Informatique en CPGE (2018-2019)
TP 11 : résolution numérique d'un système linéaire : méthode de Gauss

Exercice 1

Nous commençons par l'étude d'un système linéaire de deux équations à deux inconnues :

$$\begin{cases} a_{0,0}x & + & a_{0,1}y & = & b_0 \\ a_{1,0}x & + & a_{1,1}y & = & b_1 \end{cases}$$

La matrice qui représente ce système est $\begin{pmatrix} a_{0,0} & a_{0,1} & b_0 \\ a_{1,0} & a_{1,1} & b_1 \end{pmatrix}$

et la matrice carrée associée est $\begin{pmatrix} a_{0,0} & a_{0,1} \\ a_{1,0} & a_{1,1} \end{pmatrix}$.

On notera L_0 la ligne $\begin{pmatrix} a_{0,0} & a_{0,1} & b_0 \end{pmatrix}$ et L_1 la ligne $\begin{pmatrix} a_{1,0} & a_{1,1} & b_1 \end{pmatrix}$.

1. Ecrire une fonction `det2x2(m)` qui prend en argument une matrice de type (2,3), et renvoie le déterminant de la matrice carrée associée, soit $d = a_{0,0}a_{1,1} - a_{1,0}a_{0,1}$.

   Tester la fonction `det2x2` avec la matrice $\begin{pmatrix} 2 & 2 & -4 \\ 5 & 13 & 7 \end{pmatrix}$.

   Cette matrice représente le système $\begin{cases} 2x & + & 2y & = & -4 \\ 5x & + & 13y & = & 7 \end{cases}$ et la fonction doit renvoyer le déterminant de la matrice associée $\begin{pmatrix} 2 & 2 \\ 5 & 13 \end{pmatrix}$, soit 16.

   La matrice sera créée par le code suivant :

```
mat = [[2, 2, -4], [5, 13, 7]]
```

2. Ecrire une fonction `transvection(m)` qui prend en entrée une matrice (2,3). La fonction teste si le coefficient $a_{0,0}$ est nul ou pas ; s'il est nul la fonction échange les deux lignes de la matrice et sinon elle remplace la ligne L_1 par la ligne $L_1 - \dfrac{a_{1,0}}{a_{0,0}}L_0$. La fonction renvoie la nouvelle matrice.

   Tester la fonction `transvection(m)` avec la matrice $m_1 = \begin{pmatrix} 2 & 2 & -4 \\ 5 & 13 & 7 \end{pmatrix}$

   puis avec la matrice $m_2 = \begin{pmatrix} 0 & -3 & 6 \\ 7 & 2 & 10 \end{pmatrix}$.

3. Vérifier que les matrices m_1 et m_2 ont été modifiées après l'utilisation de la fonction `transvection`. Si on souhaite ne pas modifier la matrice d'origine, il nous faut créer et utiliser une copie de cette matrice.

   (a) La fonction `matrice`, dont le code suit, permet de créer une matrice de dimension (n, p) dont tous les coefficients sont nuls :

```
def matrice(n, p):
    return [p * [0] for i in range(n)]
```

   Ecrire une fonction `copie(m)` qui prend en argument une matrice `m` ; dans le corps de la fonction, on détermine les dimensions de `m`, puis on crée une nouvelle matrice `mat` de même dimensions que `m` avec la fonction `matrice(n,p)` définie ci-dessus et on copie alors les coefficients de `m` dans la matrice `mat`. La fonction `copie` renvoie la matrice `mat` qui est donc une copie de `m`.

(b) Modifier le corps de la fonction `transvection(m)` de la manière suivante : on commence par déterminer les dimensions de `m`, puis on crée une matrice `mat` copie de `m` à l'aide de la fonction `copie` ; on modifie alors le code restant afin de travailler sur la matrice `mat`.

(c) Ecrire une fonction `solution(m)` qui prend en entrée une matrice qui a été modifiée préalablement par la fonction `transvection`, et qui renvoie la solution du système sous la forme d'une liste (`sol=[sol[0],sol[1]]`).

Résoudre le système de la question 1.

(d) Ecrire une fonction `gauss(m)` qui prend en argument la matrice m d'un système et renvoie la solution du système. On fera appel aux fonctions `transvection` et `solution` dans le corps de la fonction `gauss`.

4. Utiliser la fonction `gauss` pour résoudre le système $\begin{cases} 10^{-20}x + y = 5 \\ x + y = 10 \end{cases}$ puis le système $\begin{cases} x + y = 10 \\ 10^{-20}x + y = 5 \end{cases}$ obtenu en échangeant les deux lignes.

Déterminer "à la main" la solution exacte.

Ceci illustre la nécessité de la recherche du plus grand pivot. Modifier alors le code de la fonction `transvection` afin d'échanger les deux lignes L_0 et L_1 si $|a_{1,0}| > |a_{0,0}|$ et résoudre à nouveau les deux systèmes.

Exercice 2

1. Ecrire une fonction `transpose(m)` qui prend en argument une matrice carrée `m` et renvoie la matrice transposée `tm` (sans modifier `m`). Rappel : si `m` a pour coefficient $a_{i,j}$ alors `tm` a pour coefficients $b_{i,j} = a_{j,i}$.
2. Avec la fonction `gauss` de l'exercice 1, (et donc avec les fonctions `matrice`, `copie`, `transvection` et `solution`), résoudre les systèmes $\begin{cases} 3x + 4y = 1 \\ 5x + 7y = 0 \end{cases}$ puis $\begin{cases} 3x + 4y = 0 \\ 5x + 7y = 1 \end{cases}$.

