques aux paramètres du problème. Prenons, pour une matrice de taille 25*25 : $N = 8$, $c_2 = 1$, $h = \frac{1}{8}$ et $f = 4096$.

Les équations linéaires seront alors de la forme :

$$\begin{aligned}
&u_{i+2,j} + u_{i-2,j} + u_{i,j+2} + u_{i,j-2} \\
&+2(u_{i+1,j+1} + u_{i+1,j-1} + u_{i-1,j+1} + u_{i-1,j-1}) \\
&-8(u_{i-1,j} + u_{i+1,j} + u_{i,j+1} + u_{i,j-1}) \\
&+20u_{i,j} = 1
\end{aligned}$$

Construisons d'abord la matrice A en donnant des valeurs à i et j dans l'équation ci-dessus.

$$\begin{gathered}
i, j = 1, 1 \\
u_{3,1} + u_{1,3} + 2(u_{2,2} + u_{2,0} + u_{0,2} + u_{0,0}) - 8(u_{0,1} + u_{2,1} + u_{1,2} + u_{1,0}) + 20u_{1,1} = 1 \\
i, j = 2, 1 \\
u_{4,1} + u_{0,1} + u_{2,3} + 2(u_{3,2} + u_{3,0} + u_{1,2} + u_{1,0}) - 8(u_{1,1} + u_{3,1} + u_{2,2} + u_{2,0}) + 20u_{2,1} = 1 \\
i, j = 3, 1 \\
u_{5,1} + u_{1,1} + u_{3,3} + 2(u_{4,2} + u_{4,0} + u_{2,2} + u_{2,0}) - 8(u_{2,1} + u_{4,1} + u_{3,2} + u_{3,0}) + 20u_{3,1} = 1 \\
i, j = 4, 1 \\
u_{6,1} + u_{2,1} + u_{4,3} + 2(u_{5,2} + u_{5,0} + u_{3,2} + u_{3,0}) - 8(u_{3,1} + u_{5,1} + u_{4,2} + u_{4,0}) + 20u_{4,1} = 1 \\
i, j = 5, 1 \\
u_{7,1} + u_{3,1} + u_{5,3} + 2(u_{6,2} + u_{6,0} + u_{4,2} + u_{4,0}) - 8(u_{4,1} + u_{6,1} + u_{5,2} + u_{5,0}) + 20u_{5,1} = 1 \\
\dots
\end{gathered}$$

On obtient alors une matrice carrée tridiagonale par blocs de dimension $(N-3)^2 \geq 25$ pour $N \geq 8$ de la forme :

$$\begin{pmatrix}
G & K & I & 0 & 0 & \dots & \dots & \dots & 0 \\
K & G & K & I & \ddots & \ddots & \ddots & \ddots & \vdots \\
I & K & G & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\
0 & I & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\
0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots
\end{pmatrix}$$

Le bloc I est la matrice identité de dimension $N-3$. Les blocs G et K sont explicités ci-dessous :

$$G = \begin{pmatrix} 20 & -8 & 1 & 0 & 0 \\ -8 & 20 & -8 & 1 & 0 \\ 1 & -8 & 20 & -8 & 1 \\ 0 & 1 & -8 & 20 & -8 \\ 0 & 0 & 1 & -8 & 20 \end{pmatrix}$$

$$K = \begin{pmatrix} -8 & 2 & 0 & 0 & 0 \\ 2 & -8 & 2 & 0 & 0 \\ 0 & 2 & -8 & 2 & 0 \\ 0 & 0 & 2 & -8 & 2 \\ 0 & 0 & 0 & 2 & -8 \end{pmatrix}$$

2.2 Simulation numérique

Pour cette partie numérique, nous avons choisi, comme le groupe de l'an dernier, de résoudre ce système à l'aide de la méthode de Cholesky. Soit A notre matrice tri-diagonale par blocs, symétrique et de dimensions 25*25. Le principe de la méthode de Cholesky est le suivant :
- Déterminer une matrice L telle que A=LyLt.
- On écrit à nouveau le système en remplaçant A par LyLt.
- On décompose le système : Ly=B et Lt=y.

