Algorithmique numérique - Projet 2 : Résolution de systèmes linéaires

Yannis YOASSI PATIPE (Coordinateur),
Ilyes BECHOUAL (Secrétaire),
Louis PEYRONDET (Programmeur),
Robin ROUET (Programmeur),
Youssef MAHJOUB (Programmeur)

Mars 2023

1 Introduction

L'étude des systèmes linéaires est une partie très importante en algorithmique numérique, en effet un très grand nombre de systèmes et d'équation fondamentales (comme celle de la chaleur) sont linéaires et un ingénieur informatique doit être en mesure de "maitriser les bases du linéaire" pour modéliser ce qui l'entoure. Dans ce projet plusieurs sujets sont traités tel que les méthodes de factorisation matricielles de Cholesky ou bien la méthode du gradient conjugué, afin de les utiliser pour résoudre numériquement l'équation linéaire de la chaleur.

2 Description algorithmique

2.1 Décomposition de Cholesky

2.1.1 Cholesky complet

La factorisation de Cholesky est une méthode de décomposition matricielle qui permet de factoriser une matrice symétrique définie positive A en un produit de deux matrices triangulaires telles qu'elles sont transposées l'une de l'autre(T et sa transposée). Cette factorisation est utile dans la résolution de systèmes linéaires.

L'algorithme de Cholesky calcule les coefficients de T selon les équations fournies dans l'énoncé. il est de complexité $\theta(n^3)$ dans le pire des cas.

Cette méthode est utilisée pour résoudre des systèmes linéaires de la forme A.x = b, où A est une matrice symétrique définie positive. La résolution d'une équation matricielle A.x = b coûte $\theta(n^3)$.

Parcontre, il convient de souligner que l'algorithme ne fonctionne pas lorsqu'il s'agit de travailler avec des matrices qui ne sont plus définies positives. En effet, la racine carrée n'est plus effectuée avec des chiffres négatifs pour obtenir les coefficients de la matrice T au cours de la première équation. Il en résulte de nombreuses erreurs.

2.1.2 Cholesky incomplet

L'algorithme de Cholesky incomplet est similaire à son homologue complet, la seule différence est que la boucle passe à l'itération suivante lorsque le terme est nul.

2.1.3 Cholesky et conditionnement

Après avoir implémenté ces deux méthodes de décomposition, elles ont été utilisé pour étudier le conditionnement. Le conditionnement est une notion qui permet d'évaluer l'instabilité numérique d'une matrice, et pour étudier cela il est nécessaire d'avoir un moyen de comparer les différents conditionnements. Dans

le code c'est la fonction : fonction comp_cond(A : Matrice) : Liste = [conditionnement avec A, conditionnement avec cholesky complet, cholesky incomplet] qui s'en occupe et qui permet d'avoir une vue d'ensemble sur les différentes valeurs de conditionnements.

2.2 Méthode du gradient conjugué

Afin d'obtenir le résultat du système d'équations linéaires $A \cdot x = b$, il est possible d'utiliser une méthode appelée "Méthode du gradient conjuguée", qui va être d'implémentée dans la suite de cette partie.

Dans un premier temps, un algorithme codé dans le langage Matlab que voici est mis à disposition :

```
function x = conjgrad(A, b, x)
    r = b - A * x;
    p = r;
    rsold = r' * r;

    for i = 1:length(b)
        Ap = A * p;
        alpha = rsold / (p' * Ap);
        x = x + alpha * p;
        r = r - alpha * Ap;
        rsnew = r' * r;
        if sqrt(rsnew) < 1e-10
            break
        end
        p = r + (rsnew / rsold) * p;
        rsold = rsnew;
    end
end
```

Mais est-il correct ?

2.2.1 Correction syntaxique

Les standards de base tels que des noms de variables explicites et des commentaires expliquant le déroulé de la fonction manque dans cette implémentation. Tout cela influe doncT dans la lisibilité du code, montrant que le formatage de ce code est perfectible également.