Quel est le lien entre les deux solutions et l'inverse de la matrice $\begin{pmatrix} 3 & 4 \\ 5 & 7 \end{pmatrix}$?

3. Ecrire une fonction `inverse(m)` qui prend en argument une matrice carrée. La fonction crée une matrice carrée nulle `mat_inv`. Ensuite elle crée la matrice correspondant au premier système précédent, le résout avec la fonction `gauss` et stocke la solution dans la première ligne de la matrice `mat_inv` ; idem pour le deuxième système dont la solution est stockée dans la deuxième ligne de la matrice `mat_inv`. Il n'y a plus qu'à compléter avec un appel à la fonction `transpose` et renvoyer la matrice `mat_inv`.
4. Tester le programme sur différentes matrices.

Exercice 3

Partie 1

L'objectif est d'écrire dans un fichier les différentes fonctions permettant la résolution d'un système linéaire à n équations et n inconnues en utilisant la méthode de Gauss avec recherche du meilleur pivot.

On suppose que les systèmes étudiés ont tous une solution unique.

Le fichier sera enregistré avec le nom gaussToto.py où Toto est à remplacer par votre nom.

Une matrice(n, p), n lignes et p colonnes, sera représentée par une liste de n listes de longueur p.

Il est conseillé de tester les fonctions une à une dès leur écriture.

Les étapes sont les suivantes :

1. Ecrire une fonction `matrice(n,p)` qui prend en argument deux entiers naturels non nuls n et p et renvoie la matrice nulle à n lignes p colonnes.
2. Ecrire une fonction `pivot(m,s)` qui prend en argument une matrice m et un entier s représentant le numéro du pivot provisoire, (l'indice de la ligne contenant le pivot), et renvoie le numéro du meilleur pivot (le plus grand en valeur absolue) qui sera utilisé.
   Par exemple si `m=[[7,5,-4,4],[0,3,-1,5],[0,6,-3,-4]]`, le pivot provisoire est le coefficient 3, donc `s=1` et l'appel `pivot(m,1)` doit renvoyer le nombre 2, indice de la ligne contenant le coefficient 6 qui est le meilleur pivot.
3. Ecrire une fonction `copie(m)` qui prend en argument une matrice m et renvoie une copie de cette matrice. Vérifier que si la copie est modifiée, l'original m ne l'est pas.
4. Ecrire une fonction `permute(m,i,j)` qui prend en argument une matrice m et deux entiers i et j ; la fonction crée une copie de m, permute les lignes i et j de cette copie et renvoie la nouvelle matrice. Vérifier que m n'est pas modifiée après l'appel de cette fonction.
5. Ecrire une fonction `transvection(m,s)` qui prend en argument une matrice m et un entier s (le numéro du pivot utilisé). La fonction crée une copie de m puis, pour i variant de s+1 à n-1 remplace chaque ligne L_i par $L_i - k \times L_s$ où pour chaque ligne L_i, $k = m[i][s]/m[s][s]$.
   Par exemple si `m=[[7,5,-4,4],[0,6,-3,-4],[0,3,-1,5]]`, la fonction renvoie la matrice `[[7,5,-4,4],[0,6,-3,-4],[0,0.0,0.5,7.0]]`.
6. Ecrire une fonction `solution(m)` qui prend en argument une matrice m représentant un système triangulaire supérieur et renvoie la solution de ce système.
   Par exemple si `[[7,5,-4,4],[0,6,-3,-4],[0,0.0,0.5,7.0]]`, la matrice représentant le système $\begin{cases} 7x + 5y - 4z = 4 \\ 6y - 3z = -4 \\ 0,5z = 7 \end{cases}$
   la fonction renvoie `[4.047619047619048, 6.333333333333333, 14.0]`.
7. Ecrire une fonction `gauss(m)` qui prend en argument une matrice m représentant un système et renvoie la solution de ce système. La fonction `gauss` utilise les fonctions `pivot`, `permute`, `transvection` et `solution`.
   Tester sur la matrice `m=[[1,1,1,1,1,5],[1,1,1,1,0,4],[1,1,1,0,0,3],[1,2,3,4,5,15],[0,1,1,1,1,4]]`.
   La solution renvoyée est `[1.0, 1.0, 1.0, 1.0, 1.0]`.

Partie 2

Le fichier "systemes.txt" contient des systèmes à résoudre. Un entier donne le nombre d'équations et les lignes qui suivent donnent les coefficients de la matrice `m` utilisée.
Ouvrir le fichier avec un éditeur pour observer son contenu.

L'objectif est de résoudre ces systèmes. Pour cela nous utilisons le fichier "gaussToto.py" écrit dans la partie 1 qui ne doit contenir que les définitions de fonctions. Commencer par écrire dans un nouveau fichier la ligne `from gaussToto import *`.

1. Ouvrir le fichier "systemes.txt" en mode lecture.
2. La valeur du premier entier donne le nombre de lignes à lire pour récupérer la matrice du premier système, matrice qu'il faut stocker dans une variable `m`.
3. Résoudre alors le système avec la fonction `gauss` et afficher la solution.
4. Le programme doit résoudre tous les systèmes sans intervention de l'utilisateur.