On résout notre système à l'aide des deux équations précédentes, par ordre d'indices décroissant pour la première et par ordre croissant pour la seconde.

Programme Pascal

Notre enseignant nous a donné le code utilisé l'année dernière. La partie d'initialisation de la matrice A a posé quelques soucis. Aussi, lors des calculs, une colonne de matrice se décalait et il fallait rectifier ce problème. Nous avons utilisé matlab pour calculer les résultats que l'on devait attendre du programme et cela nous a permis de le construire. Cependant, il n'est ici pas possible de changer la taille de la matrice A, comme ce fut fait pour le laplacien simple.

Pour ce qui est du fonctionnement du programme, le programme nous demande en premier la valeur des diagonales des blocs principaux. Dans un second temps, elle place les blocs de manière diagonale.

```
dimensions de la matrice
25
valeur diagonale principale du premier bloc
20
valeur deuxième diagonale du premier bloc
-8
valeur troisième diagonale du premier bloc
1
valeur première diagonale du deuxième bloc
-8
valeur deuxième diagonale du deuxième bloc
2

tapez 1 pour utiliser un vecteur b dont toutes les composantes valent 1, 2 pour choisir les composantes de b ou tapez 3 pour trouver x = 1
1

A =

 20 |  -8 |   1 |   0 |   0 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0
 -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0
  1 |  -8 |  20 |  -8 |   0 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0
  0 |   1 |  -8 |  20 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0
  0 |   0 |   0 |   0 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0
 -8 |   2 |   0 |   0 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0
  2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0
  0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0
  0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0 |   0
  0 |   0 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0 |   0
  1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   0 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   0
  0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0 |   0
  0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   0 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0 |   0
  0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1 |   0
  0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0 |   1
  0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   0 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0 |   0
  0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0 |   0
  0 |   0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   2 |   0
  0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8 |   0
  0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   2 |  -8
  0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0 |   0
  0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   1 |   0
  0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |  -8 |   0
  0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20 |   0
  0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |   1 |   0 |   0 |   0 |   2 |  -8 |   2 |   0 |   1 |  -8 |  20
```

On obtient le résultat suivant :

```
X=

 1.07276751127157E+000|
 2.29533383211475E+000|
 2.70978900176020E+000|
 1.89057792556917E+000|
-3.29769635996805E-001|
 1.94308910798951E+000|
 4.54174749287019E+000|
 5.45168099481384E+000|
 3.79361443023169E+000|
-1.16101143833986E+000|
 1.80874930790215E+000|
 5.20763727374475E+000|
 6.46147062730273E+000|
 4.53899599948896E+000|
-1.74446089200458E+000|
 9.50574918733496E-001|
 4.03758212119083E+000|
 5.23117335121700E+000|
 3.70194505934871E+000|
-1.55773417130370E+000|
 1.02896428286608E-001|
 1.83729578046063E+000|
 2.53817189111655E+000|
 1.86988827319729E+000|
-4.78370623921252E-001|
```

