Université de Bordeaux **Algèbre et calcul formel – Agrégation**
Mathématiques 2022–2023

FEUILLE D'EXERCICES n° 10

Algèbre linéaire : décompositions de matrices

1. Matrices définies sur un corps

Exercice 1 – [Forme de Jordan]

Nous avons déjà travaillé sur la diagonalisation des matrices. Sur un corps algébriquement clos, les matrices non diagonalisables peuvent-être triangulées, et même mises sous forme de Jordan. Pour cela, on dispose de la fonction `Jordan_form`.

On va construire un exemple de matrice non diagonalisable sur $\mathbb{C}$, dont la réduction est formée de plusieurs blocs de Jordan.

```
J1 = jordan_block(5,3)
J2 = jordan_block(2,5)
J3 = jordan_block(1,4)
J = block_diagonal_matrix(J1,J2,J3)
```

Cela doit donner une matrice sous forme de Jordan, dont on peut prendre un conjugué, pour construire un exemple non-trivial. Pour construire notre conjugué, on peut soit construire une matrice de passage P particulière dont on sait qu'elle est inversible, soit prendre une matrice P au hasard jusqu'à ce qu'elle soit inversible (ce qui a de grandes chances d'arriver).

```
M12Q = MatrixSpace(QQ,12)
P = M12Q.random_element()
A = P^(-1)*J*P
```

1) Calculer la forme de Jordan de A en utilisant `A.jordan_form()` et vérifier que c'est bien ce à quoi on s'attend.

Remarque. On a construit ci-dessus des matrices diagonales par blocs en utilisant `block_diagonal_matrix`. Remarquons au passage l'existence de la fonction `block_matrix`, qui permet la construction de matrices par blocs.

Exercice 2 – [Décomposition LU]

Théorème 1. *Soient K un corps et $A \in \mathrm{GL}_n(K)$. Alors il existe une matrice de permutation P, une matrice triangulaire supérieure U et une matrice triangulaire inférieure L dont les éléments diagonaux sont égaux à 1 telles que $A = PLU$. De plus, pour P fixée, cette décomposition est unique si elle existe.*

Cette décomposition s'obtient grâce à l'algorithme du pivot de Gauss. Une décomposition $A = LU$ (c'est-à-dire où $P = I_n$) existe si et seulement si on ne rencontre pas de « pivot nul » dans cet algorithme. En fait, on montre que la décomposition $A = LU$ existe si et seulement si pour tout $k \in [[1, n]]$, $\det A_k \neq 0$, où A_k est la sous-matrice supérieure gauche de taille k de A. Par exemple, les matrices de $\mathrm{GL}_n(\mathbb{C})$ à *diagonale strictement dominante* admettent une décomposition $A = LU$, ainsi que les matrices réelles *symétriques définies positives* et les matrices complexes *hermitiennes définies positives*.

Sur Sage, la décomposition $A = PLU$ de A est donnée par `A.LU()`.

Soit $A = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 10 \end{pmatrix}$. Donner la décomposition LU de A (on pourra essayer la fonction `LU` sans option, et aussi avec l'option `pivot='nonzero'`).

1) Soit $B = \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}$. Pour entrer B, on peut utiliser `vector`. En utilisant cette décomposition LU, et en résolvant deux systèmes triangulaires sur papier, résoudre $AX = B$.

2) Définir la concaténée C de A et B, appliquer la fonction `LU` à C et retrouver la solution de l'équation par résolution d'une équation triangulaire.

3) Vérifier le résultat obtenu en utilisant `A.solve_right(B)`.

4) Même exercice avec $A = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 8 & 8 \\ 3 & 6 & 10 \end{pmatrix}$ et $B = \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}$.

Soit maintenant une matrice A qui n'est pas supposée inversible, ni même carrée. Soit $A \in \mathcal{M}_{m,n}(K)$, le résultat précédent se transpose à ce cas : il existe une décomposition $PA = LU$, où P est une matrice de permutation de taille m, L est une matrice triangulaire inférieure de $\mathrm{GL}_m(K)$ dont les coefficients diagonaux sont égaux à 1, et U est une matrice *échelonnée par lignes* de $\mathcal{M}_{m,n}(K)$. Il n'y a pas unicité en général, même si P est fixée.

