Lycée Louis-Le-Grand, Paris 2019/2020
MPSI 4 – Informatique pour tous
N. Carré, A. Troesch

TP n° 12 : Pivot de Gauss

Ce TP est à faire en autonomie avant le 18/05/2020. Votre code (le fichier source, pas une photo ou une capture d'écran) est à envoyer à votre chargé de TP (N. Carré ou A. Troesch), selon les modalités que chacun aura décidé (mail ou autre). Les noms de fichier devront impérativement respecter le format suivant : `info12-troesch.py` par exemple pour le code du TP12 de l'élève Troesch. Si vous avez plusieurs fichiers, rajoutez une numérotation quelque part.

Les tests demandés sur les exemples sont **obligatoires**, et la ligne de commande de l'exécution de ces tests doit être laissée dans le code (en commentaire). Cela dans le but de faciliter la correction.

Exercice 1 – Échelonnement d'une matrice et résolution d'un système

Les matrices seront représentées par des tableaux du module `numpy` (`numpy.array`). On constatera que le caractère mutable des tableaux a pour effet que lors de l'utilisation d'une fonction sur un paramètre de type tableau, les coefficients du tableau passé en paramètre sont modifiés globalement. Ainsi, il n'est pas utile de retourner le tableau modifié en sortie de fonction : les modifications auront été faites sur le tableau passé en paramètre.

Tous les tests demandés se feront avec les matrice $A_1 = \begin{pmatrix} 1 & 2 & 0 & 1 & 1 \\ 2 & 1 & 0 & 0 & 1 \\ -1 & -2 & -1 & 1 & 1 \\ 0 & 1 & 1 & -1 & 2 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$, $A_2 = \begin{pmatrix} 1 & 2 & 1 & 2 & 3 \\ 1 & 3 & 1 & 2 & 0 \\ 0 & 1 & 0 & -3 & 0 \end{pmatrix}$, $A_3 = \begin{pmatrix} 1 & 2 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$, $B_1 = \begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix}$ et $B_2 = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}$.

Pour ne pas avoir à redéfinir entièrement vos matrices A_1 et A_2 après chaque essai (car elles seront transformées), vous pouvez définir une fonction `creeA1()` qui vous retourne la matrice A_1. De même pour les autres matrices.

NB : toutes les indexations commencent à 0, y compris dans les tests.

1. Écrire des fonctions `echangeL(i,j)`, `multiplicationL(a)` et `transvectionL(i,j,a)` et réalisant les opérations usuelles $L_i \leftrightarrow L_j$, $L_i \leftarrow aL_i$ et $L_i \leftarrow L_i + aL_j$. On ne s'autorisera que des opérations coefficients par coefficients. Ces fonctions ne retourneront pas de résultat mais opéreront les modifications directement sur la matrice donnée en paramètre.

   Test : sur la matrice A_1, effectuer successivement $L_0 \leftrightarrow L_2$, $L_1 \leftarrow L_1 - 4L_4$ et $L_3 \leftarrow -2L_3$. Les lignes de commande de ces tests sont à laisser dans le code (en commentaire, i.e. précédées d'un #). De même pour tous les tests suivants.

2. Écrire de même des fonctions `echangeC(i,j)`, `multiplicationCL(a)` et `transvectionC(i,j,a)` et réalisant les opérations usuelles $C_i \leftrightarrow C_j$, $C_i \leftarrow aC_i$ et $C_i \leftarrow C_i + aC_j$.

   Test : sur la matrice A_1 réinitialisée, effectuer successivement $C_0 \leftrightarrow C_2$, $C_1 \leftarrow C_1 - 4C_4$ et $C_3 \leftarrow -2C_3$.

3. Écrire une fonction `echelonne(A,B)` échelonnant une matrice A, et effectant les mêmes opérations sur une matrice B ayant le même nombre de lignes et un nombre quelconque de colonnes. On choisira de préférence le pivot de valeur absolue maximale. On retiendra dans une liste qu'on retournera en sortie (`return...`) les indices des colonnes des pivots utilisés. Ainsi, si lors de l'échelonnement, on a trouvé un pivot sur les colonnes d'indice 0, 2 et 3, la liste retournée doit être $[0, 2, 3]$. On ne normalisera pas les pivots ici.

   On remarquera pour la suite que B peut être choisie vide, mais avec le bon nombre de lignes, donc représentée sous la forme $[[], [], \ldots, []]$.

   Test : à faire avec $A = B = A_1$ (réinitialisée). Quel est l'intérêt de faire le test avec $A = B$?

4. Écrire une fonction `A, pivot,B` prenant en paramètre une matrice échelonnée A, une liste `pivot` correspondant à la position des pivots (i.e. la position de colonne des premiers termes non nuls de chaque ligne non nulle de A) sous le format donné dans la question précédente, et une matrice B, et effectuant un pivot remontant sur la matrice A (c'est-à-dire en annulant les coefficients au-dessus de chaque pivot), ainsi que les mêmes opérations sur B. On normalisera les pivots lors de ce calcul. On remarquera que si le terme d'indice i de la liste des positions de pivot est k, on a un pivot en position (i, k), puisque chacune des premières lignes possède un et un unique pivot.

   Test : avec $A = A_3$, pivot $= [0, 1, 3]$, $B = B_1$.