Programme Matlab

Nous avons de plus mis au point un court programme permettant une autre initialisation de notre matrice A, mais sous une syntaxe lisible avec Matlab.

```
A = [ 20  -8   1   0   0  -8   2   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  -8  20  -8   1   0   2  -8   2   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    1  -8  20  -8   0   0   2  -8   2   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   1  -8  20   0   0   0   2  -8   2   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0  20  -8   1   0   2  -8   2   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0
   -8   2   0   0  -8  20  -8   1   0   2  -8   2   0   0   0   1   0   0   0   0   0   0   0   0   0   0
    2  -8   2   0   1  -8  20  -8   1   0   2  -8   2   0   0   0   1   0   0   0   0   0   0   0   0   0
    0   2  -8   2   0   1  -8  20  -8   0   0   2  -8   2   0   0   0   1   0   0   0   0   0   0   0   0
    0   0   2  -8   2   0   1  -8  20   0   0   0   2  -8   2   0   0   0   1   0   0   0   0   0   0   0
    0   0   0   2  -8   2   0   0   0  20  -8   1   0   2  -8   2   0   0   0   1   0   0   0   0   0   0
    1   0   0   0   2  -8   2   0   0  -8  20  -8   1   0   2  -8   2   0   0   0   1   0   0   0   0   0
    0   1   0   0   0   2  -8   2   0   1  -8  20  -8   1   0   2  -8   2   0   0   0   1   0   0   0   0
    0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   0   0   2  -8   2   0   0   0   1   0   0   0
    0   0   0   1   0   0   0   2  -8   2   0   1  -8  20   0   0   0   2  -8   2   0   0   0   1   0   0
    0   0   0   0   1   0   0   0   2  -8   2   0   0   0  20  -8   1   0   2  -8   2   0   0   0   1   0
    0   0   0   0   0   1   0   0   0   2  -8   2   0   0   0  20  -8   1   0   2  -8   2   0   0   0   1
    0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   1   0   2  -8   2   0   0   0
    0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   1   0   2  -8   2   0   0
    0   0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   1   0   2  -8   2   0
    0   0   0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   1   0   2  -8   2
    0   0   0   0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   1   0   2  -8
    0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   1   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   1   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20  -8   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   2  -8   2   0   1  -8  20]

B = [ 1.15 ; 1.2 ; 1.3 ; 1.2 ; 1.15 ; 1.2 ; 3.26 ; 4.02 ; 3.26 ; 1.2 ; 1.3 ; 4.02 ; 5 ; 4.02 ; 1.3 ; 1.2 ; 3.26 ; 4.02 ; 3.26 ; 1.2 ; 1.15 ; 1.2 ; 1.3 ; 1.2 ; 1.15 ];

X = A\B
```

On obtient le résultat suivant :

```
X =

    1.0728
    2.2953
    2.7098
    1.8906
   -0.3298
    1.9431
    4.5417
    5.4517
    3.7936
   -1.1610
    1.8087
    5.2076
    6.4615
    4.5390
   -1.7445
    0.9506
    4.0376
    5.2312
    3.7019
   -1.5577
    0.1029
    1.8373
    2.5382
    1.8699
   -0.4784
```

2.3 Simulation physique

Nous allons tenter, dans cette partie, de valider l'utilisation du bilaplacien pour une membrane. Nous allons ensuite tenter de comparer les résultats obtenus avec ceux du laplacien simple.

La méthode employée ici est la même que pour le laplacien. Le seul manque est que la matrice A du programme est de taille unique. Cela nous impose donc un certain pas. Il sera alors impossible de faire différentes études pour différents maillages. Choisissons tout d'abord un fonction des hauteurs validant les conditions aux bords (membrane fixée et plaquée) :
$p(x, y) = x^2y^2(x-1)^2(y-1)^2$

Avec l'approximation $c_2\Delta^2 p = h^4 f$ (on donne $c_2 = 1$), on obtient :

$$288x^2y^2-288x^2y-288xy^2+288xy+24x^4+24y^4-48x^3-48y^3+72x^2+72y^2-48x-48y+8 = h^4 f$$

De manière analogue à la partie 1, on calcule le vecteur X à l'aide de notre nouveau programme.

Les résultats des calculs sont disponibles en annexe. Tout d'abord, remarquons que les hauteurs théorique des points de la membrane forment deux symétries horizontales. Ce caractère symétrique est perdu avec la matrice X correspondant à l'hypothèse du bilaplacien.

En observant les résultats obtenus, on remarque que les valeurs pratiques sont du même ordre de grandeur que les valeurs théoriques. Les erreurs sont du même style que pour le laplacien. On ne peut malheureusement pas se prononcer quand au comportement de l'erreur en fonction du pas. Il est sûr que celle-ci augmentera pour un maillage plus fin mais on ne peut se prononcer sur la manière dont elle va évoluer en fonction du pas (linéaire, quadratique, exponentielle...).

