
T.D. D'INFORMATIQUE PC*

Autour de la décomposition LU

9 octobre 2003

Dans ce TD, nous étudions en détail un type de décomposition matricielle appelée “*décomposition LU*”. Il y a beaucoup de décompositions matricielles et il ne faut pas croire que cette décomposition est aussi utile que des décompositions plus classiques comme la diagonalisation, la triangularisation ou les décompositions de Dunford et en blocs de Jordan. Elle est cependant très utilisée dans certains problèmes spécifiques d’analyse numérique. Il y a beaucoup d’autres décompositions singulières comme celle-là : la décomposition de Gram-Schmidt (dite “*QR*”), ou les décompositions de Smith pour les matrices à coefficients dans \mathbb{Z} par exemple.

Le niveau de ce TD est progressif. Chaque question a pour prétexte une question mathématique, mais elle met surtout en lumière une ou des fonctions MAPLE spécifiques. Des générateurs aléatoires pour les premières, l’utilisation simple des booléens, etc... Donc prenez des notes sur les outils utilisés dans chaque question !

Le signe (*) signifie “subsidaire”.

Dans tout le TD, nous utiliserons exclusivement la librairie `linalg`, d’algèbre linéaire de MAPLE. N’oubliez pas de la charger à chaque fois !

§ 1. PRÉLIMINAIRES

Soit \mathcal{M} une matrice inversible de taille n , à coefficients dans un corps. On appelle *décomposition LU* de \mathcal{M} l’écriture suivante :

$$\mathcal{M} = \mathcal{L}\mathcal{U},$$

où \mathcal{L} est une matrice triangulaire inférieure (“L” signifiant “low”) ne comportant que des 1 sur la diagonale et \mathcal{U} une matrice triangulaire supérieure (on aura compris que “U” signifie “up”).

Remarquons tout d’abord que U est nécessairement inversible.

Proposition § 1.1 (i) *Si une telle décomposition existe, elle est unique.*
(ii) *La matrice \mathcal{M} possède une décomposition LU (on dira aussi pour simplifier que \mathcal{M} est LU-décomposable) si et seulement si tous les mineurs principaux de \mathcal{M} sont non nuls. On entend par “mineur principal d’ordre k ” de la matrice \mathcal{M} le déterminant de la sous-matrice de \mathcal{M} formé par les k premières lignes et les k premières colonnes de \mathcal{M} , k étant un entier compris entre 1 et n , où n désigne la taille de \mathcal{M} .*

Cette proposition peut faire l’objet d’un petit exercice de khôlle. Nous la supposons acquise dans le reste du TD. Libre à vous de la démontrer ultérieurement.

Remarquez que le corps des scalaires n'intervient pas dans cette proposition, et pour cause, celle-ci reste vrai indépendamment de son choix.

ON TRAVAILLE JUSQU'À NOUVEL ORDRE AVEC DES MATRICES À COEFFICIENTS RATIONNELS (i.e. DANS \mathbb{Q}).

La première question à se poser est la suivante : la condition (ii) est-elle contraignante ?

On adoptera, pour y répondre la stratégie suivante, qui constitue d'ailleurs une démarche algorithmique tout à fait générale :

- On commencera par *générer nos objets d'étude de manière aléatoire*, c'est à dire des matrices à coefficients rationnels. Il s'agit donc de générer de manière aléatoire des matrices carrés de taille n arbitraire à coefficients bornés par une taille B arbitraire :
 - 1 – Comment peut-on générer des entiers compris entre $-B$ et B où $B \in \mathbb{N}^*$ de manière aléatoire ? On donnera un *exemple* avec la fonction `rand`.
 - 2 – Créez une petite *procédure* dépendant d'un paramètre $B \in \mathbb{N}^*$ qui génère un rationnel aléatoire dont le numérateur est compris entre $-B$ et B et le dénominateur est compris entre 1 et B .
 - 3 – Vous pouvez maintenant vous servir de votre procédure, pour en générer une autre qui prend comme paramètre l'entier B précédant et un entier $n \in \mathbb{N}^*$ et qui produit une matrice carrée de taille n , à coefficients rationnels bornés comme dans la question précédente. On n'oubliera pas de charger la librairie associée : `linalg`. On utilisera la fonction `matrix`.
- Maintenant que nous savons créer des exemples de manière automatique et *a priori* aléatoirement, testons si nos matrices sont LU -décomposable. On créera une procédure qui prend une matrice en paramètre et qui répond `true` si elle est LU -décomposable et bien sûr `false` sinon. Cherchez les fonctions adéquates dans la bibliothèque de fonctions de `linalg`.
- Vous devriez maintenant pouvoir répondre à la question posée initialement ? À votre avis, quel est la raison mathématique de cela ?

Nous savons tester si une matrice est LU -décomposable. Il va de soi que nous souhaiterions maintenant, le cas échéant, la décomposer...