Ainsi, afin de rendre le code correct, il est important de donner des noms clairs aux variables pour comprendre plus aisément son comportement (par exemple, au lieu d'utiliser r, il est préférable de l'appeler *residu*, son nom complet). De plus, des commentaires constructifs apporteraient également des informations supplémentaires à sa compréhension.

2.2.2 Comparaison avec la Décomposition de Cholesky

Précédemment, il a été discuté d'une autre méthode de résolution, la décomposition de Cholesky, qui diffère grandement de la méthode du gradient conjugué.

En effet, elles diffèrent par leur type d'implémentation, puisque la décomposition se base sur le principe de factorisation de manière récursive, alors que la méthode du gradient conjugué est une méthode itérative qui converge vers le résultat en un nombre d'étapes bien fini, en se basant sur la structure de la matrice.

La décomposition de Cholesky est une méthode qui résout un système d'équation linéaire avec une complexité en $O(n^3)$. La méthode du gradient conjugué quant à elle, permet de gagner beaucoup de temps, notamment si la matrice choisie est creuse. Ainsi, en général, la compléxité de la méthode du gradient conjugué est de l'ordre de $O(n^2)$.

2.2.3 Algorithme de la méthode du gradient conjugué avec ou sans préconditionneur en python

Il existe deux manières d'implémenter la méthode du gradient conjugué. La première se base sur la définition même de la méthode, et la seconde dites "avec preconditionneur", qui va utiliser une matrice subsidiaire P qui, en réduisant le conditionnement de la matrice, va permettre de converger plus rapidement vers la solution. Cette matrice P est définie comme étant une approximation de la matrice inverse de la matrice A (la matrice à résoudre).

Ces deux méthodes ont donc été implémentée dans le langage python, en se basant sur leurs définitions et sur le programme en Matlab évoqué au début de cette partie. ($Cf. partie2.py$)

2.3 Application à l'équation de la chaleur

La partie application des concepts vue en cours et dans les précédentes parties a été au début très compliquée car le sujet manquait de précision, d'une part sur comment réaliser et surtout sur les objets à manipuler.
Mais après de longs échanges avec les intervenants, une trajectoire pour réaliser l'implémentation de l'équation de la chaleur et de sa résolution a été définie. L'équation à résoudre est $A \cdot T = F$ où T est l'inconnue, soit le champ de température après émission d'un flux F.
Le champ étant de 2 dimensions modélisé par une matrice $N * N$, l'énoncé a choisi de la transformer en une matrice colonnes N^2. Ce choix est très important et il est en relation avec la partie précédente sur la résolution par la méthode du gradient conjugué. La méthode du gradient conjugué va donc être utilisée pour résoudre le problème. Avant cela, l'énoncé définit la matrice A comme le produit de $1/h^2 \cdot Mc$ avec $h = 1/N + 1$ c'est donc la distance entre deux points adjacents et la matrice Mc est définie comme la figure suivante :

Pourquoi la matrice est elle définie de la sorte ? Pour deux raisons, la première était pour que la matrice soit à diagonale dominante, condition suffisante (mais

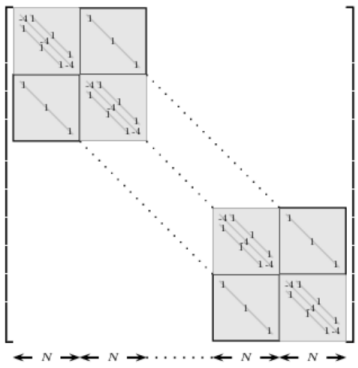

FIGURE 1 – Matrice Mc

pas nécessaire) pour que $M_c \in \mathrm{S}_{n++}$. La deuxième est car elle correspond à une discrétisation de l'opérateur laplacien présent dans l'équation de l a chaleur. Ainsi, il suffisait de créer ces variables, de prendre un flux F, de résoudre l'équation et de l'afficher grâce au module matplotlib.pyplot sous forme de différence de chaleur. Voici quelques exemples avec des flux différents :

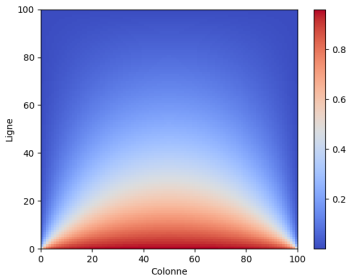

FIGURE 2 – Un flux de chaleur partant du bas.