Ce que l'on peut dire du calcul avec le laplacien, c'est que son utilisation n'est pas justifiée par un besoin de précision accru. En effet, on a bien vu que les erreurs sont comparables et même légèrement supérieures à celles du laplacien. Cela peut être expliqué par le fait que, pour un même maillage, le laplacien double implique plus de composantes dans ses équations que le laplacien. On peut facilement dire que l'erreur machine sera alors mécaniquement plus grande. Il n'est ici pas possible de dire que tel ou tel modèle est plus précis.

Ce que l'on peut dire, en revanche, c'est que ces modèles sont utilisés pour des comportements de membrane bien différents, de par les équations mises en jeu ainsi que par les condition aux bords complètement différentes. Or, une membrane d'un matériau donné ne peut pas être classifiée dans une classe ou une autre. C'est pourquoi Mr Gleyse nous a mentionné le besoin futur d'étudier un modèle qui est la combinaison linéaire des deux précédents étudiés, c'est-à-dire $c_2\Delta^2 p - c_1\Delta p = f$

Ainsi, on sera en mesure de caractériser toute membrane en fonction des coefficients c_1 et c_2 et d'appliquer le modèle trouvé pour calculer des hauteurs de points de la membrane en ayant pour donnée la force appliquée à cette membrane.

Méthodologie et organisation du travail

Afin de mener ce projet à bien, nous avons commencé par effectuer des recherches mises en communs sur une fiche bibliographique. Ces recherches nous ont permis de mieux cerner le sujet et de comprendre les travaux déjà entreprise par le groupe de l'année d'avant. M. Gleyse nous a ensuite donné le programme du groupe de l'an dernier s'appuyant sur la méthode de Cholesky. Morgan s'est occupé de l'adapter au bilaplacien et de créer un programme qui génère une matrice 25*25. Il a aussi modifié le programme de Cholesky afin de pouvoir rentrer le second membre B, il a ensuite adapté ce programme en Matlab pour le laplacien et le bilaplacien. Mathieu et Ninon se sont intéressés à la modélisation mathématique. Mathieu a ensuite vérifié la validité du laplacien simple et du bilaplacien en comparant les résultats théoriques et expérimentaux. Laura et Ou ont récupéré le rapport latex du groupe de l'an dernier et ont commencé à rédiger le notre. Monsieur Gleyse nous a expliqué comment utiliser mapple afin de pouvoir modéliser notre membrane et la mettre en page de couverture. Nous nous sommes ensuite retrouvés en dehors des heures de cours pour avancer le projet et le rapport. Enfin l'aide et les explications de Mr Gleyse nous ont été d'une grande utilité pour l'aboutissement de ce projet.

Conclusion et perspectives

Durant ce projet, nous avons axé notre étude du projet suivant trois axes : mathématique, numérique et physique. Nous avons vérifié le modèle de l'année passée et en avons proposé un nouveau. Nous avons tenté à chaque fois de justifier la validité de chaque modèle. Nous avons constaté que les deux modèles ne se différencient non pas par leur précision, mais bien par la nature du matériau employé dans la membrane ainsi que des conditions dans lesquelles cette membrane est placée.

Nous allons maintenant donner quelques pistes pour un futur projet. Deux axes de progression sont possibles. On pourra dans un premier temps tenter d'effectuer le même travail qu'expliqué ici, mais de l'appliquer à une nouvelle équation combinaison des deux que l'on a étudiées. Ensuite, et cela sera sans doute très compliqué, il faudrait essayer de tenter une discrétisation de la membrane différente en fonction de la zone que l'on souhaite étudier (voir figure 1 en annexe 1).