§ 2. MÉTHODE NAÏVE

On cherche tout d'abord à implémenter une décomposition LU naïve. En algorithmique, une telle méthode s'appellerait *Brute Force*.

Décrivez *sur le papier* un algorithme "naïf" de décomposition LU . Inspirez-vous, par exemple, du système que vous chercheriez à résoudre pour $n = 3$, sans ordinateur, pour une matrice du type suivant :

$$\begin{pmatrix} -85 & -55 & -37 \\ -35 & 97 & 50 \\ 79 & 56 & 49 \end{pmatrix}.$$

- 1 - On écrira cette procédure. Mais on préférera la décomposer en sous-procédures indépendantes de manière à limiter les erreurs...

2 - (*) Testez le temps de calcul de cette procédure. Qu'en pensez-vous ? Comparez votre algorithme à la fonction MAPLE de décomposition LU (elle s'appelle `LUdecomp`). On ne se fera pas trop d'espoir...

§ 3. MÉTHODE PAR PIVOT DE GAUSS

Nous allons maintenant traiter la méthode "classique" de décomposition LU .

3.1.

Deux petits rappels : considérons l'opération élémentaire sur une matrice M de taille n consistant à transformer la ligne i en la somme de la ligne i plus un coefficient α que multiplie la ligne j pour $1 \leq i \neq j \leq n$. Nous écrivons cette opération :

$$\mathcal{L}_i \leftarrow \mathcal{L}_i + \alpha \mathcal{L}_j.$$

À quelle multiplication matricielle cette opération correspond-elle ? Même question pour l'opération sur les colonnes :

$$\mathcal{C}_j \leftarrow \mathcal{C}_j + \alpha \mathcal{C}_i.$$

3.2.

Considérons tout d'abord un exemple : on se donne la matrice suivante :

$$M = \begin{pmatrix} 4 & 1 & 3 \\ 5 & 2 & 7 \\ 3 & 6 & 2 \end{pmatrix}.$$

Vérifiez que cette matrice est bien LU -décomposable.

Par la méthode du "pivot de Gauss" écrivez la liste des pivots nécessaires de manière à transformer cette matrice en matrice triangulaire supérieure. On effectuera des pivots de la forme :

$$\mathcal{L}_i \leftarrow \mathcal{L}_i + \alpha \mathcal{L}_j$$

où $1 \leq i < j \leq n$. Déduisez-en les matrices L et U de la décomposition de M .

3.3.

Nous voulons automatiser cette méthode. Considérez l'algorithme suivant :

Algorithme 3 Décomposition LU par pivot

ENTRÉE: une matrice M qui est LU -décomposable

SORTIE: les matrices L et U

Step 1. $L \leftarrow I_n$

$$U \leftarrow A$$

Step 2. $\mathcal{L}_j \leftarrow \mathcal{L}_j - \frac{U[j,i]}{U[i,i]} \mathcal{L}_i$ pour U [$1 \leq i \leq n$ et $i < j \leq n$]

$$\mathcal{C}_j \leftarrow \mathcal{C}_j + \frac{U[j,i]}{U[i,i]} \mathcal{C}_i$$
 pour L

- Preuve de l'algorithme :
 - 1 - On pourra introduire les matrices $L_{i,j}$ et $U_{i,j}$ où $1 \leq i \leq n$ et $i < j \leq n$ qui sont les matrices L et U à la (i, j) -ième étape. On montrera que cette suite est bien définie,
 - 2 - Montrer que pour tout (i, j) , on a : $M = L_{i,j}U_{i,j}$,
 - 3 - Conclure.
- Implémentez en MAPLE cet algorithme à l'aide des fonctions `addrow` et `addcol`.
- (*) Que dire du temps de calcul ? Est-il plus rapide ?
- (*) Pouvez-vous majorer en fonction de n le nombre d'opérations (en mélangeant additions et multiplications) effectuées pour terminer l'algorithme.

NOTE : En fait, on a le résultat plus complet suivant :

Proposition § 3.1 *Pour toute matrice inversible \mathcal{M} , il existe une matrice de permutation \mathcal{P}_σ telle que $\mathcal{P}_\sigma * \mathcal{M}$ est LU-décomposable.*

On rappelle qu'une matrice de permutation de taille n est une matrice de la forme suivante : soit σ une permutation de l'ensemble $\{1, \dots, n\}$, ou en d'autre terme une bijection de cet ensemble dans lui-même, la matrice de permutation associée est la matrice \mathcal{P}_σ , où :

$$\mathcal{P}_{\sigma i,j} = 1 \text{ si } \sigma(i) = j \text{ et } 0 \text{ sinon.}$$

Cette matrice provient simplement des cas où les pivots rencontrés $U[i, i]$ sont nuls. Comme la matrice est inversible, il en existe un autre non nul et il est alors nécessaire de faire une permutation entre les lignes.