Rappelons ce qu'est une matrice échelonnée par lignes.

Définition 2. Soit $E = (e_{ij})_{i,j} \in \mathcal{M}_{m,n}(K)$. On dit que E est *échelonnée par lignes* s'il existe un entier $r \leqslant \min(m, n)$ et une fonction $f \colon [[1, r]] \longrightarrow [[1, n]]$ strictement croissante telle que les propriétés suivantes sont vérifiées.

(1) Pour tout $i \in [[1, r]]$, $e_{if(i)} \neq 0$.
(2) Si $j < f(i)$, alors $e_{ij} = 0$.
(3) Si $i > r$, alors $e_{ij} = 0$.

Le rang de E est alors égal à r.

La matrice E est dite *échelonnée réduite par lignes* si de plus pour tout $i \in [[1, r]]$:

(1) $e_{if(i)} = 1$
(2) pour tout $k < i$, $e_{kf(i)} = 0$.

Dans sage, la matrice U donnée par `A.LU()` n'est pas échelonnée par lignes, mais seulement triangulaire supérieure. Par contre, la commande `A.echelon_form()` rend une matrice $E \in \mathcal{M}_{m,n}(K)$ échelonnée réduite par lignes telle qu'il existe $G \in \mathrm{GL}_m(K)$ vérifiant $A = GE$. On peut souhaiter récupérer la matrice G. C'est utile par exemple pour calculer l'image de A. La fonction `extended_echelon_form` donne E et G, sous la forme concaténée $(E|G)$.

5) Essayer la commande `LU` sur `A=matrix(QQ,[[1,2,3],[3,2,1],[1,1,1]])`. On voit sur sa décomposition LU que A n'est pas inversible. En utilisant cette décomposition, calculer $\mathrm{Im}\, A$, $\mathrm{Ker}\, A$, puis résoudre $AX = B$, où $B = {}^t(1, 2, 3)$. Essayer aussi avec $B = {}^t(2, 2, 1)$. Retrouver ces résultats en utilisant les commandes `column_space`, `right_kernel`, et `solve_right`. (Ne pas oublier que `kernel` et `solve` sont à gauche par défaut.)

6) Soit $A = \begin{pmatrix} 0 & 1 & 2 & 1 & 1 \\ 0 & 1 & 2 & 0 & 2 \\ 0 & 2 & 4 & 3 & 1 \\ 0 & -1 & -2 & 3 & 0 \end{pmatrix}$. En utilisant les fonctions citées ci-dessus, Calculer $\mathrm{Ker}\, A$ et $\mathrm{Im}\, A$, puis résoudre $AX = B$, où $B = {}^t(1, 1, 1, 1)$, puis où $B = {}^t(1, 1, 2, 0)$.

Exercice 3 – [LU : VÉRIFICATION EXPÉRIMENTALE DE LA COMPLEXITÉ]

1) Quelle est la complexité asymptotique de la factorisation LU, en terme d'opérations dans le corps de base K (en fonction de la dimension n de la matrice).

2) Vérifiez expérimentalement cette complexité, en prenant des matrices carrées au hasard dans $\mathbb{F}_7$ et en faisant augmenter la dimension (cf TD09).

Exercice 4 – [DÉCOMPOSITION QR]

Théorème 3. *Soit $A \in \mathrm{GL}_n(\mathbb{R})$. Alors il existe une matrice triangulaire supérieure R dont les coefficients diagonaux sont strictement positifs et une matrice orthogonale Q telles que $A = QR$. De plus, cette décomposition est unique.*

Cette décomposition permet de ramener la résolution d'une équation $AX = B$ à la résolution d'une équation triangulaire. En effet, si $A = QR$ est la décomposition QR de A (supposée inversible), alors l'équation $AX = B$ est équivalente à l'équation $RX = {}^tQB$.