Ce projet physique a nécessité l'apport de tous dans un travail qui suivait différentes pistes. Ce travail a fait appel à des connaissances variées, aussi bien en mathématiques (développement de Taylor, matrices, dérivées d'ordre 4) qu'en informatique (maîtrise du langage Pascal). Ce projet nous a permis de découvrir certaines méthodes mathématiques et nous a aussi introduit au logiciel Maple, très utile pour la visualisation de l'allure de la membrane et pour le le calcul de fonctions de deux variables en de nombreux points. Le sujet de déformation de membrane à bien sûr aussi été source de curiosité scientifique, permettant à plusieurs élèves de spécialités différentes de découvrir une application physique et concrète d'une formule capable d'être utilisée en résistance des matériaux.
Au delà des connaissances scientifiques, ce projet nous a été profitable sur le plan personnel et humain. Il nous as permis de développer des capacités d'organisation, de communication et de discipline, qualités indispensable à tout ingénieur.

Bibliographie

[1] NOM, Prénom *Titre*, edition, date.

[2] NOM DES AUTEURS, "*Titre de l'article*", Titre du journal, Volume, pages, année.

[3] http://www-ljk.imag.fr/membres/Guillaume.James/sujet_TP_MN_09.pdf (Valide à la date du 04/04/2008)

bibitemLien Internet 01 http://www.phys.ens.fr/~dormy/Palais/node1.html (Valide en janvier 1996)

bibitemLien Internet 01 http://www.ecf.utoronto.ca/~bentz/mhome.shtml (Valide en 2012) http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.49.79&rep=rep1&type=pdf (Date de validité indéterminable)

https://www.ljll.math.upmc.fr/~ledret/M1/ComplementsM1ApproxEDP.pdf (Valide en janvier 2009)

http://www.ufrmeca.univ-lyon1.fr/~buffat/COURS/COURSDF_HTML/node32.html (Valide en avril 2008)

http://www.math.iit.edu/~fass/Notes461_Ch7.pdf(Valide en automne 2011)

http://www-solar.mcs.st-and.ac.uk/~alan/MT3601/Fundamentals/node60.html (Valide en juin 2000)

http://math.univ-lyon1.fr/~saleh/Docs/MEF/CoursUFE_KhaledSaleh.pdf (Valide en décembre 2011)

http://www.ufrmeca.univ-lyon1.fr/~buffat/COURS/COURSDF_HTML/node32.html (Valide en avril 2008)

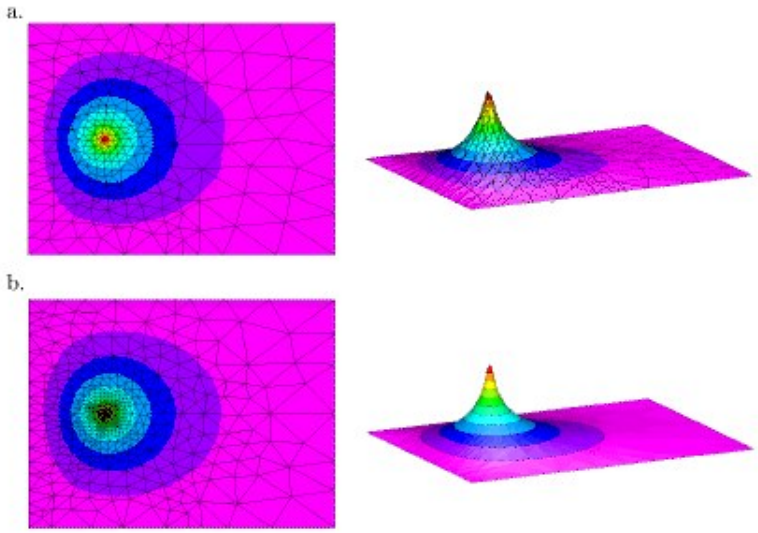
http://didel.script.univ-paris-diderot.fr/claroline/work/user_work.php?cmd=exDownload&authId=1038&assigId=1&workId=11&cidReset=true&cidReq=3641XXXXPHYNUMPMA (Valide en février 2014)

https://books.google.fr/books?id=7NVu0W7CDaIC&pg=PA152&lpg=PA152&dq=bilaplacien+membrane&source=bl&ots=y6tUCUHCHH&sig=gBMoZPlQ66LVULCwPLqUlubtp30&hl=fr&sa=X&ved=0ahUKEwiSjda-99PLAhUJ2RoKHX2KDCUQ6AEINzAD#v=onepage&q=bilaplacien%20membrane&f=false (Valide en février 2005)

http://math.univ-lyon1.fr/~herbach/download/Rapport_stage_LATP.pdf (page 50) (Valide en 2013)

