Rapport de projet Mathématiques.

December 14, 2022

Sommaire

Contents

1 Introduction

1.1 Motivation

1.2 Application théorique

Le but est de résoudre l'équation linéaire de matrice : $AX = B$ dans $\mathbb{R}^N$ sans avoir à calculer A^{-1}. Car calculer A^{-1} peut poser problème lorsque N devient grand. En effet, pour $N = 20$, il faudra 5 fois l'age de l'univers pour calculer A^{-1}.

Nous allons donc à l'aide de plusieurs méthodes, tenter de simplifier le problème en sous-problèmes plus simple à résoudre et procéder à des optimisations graduelles afin d'optimiser le cout en temps de calcul et l'espace mémoire utilisé pour le cacul.

2 Références

2.1 Bibliographie

References

2.2 Biographies

2.2.1 Gauss

Avant d'étudier la méthode de Gauss, il peut être pertinent d'établir une courte biographie de son auteur. Surnommé “prince des mathématiques” par ses pairs, ce mathématicien, né le 30 avril 1777, a contribué de bien des manières à développer non seulement les mathématiques mais également les méthodes en astrophysique et en électromagnétisme. Johann Carl Friedrich Gauss a pu se pencher sur des problèmes dits classiques (depuis l'antiquité) en adoptant des méthodes et raisonnements modernes. Il démontre alors le théorème fondamental de l'algèbre que l'on connaît aujourd'hui sous le nom du théorème de d'Alembert-Gauss, il dédie également un ouvrage à la théorie des nombres contenant plusieurs démonstrations qui révolutionnent l'arithmétique, on lui doit en partie la forme actuelle des nombres complexes. En astrophysique, il met au point la méthodes des moindres carrés permettant de minimiser les incertitudes dues aux mesures ce qui lui permet de déterminer exactement la position de Ceres, une planète naine du système solaire. Avec la contribution de Wilhem Weber, Gauss formule deux théorèmes essentiels en électromagnétisme réfutant l'existence de monopôle magnétique et établissant une relation entre le flux d'un champ électrique sur une surface fermée et la charge électrique totale à l'intérieur de cette surface. Cette liste est loin de résumer tous ses travaux, dont une partie a été publiée après son décès le 23 février 1855, ces derniers sont très nombreux et ont servi de base de recherche pour d'autres mathématiciens et physiciens après lui. Il est donc intéressant d'étudier la solution apportée par Gauss et de l'appliquer à notre problématique. L'élimination de Gauss est expliquée ci-après.

2.2.2 Choleski

<Faire la biographie de Choleski>

3 Travail préliminaire

3.1 Gestion des matrices

Tout d'abord, en C, commme la plupart des langages de programmation, il est impossible d'exprimer l'intégralité des réels, entiers, complexes ... car il faudrait disposer d'une mémoire illimitée.

les concepts de matrice et de vecteur en C ne sont pas implémenté directement dans le langage. Cependant, nous pouvons nous servir des listes et tableau en C pour imiter le comportement d'une matrice.

Pour les vecteurs, l'allocation se faire à l'aide de la fonction "Allocation_Vecteur" qui contient le code suivant :

```
d
```

4 Méthodes directes

4.1 Méthode triangulaire inférieure

Méthode

On chercher le vecteur X de taille N tel que $AX = B$, où A sera une matrice triangulaire inférieure carré de de taille N.

La formule générale pour les termes de X est : $x_i = \frac{b_i - \sum_{j=1}^{i-1} a_{ij}x_j}{a_{ii}}$, $i = 1, \ldots, N$.

Exemple

Soit $N = 3$:

$$\begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \Leftrightarrow \left\{ \begin{array}{c} a_{11}x_1 = b_1 \\ a_{21}x_1 + a_{22}x_2 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3 \end{array} \right.$$

$$\Leftrightarrow \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{22}} - \frac{a_{21}}{a_{22}}x_1 \\ \frac{b_3}{a_{33}} - \frac{a_{31}}{a_{33}}x_1 - \frac{a_{32}}{a_{33}}x_2 \end{pmatrix} = \begin{pmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{22}} - \frac{a_{21}}{a_{22}}x_1 \\ \frac{b_3 - \sum_{k=1}^{2} a_{3k}x_k}{a_{33}} \end{pmatrix}$$

$x_i = \frac{b_i - \sum_{j=1}^{i-1} a_{ij}x_j}{a_{ii}}$, $i = 1, \ldots, N$.

Algorithme

Algorithme de la remontée :

Soit A une matrice carrée de taille N, X et B un vecteur de taille N.
pour i = 1 à N faire
sum = 0
pour j=1 à i - 1 faire
sum=sum+A[i][j]*x[j]
fin pour
x[i] = (b[i] - sum)/Aii
fin pour

4.2 Méthode triangulaire supérieure

Méthode

On chercher le vecteur X de taille N tel que $AX = B$, où A sera une matrice triangulaire supérieure carré de de taille N.

