UNIVERSITÉ DE NANTES

BA RAGAA Mohammed Ali Ahmed
Groupe 485L

Algorithmique numérique

Décomposition LU

Licence 2 informatique

Le 02 Mars 2020

Introduction

Ce projet a pour sujet la notion de Décomposition LU qui possède plusieurs applications dans la vraie vie comme résoudre un système, trouver le déterminant et plein d'autre.
nous allons faire dans ce projet un algorithme itératif qui calcule le déterminant et une autres qui résout un système linéaire en utilisant la décomposition LU.

GitHub

Ce projet est disponible (code + fichier d'exécution) sur GitHub.
https://github.com/BARAGAA/Creation-numerique

Plan de contenu

decompLU

L'algorithme decompLU

fonction decompLU(Pointeur vers un tableau de réels *A, entier non signé n) : réel

```
Variables :
précondition : les pivot != 0 sans changement de ligne
sousPivot, lineEnd, i ,l ,s  : entiers
début:
    pour i allant de 0 à n²  avec incrémentation de n faire
        si A[i] == 0
        retourner faux
        fin si
        pour l allant de (i+n) à n² avec incrémentation de n faire
            sousPivot ← A[l] / A[i]
            A[l] ← sousPivot
            lindEnd ← n-(l mod n)
            pour s allant de 1 à lineEnd faire
                A[l+s] ← A[l+s]  - (sousPivot * A[i+s] )
            fin pour
        fin pour
    fin pour
    retourner true
fin
```

Complexité temporelle

Comme l'algorithme d'élimination Gausse cet algorithme à O(n3) comme complexité temporelle car il a **3 boucles pour imbriquées**, pourtant où n'est la dimension de matrice et pas le nombre d'éléments comme la plupart des algorithmes.

Limitations

Comme indiqué aux préconditions, DecompLU n'est pas valide aux cas où :

1) Le déterminant = 0, autrement dit la matrice n'est pas inversible ni solvable.
2) La matrice est solvable mais elle requit un permutation de ligne .

dans le deuxième cas, elle existe une manière de décomposer la matrice en utilisant l'algorithme de décomposition PLU où la matrice P est la matrice de permutation (matrice d'identité dont les lignes sont permutés avec la matrice originale) .

Le Déterminant

L'algorithme det

Pseudocode

```
fonction det( Pointeur vers un tableau de réels *A, entier non signé n) : réel
        Variables :
                reels   d
                entier i
        début
                d← 1
                i← 0
        si (decompLU(A,n)) alors
                pour i allant de 0 à n² avec incrémentation de (n+1)    faire :
                        d ← d x A[i]
                fin pour

                retourner d
        sinon
                retourner 0
        fin
```

Complexité temporélle

comme l'algorithme de décomposition, cette algorithme a$O(n^3)$. pour la complexité temporelle parce que la complexité de celle-ci est plus grand que la boucle pour dans le corbe de la fonction.

Testes

Pour la matrice A nous obtenons les résultats x1, x2 pour les vecteurs associés b1 et b2

$$A = \begin{bmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{bmatrix}$$

$$b1 = \begin{bmatrix} 32 \\ 23 \\ 33 \\ 31 \end{bmatrix} \qquad x1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$b2 = \begin{bmatrix} 32.1 \\ 22.9 \\ 33.1 \\ 30.9 \end{bmatrix} \qquad x2 = \begin{bmatrix} 9.2 \\ -12.6 \\ 4.5 \\ -1.1 \end{bmatrix}$$

Une petite modification sur le vecteur b produit une grosse différence sur le résultat, qui nous permet de présumer que le système est (ill-conditioned system) .

Pour la matrice b nous obtenons les résultats x1, x2 pour les vecteurs associés b1 et b2

$$B = \begin{bmatrix} 1 & 7 & 2 & 1 \\ 7 & 5 & 1 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 1 \end{bmatrix}$$

$$b1 = \begin{bmatrix} 32 \\ 23 \\ 33 \\ 31 \end{bmatrix} \qquad x1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$b2 = \begin{bmatrix} 32.1 \\ 22.9 \\ 33.1 \\ 30.9 \end{bmatrix} \qquad x2 = \begin{bmatrix} 0.955045 \\ 1.012908 \\ 1.013650 \\ 1.027300 \end{bmatrix}$$

Nous pouvons observer en opposé de la première que résultat cette matrice à un changement faible dans sa résultat ce qui nous indique que le système est (well conditioned).

Calculs de conditionnement

$$A = \begin{bmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{bmatrix} \qquad \mathrm{A}^{-1} = \begin{bmatrix} 25 & -41 & 10 & -6 \\ -41 & 68 & -17 & 10 \\ 10 & -17 & 5 & -3 \\ -6 & 10 & -3 & 2 \end{bmatrix}$$

L1 = |10| + |7| + |8| + |7| = 32
L2 = |7| + |5| + |6| + |5| = 23
L3 = |8| + |6| + |10| + |9| = 33
L4 = |7| + |5| + |9| + |10| = 31

||A|| = 33

L1 = |25| + | - 41| + |10| + | - 6| = 82
L2 = |-41|+|68|+|-17|+|10| = 136
L3 = |10| + | - 17| + |5| + | - 3| = 35
L4 = | - 6| + |10| + | - 3| + |2| = 21

||A-1|| = 136

Cond[A] = 33 x 136 = 4488

On peut voir que le conditionnement est grand, ce qui nous confirmer que le système est (ill conditioned).

$$B = \begin{bmatrix} 1 & 7 & 2 & 1 \\ 7 & 5 & 1 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 1 \end{bmatrix} \qquad \mathrm{B}^{-1} = \begin{bmatrix} -0.109 & 0.154 & 0.085 & 0.101 \\ 0.16 & 0.015 & -0.025 & -0.009 \\ -0.003 & -0.128 & 0.065 & 0.0504 \\ -0.006 & -0.005 & 0.13 & -0.143 \end{bmatrix}$$

L1 = |1| + |7| + |2| + |1| = 11
L2 = |7| + |5| + |1| + |5| = 18
L3 = |8| + |6| + |10| + |9| = 33
L4 = |7| + |5| + |9| + |1| = 22

||B|| = 33

L1 = 0.449
L2 = 0.20
L3 = 0.246
L4 = 0.284

||B-1|| = 0.449

On peut voir que le conditionnement est grand, ce qui nous confirmer que le système est (well conditioned).

Conclusion

En conclusion nous avons utilisé la connaissance et la compétence acquises dans cette partie du module (Algorithmique numérique) dans de projet et le TP3 pour résoudre un système linéaire en plusieurs manières et le comparer chacune son point fort et faible.

Nous avons commencé ce travail par bien comprendre ce qui est demandé et ensuite écrire les algorithmes demandés avec ces pseudo-codes et effectuer les tests en utilisant le travail réalisé en TP3 .