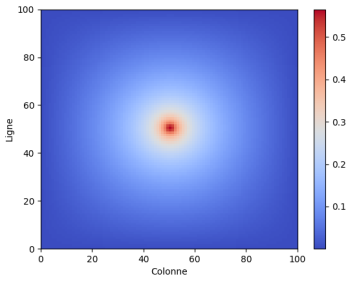

FIGURE 3 – Un flux de chaleur partant du centre.

La variation de température et la chaleur se propageant de proche en proche comme le décrit l'équation de la chaleur peut être observé.

3 Analyse des résultats

3.1 Cholesky et effet sur le conditionnement

Le conditionnement est une notion très importante en algorithmique numérique. En effet plus le conditionnement est éléve plus il y aura des instabilités numériques. Une technique pour diminuer le conditionnement est d'utiliser un préconditionneur

tel que cond(M^{-1} A) soit inférieur à cond(A) où M^{-1} est très proche de A^{-1}. Dans ce projet les résultats de cette méthode ont testé puis comparé avec M issue de Cholesky complet, M issue de Cholesky incomplet et A sans préconditionnement. Les résultats montrent que les conditionnements calculés avec les matrices issues de la factorisation de Cholesky son plus faible que cond(A). Ce qui vient encore renforcer la propriété de l'exercice qui dit que les méthodes de Cholesky ont un rôle important dans la stabilité numérique.

4 Commentaires

4.1 Youssef MAHJOUB

La partie mathématique ne me cause aucun problème. Ma préoccupation est toujours liée au codage. Quoi qu'il en soit, j'ai appris des choses intéressantes qui ont enrichi mon savoir-faire.

4.2 Robin ROUET

N'étant pas très à l'aise avec les problèmes mathématiques de ce niveau, j'apprehendai un peu de faire ce TD. Cependant, grace a l'aide de mes camarades et des encandrants, j'ai pu mieux comprendre le sujet et avancer au même rythme qu'eux. De plus, le fait d'implémenter des formules liées à la physique et d'avoir un apercu visuel de ce qu'elle représentent était vraiment une expérience enrichissante.

4.3 Louis PEYRONDET

Les notions de calcul matriciel étant nouvelle pour moi, ce projet m'a permis de commencer à développer ces compétences grâce aux autres membres de mon équipe qui ont su m'aider en répondant à mes questions. De plus le sujet du projet était engageant et les visualisations obtenues permettent de bien comprendre les résultats.

4.4 Ilyes BECHOUAL

Ce projet était plus difficile, mais beaucoup plus intéressant, notamment la partie application. Le cours était un support pour l'application des méthodes, mais c'est surtout dans des cas concrets que nous avons pu retenir les enseignements de manière plus efficace. La répartition du travail s'est bien déroulée, et chaque membre de l'équipe a pu contribuer de manière significative au projet. La cohésion était essentielle, car les différentes parties du projet n'étaient pas indépendantes. Nous avons dû nous coordonner, nous entraider et nous faire confiance lors de la programmation.

4.5 Yannis YOASSI PATIPE

j'ai trouvé ce deuxième projet très intéressant. Premièrement car il est toujours très agréable et plus facile de travailler en groupe (partage des idées, des méthodes, des astuces etc.) . Deuxièmement car dans ce projet nous n'avons pas codé pour coder, mais nous avons codé pour comprendre et modéliser un phénomène. Grâce à ce projet nous avons pu assurer le passage de l'abstrait (équation de la chaleur, gradient, formule) au concret/visible (graphe du flux de chaleur etc.) et c'est ce qui m'a vraiment plu.