Ainsi, de manière de analogue à résolution de matrice inférieure, la formule générale pour les termes de X est : $x_i = \frac{b_i - \sum_{j=1}^{i-1} a_{ij}x_j}{a_{ii}}$, $i = 1, \ldots, N$.

Exemple

Soit $N = 3$:

$$\begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \Leftrightarrow \left\{ \begin{array}{c} a_{11}x_1 = b_1 \\ a_{21}x_1 + a_{22}x_2 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3 \end{array} \right.$$

$$\Leftrightarrow \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{22}} - \frac{a_{21}}{a_{22}}x_1 \\ \frac{b_3}{a_{33}} - \frac{a_{31}}{a_{33}}x_1 - \frac{a_{32}}{a_{33}}x_2 \end{pmatrix} = \begin{pmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{22}} - \frac{a_{21}}{a_{22}}x_1 \\ \frac{b_3 - \sum_{k=1}^{2} a_{3k}x_k}{a_{33}} \end{pmatrix}$$

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} a_{ij}x_j}{a_{ii}}, \; i = 1, \ldots, N.$$

Algorithme

`Ecrire algorithme ici`

4.3 Elimination de Gauss

Méthode

On cherche a résourde $AX + B$, on va chercher à les transformer l'expression en $UX = e$

Conclusion

Cette méthode est relativement efficace, cependant, il faut reclaculer totalement les matrices U et e si B change, ce qui est couteux en temps de calcul.

4.4 Factorisation "LU"

On factorise A en une matrice triangulaire supérieure qu'on appele U et une matrice triangulaire inférieure qu'on appele L tel que $A = LU$. Cette opération est réalisé à l'aide de la fonction "LU". La diagonale de L sera rempli de 1 afin de faciliter la factorisation.

Comme on a $A = LU$ et $AX = B$, on peut en déduire que : $LUX = B$.

Ensuite, on cherche à obtenir la matrice Y $Y = UX$ tel que $LY = B$ avec la fonction "Sol_Inf".

Enfin, on peut trouver X avec $UX = Y$ avec la fonction "Sol_Sup".

Code

```
void LU(double **L, double **A, int Taille)
{
    Nettoyer_Matrice(L, Taille, Taille);

    // Rempli les 1 en diagonale
    for (int i = 0; i < Taille; i++)
    {
        L[i][i] = 1;
    }

    // Calcule L et U
    for (int i = 0; i < Taille - 1; i++)
    {
        for (int k = i + 1; k < Taille; k++)
        {
            double C = A[k][i] / A[i][i];

            L[k][i] = C;

            for (int j = 0; j < Taille; j++)
            {
                A[k][j] = A[k][j] - (C * A[i][j]);
            }
        }
    }
}
```

Conclusion

4.5 Factorisation de Cholesky

Méthode

Soit la matrice A qui est :

- Symétrique : c'est à dire que $A = A^T$.
- Définie positive, c'est à dire quelle est positive et inversible, tel que $\langle AY, Y\rangle > 0, \forall Y \in \mathbb{R}^N \setminus \left\{ \overrightarrow{0} \right\}$

Alors, on a : $A = L \cdot L^T$ ou $L = \begin{pmatrix} l_{11} & 0 & 0 \\ \vdots & \ddots & 0 \\ l_{1N} & \cdots & l_{NN} \end{pmatrix}$ avec $L_{ii} > 0$ pour $i = 0, \ldots, N$ (une matrice triangulaire inférieure dont la diagonale est positive).

On a donc $A = L \cdot L^T$ et $AX = B$, si on remplace A, $L \cdot L^T X = B$ et simplifier avec $LY = B$ ou $Y = L^T X$. On peut donc résoudre $LY = B$ avec la fonction "Sol_Inf" et efin résoudre $L^T X = Y$ avec la fonction "Sol_Sup".

Exemple

Soit L une matrice de taille $N * N$ ou $N = 4$

On créer une matrice L tel que $L \cdot L^T = A$, on a :

$$\underbrace{\begin{pmatrix} l_{11} & 0 & 0 & 0 \\ l_{21} & l_{22} & 0 & 0 \\ l_{31} & l_{32} & l_{33} & 0 \\ l_{41} & l_{42} & l_{43} & l_{44} \end{pmatrix}}_{L} \underbrace{\begin{pmatrix} l_{11} & l_{21} & l_{31} & l_{41} \\ 0 & l_{22} & l_{32} & l_{42} \\ 0 & 0 & l_{33} & l_{43} \\ 0 & 0 & 0 & l_{44} \end{pmatrix}}_{L^T} = \underbrace{\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}}_{A}$$