5. Se servir de la fonction précédente pour écrire une fonction `rang(A)` calculant le rang d'une matrice A.

   Test : à faire avec A_2

6. Écrire de même une fonction calculant l'inverse d'une matrice (et retournant un message d'erreur « `Matrice non inversible` » si la matrice n'est pas inversible)

   Test : à faire avec A_1.

   Comparer le temps d'exécution suivant la taille de matrices dont les coefficients seront choisis aléatoirement entre 1 et 1000. Confirmer numériquement la complexité de l'algorithme du pivot.

7. Écrire une fonction calculant le déterminant d'une matrice.

   Test : avec A_1

8. Écrire une fonction résolvant le système $AX = B$, à savoir :
   - retournant un message d'erreur si le système n'a pas de solution
   - retournant un couple (X_0, B), où X_0 est une solution particulière, et B une liste de vecteurs formant une base de l'espace des solutions de l'équation homogène associée.

   On rappelle qu'une fois le système échelonné, avec pivot remontant (donc les pivots sont seuls sur leur colonne) et normalisé :
   - La condition d'existence d'une solution s'obtient en vérifiant que les coefficients de B associés aux lignes nulles de A sont nuls également ;
   - une solution particulière est obtenue en annulant toutes les inconnues secondaires, les inconnues principales prennent alors la valeur du second membre. Ainsi, une solution particulière est obtenue en réalisant un « shuffle » du vecteur nul et du second membre.
   - une base de l'espace des solutions de l'équation homogène est obtenu en énumérant une base de l'espace formé par les inconnues secondaires (par exemple la base canonique), et en calculant les inconnues principales. Ainsi, un vecteur de base est obtenu en mettant à 1 l'inconnue secondaire x_k, à 0 toutes les autres et en calculant les inconnues principales. ce faisant, on se rend compte que les inconnues secondaires prennent la valeur de l'opposé de la colonne C_k de la matrice A. Ainsi, un vecteur de base est obtenu en effectuant un « shuffle » d'un vecteur de la base canonique (pour les inconnues secondaire) et de l'opposé de la colonne de A correspondant à cette inconnue secondaire.

   On remarquera qu'un tel shuffle est ici rendu simple par le fait que l'un des deux vecteurs qu'on mélange est presque nul : on pourra donc partir d'un vecteur global nul, modifier les variables principales, puis modifier l'éventuelle inconnue secondaire non nulle.

   Test : à faire avec A_2, et chacun des seconds membres B_1 et B_2.

On pourra réutiliser les fonctions de l'exercice précédent, notamment les fonctions de résolution de système pour les comparaisons demandées en fin d'exercice.

Exercice 2 – (Décomposition LU)

Soit A une matrice telle que toute sous-matrice principale (sous-matrice carrée calée dans le coin supérieur gauche) soit inversible. On a montré dans le cours qu'alors la matrice A admet une décomposition $A = LU$, où L est une matrice triangulaire inférieure et U une matrice triangulaire supérieure. La matrice L est obtenue en appliquant à la matrice identité les opérations sur les colonnes correspondant aux inverses des matrices d'opérations sur les lignes permettant de trianguler A par la méthode de Gauss (sans échange de ligne).

L'intérêt réside dans le fait qu'on ramène la résolution de $AX = B$ à la résolution de 2 systèmes triangulaires. Ceci s'avère notamment intéressant lorsqu'on a plusieurs systèmes à résoudre associés à la même matrice A.

Les tests seront effectués à partir des la matrice $A_1 = \begin{pmatrix} 1 & 2 & 0 & 1 & 1 \\ 2 & 1 & 0 & 0 & 1 \\ -1 & -2 & -1 & 1 & 1 \\ 0 & 1 & 1 & -1 & 2 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$, $A_2 = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix}$, $B_1 = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$

1. Écrire une fonction `LU(A)` retournant les deux matrices L et U de la décomposition ci-dessus, pour une matrice A donnée. On isolera dans des fonctions annexes les opérations sur les lignes. On s'arrangera autant que possible pour éviter les calculs inutiles sur les coefficients qu'on sait être nuls (cela nécessite peut-être d'adapter un peu les fonctions de l'exercice précédent)

   Test : sur la matrice A_1.

2. Écrire une fonction `SystemeTriangulaireSup(A,B)` de résolution d'un système triangulaire supérieur, en évitant tout calcul inutile.

   Test : $A = A_1$ et $B = B_1$

3. Écrire une fonction `SystemeTriangulaireInf(A,B)` de résolution d'un système triangulaire inférieur, en évitant tout calcul inutile.

   Test : $A = {}^tA_1$ et $B = B_1$

4. Soit $A \in \mathcal{M}_{100}(\mathbb{R})$, choisie aléatoirement (les coefficients étant pris dans $[\![0, 1000]\!]$). Comparer :
   - le temps de résolution de 25 systèmes associés à A, en calculant une fois pour toutes la décomposition LU de A, puis en résolvant 2 systèmes triangulaires pour chaque second membre B (les 25 seconds membres seront choisis aléatoirement)
   - le temps de résolution de 25 systèmes associés à A, en reprenant un pivot complet pour chaque système
   - le temps de résolution de 25 systèmes associés à A, en calculant A^{-1} par la méthode du pivot, une fois pour toutes puis en calculant $A^{-1}B$ pour chaque second membre.