Informatique Appliquée au Calcul Scientifique 2
Séance 12

Résolution de système linéaire : méthode de Gauss

Table des matières

I. TP8 : Méthode d'élimination de Gauss 2
II. Méthode de Gauss 3
III. TP8 suite 5

I. TP8 : Méthode d'élimination de Gauss

Il s'agit de développer un programme permettant de transformer un système matriciel $A.X = B$ en un système équivalent $\tilde{A}.X = \tilde{B}$ où $\tilde{A}$ est une matrice triangulaire supérieur déterminée à partir de la matrice A puis un autre système équivalent $\tilde{\tilde{A}}.X = \tilde{\tilde{B}}$ où cette fois-ci $\tilde{\tilde{A}}$ est une matrice diagonale.

1. La méthode à la main :
Principe :
La méthode du pivot de Gauss transforme un système d'équations linéaires en une forme **échelonnée réduite** (ou triangulaire) par des opérations sur les lignes (addition, multiplication, échange). L'objectif est d'éliminer les variables une à une pour obtenir un système facile à résoudre par **substitution arrière**. Les étapes sont :

a. Écrire le système sous forme de **matrice augmentée**.
b. Utiliser des opérations sur les lignes pour mettre la matrice en forme **triangulaire supérieure**.
c. Résoudre par substitution arrière ou continuer pour obtenir la forme échelonnée réduite.

Opérations permises :

- Échanger deux lignes.
- Multiplier une ligne par un scalaire non nul.
- Ajouter un multiple d'une ligne à une autre.

Résolvons le système suivant avec la méthode du pivot de Gauss :

$$\begin{cases} x + 2y + 3z = 1 \\ 2x + 2y - 4z = 2 \\ 4x - y + z = 3 \end{cases}$$

…

En pratique, le système s'écrit sous forme matricielle $A\,X\ =\ B$, où :

$A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 2 & -4 \\ 4 & -1 & 1 \end{pmatrix}$, $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ et $B = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$.

La matrice augmentée combine A et B :

$$G = \begin{pmatrix} 1 & 2 & 3 & | & 1 \\ 2 & 2 & -4 & | & 2 \\ 4 & -1 & 1 & | & 3 \end{pmatrix}$$

Nous allons mettre la matrice A en forme triangulaire supérieur.

$$G = \begin{pmatrix} 1 & 2 & 3 & | & 1 \\ 2 & 2 & -4 & | & 2 \\ 4 & -1 & 1 & | & 3 \end{pmatrix} \longrightarrow G_1 = \begin{pmatrix} 1 & 2 & 3 & | & 1 \\ 0 & -2 & -10 & | & 0 \\ 0 & -9 & -11 & | & -1 \end{pmatrix} \longrightarrow G_2 = \begin{pmatrix} 1 & 2 & 3 & | & 1 \\ 0 & -2 & -10 & | & 0 \\ 0 & 0 & 34 & | & -1 \end{pmatrix}$$

$$\longrightarrow G_3 = \begin{pmatrix} 1 & 2 & 3 & | & 1 \\ 0 & -2 & -10 & | & 0 \\ 0 & 0 & 1 & | & \frac{-1}{34} \end{pmatrix} \longrightarrow G_4 = \begin{pmatrix} 1 & 2 & 3 & | & 1 \\ 0 & -2 & 0 & | & \frac{-5}{17} \\ 0 & 0 & 1 & | & \frac{-1}{34} \end{pmatrix} \longrightarrow G_5 = \begin{pmatrix} 1 & 2 & 3 & | & 1 \\ 0 & 1 & 0 & | & \frac{5}{34} \\ 0 & 0 & 1 & | & \frac{-1}{34} \end{pmatrix}$$

$$\longrightarrow G_6 = \left(\begin{array}{ccc|c} 1 & 0 & 0 & \frac{27}{34} \\ 0 & 1 & 0 & \frac{5}{34} \\ 0 & 0 & 1 & \frac{-1}{34} \end{array}\right)$$

On a ainsi les solutions de notre système : $x = \frac{27}{34}, y = \frac{5}{34}$ et $z = \frac{-1}{34}$.

Réalisez avec cette méthode la résolution du système suivant :

$$\begin{cases} x + y + 2z = -1 \\ 2x - y + 2z = -4 \\ 4x + y + 4z = -2 \end{cases}$$

2. À partir des notations suivantes :

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n} \end{pmatrix} \; ; \; X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \; ; \; B = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$$

Écrire les calculs qui permettent d'annuler tous les coefficients de la première colonnes sauf $a_{1,1}$.

3. À l'aide du 2., écrire le programme Python permettant de déterminer les matrices $\tilde{A}$ et $\tilde{B}$ (vous pourrez travailler avec la matrice augmentée).

4. À l'aide du 3, écrire le programme qui permet d'obtenir $\tilde{\tilde{A}}$ et $\tilde{\tilde{B}}$ et trouver la matrice X(vous pourrez travailler avec la matrice augmentée).