Par exemple, on définit la matrice
`A=matrix(ZZ,[[1,2,3],[1,-2,0],[-2,2,1 ]])`

Si on essaie `A.QR()`, ça ne marche pas. Dans l'aide à la décomposition QR, le corps de base utilisé est une clôture algébrique de $\mathbb{Q}$. Essayons.
`M3Qbar = MatrixSpace(QQbar,3,3); A = M3Qbar(A); A.QR()`

On doit trouver la décomposition QR de A (vérifier que QR est bien une approximation numérique de A).

Comme pour la décomposition LU, le théorème précédent se transpose au cas où A n'est pas inversible, et au cas où A n'est pas carrée. Soit donc $A \in \mathcal{M}_{m,n}(\mathbb{R})$. Il existe une décomposition $A = QR$, où $Q \in \mathcal{O}_m(\mathbb{R})$ et où R est une matrice *échelonnée par lignes* de $\mathcal{M}_{m,n}(\mathbb{R})$. Il n'y a pas unicité en général.

En utilisant la décomposition QR, retrouver les résultats déjà obtenus ci-dessus (grâce à la décomposition LU de A), pour $A = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}$.

2. Matrices à coefficients dans $\mathbb{Z}$

Exercice 1 – [Forme normale d'Hermite]

Définition 1. Soit $E = (e_{ij}) \in \mathcal{M}_{m,n}(\mathbb{Z})$ une matrice échelonnée par ligne de rang r. Pour $i \in [[1, r]]$, on note $f(i)$ le plus petit entier tel que $e_{if(i)} \neq 0$ (c'est la fonction f de la définition 2 du paragraphe 2). Alors E est dite *échelonnée réduite* si pour tout $i \in [[1, r]]$

- $e_{if(i)} > 0$,
- pour tout $k \in [[1, i-1]]$, $0 \leqslant e_{k,f(i)} < e_{i,f(i)}$.

Théorème 2. *Soit $A \in \mathcal{M}_{m,n}(\mathbb{Z})$. Il existe une unique matrice échelonnée réduite H de $\mathcal{M}_{m,n}(\mathbb{Z})$ et une matrice unimodulaire $U \in \mathrm{GL}_n(\mathbb{Z})$ telles que $A = UH$. La matrice H s'appelle la forme normale d'Hermite de A.*

Cela s'obtient par des opérations sur les lignes, en utilisant la relation de Bézout sur $\mathbb{Z}$, et donc l'algorithme d'Euclide étendu (une des briques de base pour ce calcul est ce qu'on a fait dans l'exercice 2 du TD7).

En sage, pour calculer la forme normale d'Hermite d'une matrice entière A, on utilise `A.hermite_form()`. Si on veut obtenir la matrice unimodulaire U en plus de la forme normale de Hermite H, on peut utiliser `A.hermite_form(transformation = True)` (c'est rappelé dans l'aide).

Applications.

Équation diophantienne linéaire. On veut trouver les solutions entières de l'équation

$$2x_1 + 3x_2 + 5x_3 = 0,$$

c'est-à-dire $AX = 0$, où $A = (2, 3, 5)$. Ici, la matrice A est échelonnée par lignes, mais cela ne nous aide pas. Il vaut mieux l'échelonner par colonnes, en appliquant `hermite_form` à la transposée de A. On trouve alors une matrice U de $\mathrm{GL}_3(\mathbb{Z})$ et un entier a tels que $AU = (a, 0, 0)$. En écrivant $X = UY$, on est amené à résoudre

$$(1, 0, 0)Y = 0.$$

On trouve alors l'ensemble des solutions grâce à l'expression $X = UY$.

Essayez aussi la commande `kernel` (ou `right_kernel`).

Système d'équations diophantiennes linéaires. Choisir une matrice au hasard A dans $\mathcal{M}_{3,4}(\mathbb{Z})$ et un vecteur B au hasard dans $\mathcal{M}_{3,1}(\mathbb{Z})$, et résoudre l'équation $AX = B$ dans $\mathbb{Z}$.