$$\Leftrightarrow \begin{pmatrix} l_{11}^2 & l_{11}l_{21} & l_{11}l_{31} & l_{11}l_{41} \\ l_{21}l_{11} & l_{21}^2 + l_{22}^2 & l_{21}l_{31} + l_{22}l_{32} & l_{21}l_{41} + l_{22}l_{42} \\ l_{31}l_{11} & l_{31}l_{21} + l_{32}l_{22} & l_{31}^2 + l_{32}^2 + l_{33}^2 & l_{31}l_{41} + l_{32}l_{42} + l_{33}l_{43} \\ l_{41}l_{11} & l_{41}l_{21} + l_{42}l_{22} & l_{41}l_{31} + l_{42}l_{32} + l_{43}l_{33} & l_{41}^2 + l_{42}^2 + l_{43}^2 + l_{44}^2 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

On a donc pour les termes en diagonale :

- $l_{11}^2 = a_{11} \Rightarrow l_{11} = \sqrt{a_{11}}$
- $l_{21}^2 + l_{22}^2 = a_{22} \Rightarrow l_{22} = \sqrt{a_{22} - l_{21}^2}$
- $l_{31}^2 + l_{32}^2 + l_{33}^2 = a_{33} \Rightarrow l_{33} = \sqrt{a_{33} - l_{31}^2 - l_{32}^2}$
- $l_{41}^2 + l_{42}^2 + l_{43}^2 + l_{44}^2 = a_{44} \Rightarrow l_{44} = \sqrt{a_{44} - l_{41}^2 - l_{42}^2 - l_{43}^2}$

On constate que : $l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}$

On a également pour les autres termes :

- $l_{21}l_{11} = a_{21} \Rightarrow l_{21} = \frac{a_{21}}{l_{11}}$
- $l_{31}l_{11} = a_{31} \Rightarrow l_{31} = \frac{a_{31}}{l_{11}}$
- $l_{31}l_{11} = a_{41} \Rightarrow l_{41} = \frac{a_{31}}{l_{11}}$

On constate que $l_{ij} = \frac{a_{ij}}{l_{jj}} - \sum_{k=1}^{j-1} l_{ik}l_{jk}$

Algorithme

```
Ecrire code
```

Conclusion

5 Méthodes itératives

Contrairement aux méthodes directes précédents où l'on cherche directement le vecteur X pour résoudre $AX = B$. Ici, nous allons nous intéresser.

5.1 Méthode de Jacobi

Méthode

Soit A une matrice carrée de taille N et B un vecteur de taille N, on cherche toujours à résoudre l'équation $AX = B$.

On peut décomposer A en 3 matrices tel que $A = E + F + G$:

- D la matrice contenant la diagonnale de A : $D = \begin{pmatrix} A_{11} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & A_{nn} \end{pmatrix}$.
- E la matrice contenant la partie inférieur de A (sans sa diagonale) : $E = \begin{pmatrix} 0 & \cdots & \cdots & 0 \\ A_{1j} & \ddots & \iddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ A_{ij} & \cdots & A_{(i-1)j} & 0 \end{pmatrix}$
- F la matrice contenant la partie supérieure de A (sans la diagonale) : $F = \begin{pmatrix} 0 & A & \cdots & A_{ij} \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \iddots & \ddots & A \\ 0 & \cdots & \cdots & 0 \end{pmatrix}$

L'équation initiale $AX = B$ devient alors : $(E + D + F)\,X = B$, ce qui donne

$$\Rightarrow DX = b - (E + F)\,X$$

$$\Rightarrow X = D^{-1}\,(b - [E + F]\,X)$$

$$\begin{cases} X^0 & k = 0 \\ X^{k+1} = D^{-1}\left(b - [E + F]\,X^k\right) & k = 1, \ldots, N \end{cases}$$

Ainsi, pour un X^0 donné, on peut calculer $X_i^{k+1} = \frac{b_i - \sum_{j=1, j\neq i}^{N} A_{ij} X_j^k}{A_{ii}}$ pour $i = 1, \ldots, N$.

On peut alors calculer la norme : Norm $= \|x_{k+1} - x_k\|_2 = \sqrt{\sum_{n=0}^{N-1} \left(x_{k+1}\,[n] - x_k\,[n]\right)^2}$

On définit $\varepsilon = 0,001$ et $X_k = X_0$.

Algorithme

```
Tant que ((Norme > Epsilon) et (it < it_max)) Faire
Début

Fin
```

5.2 Méthode de Gauss-Seidel

Méthode

d

Algorithme

f

6 Conclusion

Ainsi, nous avons trouvé différentes méthodes pour résoudre l'équation de matrice : $AX = B$ sans à calculer directement A^{-1}.