Annexe A

Discrétisation locale plus importante



Annexe B

Propositions de sujets de projets (en lien ou pas avec le projet réalisé)

Annexe C

Codes

Trois programmes ont été utiles dans le cadre de ce projet :

— genMatLab2.pas
— deformationMembrane.pas
Les codes de ces programmes sont joints ci-dessous.

```
program genMatlab2;


const q=100 ;
Type matrice = array [1..q,1..q] of Real; matrice2 = array [1..q] of Real ;
var A : matrice; i,j,p, rang : Integer ; valeur, valeur2 : Real;

procedure initialiserMatrice(dim : integer ; var A : Matrice);
var i, j : Integer;
begin
 For i:= 1 to dim do
  For j := 1 to dim do
   A[i, j] := 0;
end;

procedure affichage_matrice (mat: matrice);
var i,k:Integer ;
begin
 For i:= 1 to p do
  begin
   writeln();
   For k:=1 to p-1 do
    write(mat[i,k]:4:1,' | ') ;
   write(mat[i,p]:4:1) ;
  end;
end;

procedure affichage_matrice2 ( mat: matrice2);
var k:Integer ;
begin
   writeln();
   For k:=1 to p do
```

```
    writeln(mat[k],'|') ;
end;

begin

 writeln('Rang : ');
 readln(p);

 initialiserMatrice(p, A);

 write('Valeur pour la diagonale du bloc central : ');
 readln(valeur);
 For i := 1 to p do
  A[i, i] := valeur;

 write('Valeur pour les diagonales sup et inf du bloc central : ');
 readln(valeur2);


 For i := 1 to p do
  if (i MOD 3) <> 0 then
   begin
    A[i + 1, i] := valeur2;
    A[i, i + 1] := valeur2;
   end
  else
   A[i, i] := valeur;

 write('Valeur pour les diagonales des blocs sup et inf : ');
 readln(valeur);

 rang := 4;
 For i := 1 to (1 + p - rang) do
  begin
   A[i, rang + i - 1] := valeur;
   A[rang + i - 1, i] := valeur;
  end;


 writeln();
 write('[');
 For i := 1 to p do
  begin
   For j := 1 to p do
    write(A[i, j]:2:2, ' ');
   If i < p then
    write('; ');
  end;
 write(']');
```

```
end .
```

```
program deformationMembrane;

const q=100 ;
Type matrice = array [1..q,1..q] of Real; matrice2 = array [1..q] of Real ;
var A,L,Lt : matrice; b,Y,X : matrice2; i,j,m,p, rang: Integer ; K, N, valeu

procedure initialiserMatrice(dim : integer ; var A : Matrice);
var i, j : Integer;
begin
 For i:= 1 to dim do
  For j := 1 to dim do
   A[i, j] := 0;
end;

procedure affichage_matrice (mat: matrice);
var i,k:Integer ;
begin
 For i:= 1 to p do
  begin
   writeln();
   For k:=1 to p-1 do
    write(mat[i,k]:4:1,' | ') ;
   write(mat[i,p]:4:1) ;
  end;
end;

procedure affichage_matrice2 ( mat: matrice2);
var k:Integer ;
begin
   writeln();
   For k:=1 to p do
    writeln(mat[k],'|') ;
end;


begin
 writeln('Rang de la matrice? rang ',q,' maximum et entier') ;
 readln(p);

 initialiserMatrice(p, A);


 write('Valeur pour la diagonale du bloc central : ');
 readln(valeur);
 For i := 1 to p do