5. Application numérique :

$$A = \begin{pmatrix} 2 & -5 & 1 & 3 \\ 4 & 7 & 8 & 2 \\ 3 & 1 & 1 & 6 \\ 4 & 1 & 7 & 9 \end{pmatrix} \; ; \; X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \; ; \; B = \begin{pmatrix} 5 \\ 10 \\ 2 \\ 6 \end{pmatrix}$$

Point Python :
Pour définir une matrice, vous pourrez utiliser numpy comme suit :

```
import numpy as np
A = np.array([[2, -5, 1, 3],
              [4, 7, 8, 2],
              [3, 1, 1, 6],
              [4, 1, 7, 9]], dtype=float)
```

II. Méthode de Gauss

On souhaite résoudre le système linéaire suivant :

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1 \\ \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \cdots + a_{n,n}x_n = b_n \end{cases}.$$

On peut écrire ce système sous forme matricielle $A.X = B$ avec :

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n} \end{pmatrix} \; ; \; X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \; ; \; B = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}.$$

Les calculs qui suivent illustrent l'idée de la méthode de Gauss : opérer par combinaisons linéaires sur les lignes L_i du système pour obtenir un système triangulaire.

Une première étape consiste à éliminer x_1 à partie de la deuxième ligne :

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,n}x_n = b_2 \\ \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \cdots + a_{n,n}x_n = b_n \end{cases}$$

$$\Leftrightarrow \begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1 \\ (a_{2,2} - \frac{a_{2,1}.a_{1,2}}{a_{1,1}})x_2 + (a_{2,3} - \frac{a_{2,1}.a_{1,3}}{a_{1,1}})x_3 + \cdots + (a_{2,n} - \frac{a_{2,1}.a_{1,n}}{a_{1,1}})x_n = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1 \\ \vdots \\ (a_{n,2} - \frac{a_{2,1}.a_{n,2}}{a_{1,1}})x_2 + \cdots + (a_{n,n} - \frac{a_{2,1}.a_{n,n}}{a_{1,1}})x_n = b_n - \frac{a_{n,1}}{a_{1,1}} b_1 \end{cases}$$

Ces opérations, qui supposent $a_{1,1} \neq 0 *$ ($a_{1,1}$ est un pivot), consistent à remplacer les lignes L_i pour $i \geq 2$ par la ligne $L_i - \frac{a_{i,1}}{a_{1,1}} L_1$.

Au niveau matriciel, on obtient les matrices $A^{(2)}$ et $B^{(2)}$ comme suit :

$$A^{(2)} = \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ 0 & a_{2,2} - \frac{a_{2,1}.a_{1,2}}{a_{1,1}} \quad \cdots & a_{2,n} - \frac{a_{2,1}.a_{1,n}}{a_{1,1}} \\ \vdots & \vdots & \vdots \\ 0 & a_{n,2} - \frac{a_{2,1}.a_{n,2}}{a_{1,1}} \quad \cdots & a_{n,n} - \frac{a_{2,1}.a_{n,n}}{a_{1,1}} \end{pmatrix} \; ; \; B^{(2)} = \begin{pmatrix} b_1 \\ b_2 - \frac{a_{2,1}}{a_{1,1}} b_1 \\ \vdots \\ b_n - \frac{a_{n,1}}{a_{1,1}} b_1 \end{pmatrix}$$

On obtient des matrices de la forme :

$$A^{(2)} = \begin{pmatrix} a_{1,1}^{(2)} & \cdots & a_{1,n}^{(2)} \\ \vdots & \ddots & \vdots \\ a_{n,1}^{(2)} & \cdots & a_{n,n}^{(2)} \end{pmatrix} \; ; \; B^{(2)} = \begin{pmatrix} b_1^{(2)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{pmatrix}$$

On recommence en remplaçant les lignes L_i pour $i \geq 3$ avec $a_{2,2}^{(2)} \neq 0 *$, par la ligne $L_i - \frac{a_{i,2}^{(2)}}{a_{2,2}^{(2)}} L_2$.

Et ainsi de suite jusqu'à obtenir une matrice triangulaire.

$$A^{(n)} = \begin{pmatrix} a_{1,1}^{(n)} & \cdots & & a_{1,n}^{(n)} \\ 0 & a_{2,2}^{(n)} & \cdots & a_{2,n}^{(n)} \\ \vdots & 0 & & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & a_{n,n}^{(n)} \end{pmatrix}$$

À partir de là, on peut trouver la valeur de x_n : $\frac{b_n^{(n)}}{a_{n,n}^{(n)}}$ et en déduire $x_{n-1}, x_{n-2}, \ldots, x_0$.

Où, poursuivre les opérations pour transformer $A^{(n)}$ en matrice diagonale.
Ainsi, on va commencer par la dernière colonne, et chercher à annuler tous les coefficients sauf $a_{n,n}^{(n)}$:
Pour tous les L_i, pour $i < n$ avec $a_{n,n}^{(n)} \neq 0$, on la remplace par : $L_i - \frac{a_{i,n}^{(n)}}{a_{n,n}^{(n)}} L_n$.
Et ainsi de suie jusqu'à obtenir une matrice diagonale.

***Remarque :**
En condition de chaque combinaison linéaire, nous avons vu qu'il fallait que le pivot soit non nul afin de pouvoir diviser par celui-ci.
Si ce n'est pas le cas, une permutation entre ligne permet de palier à ce problème.

III. TP8 suite

Reprenez le programme trouver dans le I., afin d'ajouter un test judicieusement placer qui permettra une permutation pour éviter le problème du pivot nul.