  A[i, i] := valeur;

 write('Valeur pour les diagonales sup et inf du bloc central : ');
 readln(valeur2);
```

```
For i := 1 to p do
 if (i MOD 3) <> 0 then
  begin
   A[i + 1, i] := valeur2;
   A[i, i + 1] := valeur2;
  end;


write('Valeur pour les diagonales des blocs sup et inf : ');
readln(valeur);
rang := 4;
For i := 1 to (1 + p - rang) do
 begin
  A[i, rang + i - 1] := valeur;
  A[rang + i - 1, i] := valeur;
 end;            {Les 3 saisies precedentes permettent de creer une matrice

writeln('tapez 1 pour taper le vecteur b');
readln(i);
if i=1 then
 for j:=1 to p do
  begin
   readln(K);
   b[j]:=K;
  end
else
 for j:=1 to p do
  begin
   readln(K);
   X[j]:=K;
  end;
{calcul de B tq AX=B}
{ for i:=1 to p do
  begin
   b[i]:=0;
   for j:=1 to p do
   b[i]:=b[i]+A[i,j]*X[j]
  end;}


writeln();
writeln('A = ');
affichage_matrice(A);
writeln();

writeln('X=');
affichage_matrice2(X) ;
writeln();
```

```
writeln();
writeln('b = ');
affichage_matrice2(b) ;
writeln() ;


For i := 1 to p do
 For j := 1 to p do
  L[i,j] := 0 ;   { initialisation de la matrice L triangulaire inferieur}


affichage_matrice( A);

L[1,1] := sqrt(A[1,1]);

affichage_matrice( L);

For i := 2 to p do
 For j := 1 to i do
  begin

  If j>1 then
   begin
    K:=0 ;
    N:=0 ;
    For m := 1 to j-1 do
     begin
      K := K + sqr(L[j,m]) ;
      N := N + L[i,m]*L[j,m] ;
     end;
    If j = i then
     begin
      L[j,j] := sqrt( A[j,j] - K);
     end
    Else
     if L[j, j] <> 0 then
      L[i,j] := (A[i,j] - N)/(L[j,j])
     else
      writeln('ERREUR : DIVISON PAR ZERO');
   end
  Else
   begin
    If   j = i  then
    L[j,j] := sqrt( A[j,j])
    Else
     if L[j, j] <> 0 then
      L[i,j] := (A[i,j])/(L[j,j])
     else
      writeln('ERREUR : DIVISON PAR ZERO');
   end;
```

```
    end;

            {calcul de la matrice L avec les formules}
 writeln('L = ');
 affichage_matrice( L);
 writeln();

 For i := 1 to p do
  For j := 1 to p do
   Lt[i,j] := L[j,i];   {Calcul de la transposee de L => Lt}
 writeln();
 writeln('Lt = ');
 affichage_matrice(Lt);
 writeln();

 Y[1] := b[1]/L[1,1] ;
 For i:= 2 to p do
  begin
  K := 0 ;
  For j := 1 to i-1 do
   K := K + L[i,j]*Y[j] ;
  Y[i] := (b[i]-K)/L[i,i] ;
  end;
 writeln();
 writeln('Y = ');
 affichage_matrice2(Y) ;
 writeln()    ;  {Calcul de Y tel que L.Y=b}


 X[p] := Y[p]/Lt[p,p] ;

 For i := p-1 downto 1 do
  begin
   K := 0 ;
   For j := i to p do
    K := K + Lt[i,j]*X[j] ;
   X[i] := (Y[i]-K)/(Lt[i,i]);
  end;      {Calcul de X tel que Lt.X=Y}
     begin
      y.I:=-y.I+2046;
      y.I := y.I shl 52;
      z.I:=(x.I and $000FFFFFFFFFFFFF) or $3FF0000000000000;


 writeln('X=');
 affichage_matrice2(X) ;
 writeln();
 readln();
end.
```