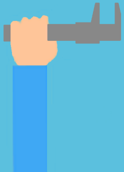
Pivot de Gauss

Renaud Costadoat

Lycée Dorian



Système d'équations

Soit le système d'équations suivant :

$$\left\{\begin{array}{l} y_0 = a_{0,0}.x_0 + a_{0,1}.x_1 + a_{0,2}.x_2 + \ldots + a_{0,n-1}.x_{n-1} \\ y_1 = a_{1,0}.x_1 + a_{1,1}.x_2 + a_{1,2}.x_3 + \ldots + a_{1,n-1}.x_{n-1} \\ \ldots \\ y_{n-1} = a_{n-1,0}.x_0 + a_{n-1,1}.x_1 + a_{n-1,2}.x_2 + \ldots + a_{n-1,n-1}.x_{n-1} \end{array}\right.$$

Ce système d'équations peut s'écrire sous la forme suivante: $Y = A.X$, avec:

$$X = \begin{pmatrix} x_0 \\ x_1 \\ \ldots \\ x_{n-1} \end{pmatrix}, Y = \begin{pmatrix} y_0 \\ y_1 \\ \ldots \\ y_{n-1} \end{pmatrix}, A = \begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & \ldots & a_{0,n-1} \\ a_{1,0} & a_{1,1} & a_{1,2} & \ldots & a_{1,n-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{n-1,0} & a_{n-1,1} & a_{n-1,2} & \ldots & a_{n-1,n-1} \end{pmatrix}$$

Ces systèmes sont dit de Cramer, c'est à dire qu'ils admettent une unique solution. La matrice A est donc inversible.

Cas triangulaire

Si la matrice A est triangulaire, la résolution du problème est simple.

Ce système d'équations peut s'écrire sous la forme suivante: $Y = A.X$, avec:

$$A = \begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & \dots & a_{0,n-1} \\ 0 & a_{1,1} & a_{1,2} & \dots & a_{1,n-1} \\ 0 & 0 & a_{2,2} & \dots & a_{2,n-1} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & a_{n-1,n-1} \end{pmatrix}$$

La dernière équation devient: $y_{n-1} = a_{n-1,n-1}.x_{n-1}$, ainsi $x_{n-1} = \dfrac{y_{n-1}}{a_{n-1,n-1}}$. En remontant tout le système l'ensemble des x_i peuvent être déterminés.

Ainsi, $\forall i \in [0, n-2], x_i = \dfrac{y_i - \sum\limits_{j=i+1}^{n-1} a_{i,j}.x_j}{a_{i,i}}$

Cas triangulaire

Soit la matrice triangulaire A est le vecteur Y suivant:

$$A = \begin{pmatrix} 2 & 5 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 4 \end{pmatrix}, Y = \begin{pmatrix} 6 \\ 4 \\ 9 \end{pmatrix}$$

Cela correspond au système d'équations suivant :

$$\begin{cases} 6 = 2.x_0 + 5.x_1 + 3.x_2 \\ 4 = 1.x_1 + 2.x_2 \\ 9 = 4.x_2 \end{cases}$$

Cela mène à la résolution suivante :

$$\begin{cases} x_0 = \\ x_1 = \\ x_2 = \end{cases}$$

Cas triangulaire

Soit la matrice triangulaire A est le vecteur Y suivant:

$$A = \begin{pmatrix} 2 & 5 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 4 \end{pmatrix}, Y = \begin{pmatrix} 6 \\ 4 \\ 9 \end{pmatrix}$$

Cela correspond au système d'équations suivant :

$$\begin{cases} 6 = 2.x_0 + 5.x_1 + 3.x_2 \\ 4 = 1.x_1 + 2.x_2 \\ 9 = 4.x_2 \end{cases}$$

Cela mène à la résolution suivante :

$$\begin{cases} x_0 = 0.875 \\ x_1 = -0.5 \\ x_2 = 2.25 \end{cases}$$

Cas triangulaire: Code Python

Cas triangulaire: Code Python

```
def triangle(A,b):
    n =len(b)
    x = [0 for i in range(n)]
    x[n-1] = b[n-1]/A[n-1][n-1]
    for i in range(n-2,-1,-1):
        s = 0
        for j in range(i+1,n):
            s = s + A[i][j]*x[j]
        x[i] = (b[i] - s)/ A[i][i]
    return x
```

Etape 1: Placer le pivot sur la première ligne

Soit la matrice $A = \begin{pmatrix} 2 & 2 & -3 & 4 \\ -2 & -1 & -3 & 2 \\ 6 & 4 & 4 & 5 \\ 1 & 5 & 3 & 2 \end{pmatrix}, Y = \begin{pmatrix} 2 \\ -5 \\ 16 \\ 11 \end{pmatrix}$

1. Recherche de la plus grande valeur $A[0]_max$ de la première colonne de la matrice: ici $A[2][0] = 6$,
2. Inverser cette ligne et la première de la matrice et faire de même pour le vecteur.

$$A = \begin{pmatrix} 6 & 4 & 4 & 5 \\ -2 & -1 & -3 & 2 \\ 2 & 2 & -3 & 4 \\ 1 & 5 & 3 & 2 \end{pmatrix}, Y = \begin{pmatrix} 16 \\ -5 \\ 2 \\ 11 \end{pmatrix}$$

Etape 2: Mettre des 0 sur le reste de la première colonne

Soit la matrice $A = \begin{pmatrix} 6 & 4 & 4 & 5 \\ -2 & -1 & -3 & 2 \\ 2 & 2 & -3 & 4 \\ 1 & 5 & 3 & 2 \end{pmatrix}, Y = \begin{pmatrix} 16 \\ -5 \\ 2 \\ 11 \end{pmatrix}$

Pour chaque ligne sauf la première

1. Calcul du coefficient $x = A[i][0]/A[0][0]$,
2. Calculer les nouveaux coefficients qui permettent d'annuler le premier: $A[i][k] = A[i][k] - x * A[j][k]$

$$A = \begin{pmatrix} 6 & 4 & 4 & 5 \\ & & & \\ & & & \\ & & & \end{pmatrix}, Y = \begin{pmatrix} 16 \\ 0.33 \\ -3.33 \\ 8.33 \end{pmatrix}$$

Etape 2: Mettre des 0 sur le reste de la première colonne

Soit la matrice $A = \begin{pmatrix} 6 & 4 & 4 & 5 \\ -2 & -1 & -3 & 2 \\ 2 & 2 & -3 & 4 \\ 1 & 5 & 3 & 2 \end{pmatrix}, Y = \begin{pmatrix} 16 \\ -5 \\ 2 \\ 11 \end{pmatrix}$

Pour chaque ligne sauf la première

1. Calcul du coefficient $x = A[i][0]/A[0][0]$,
2. Calculer les nouveaux coefficients qui permettent d'annuler le premier: $A[i][k] = A[i][k] - x * A[j][k]$

$$A = \begin{pmatrix} 6 & 4 & 4 & 5 \\ 0 & 0.33 & -1.66 & 3.66 \\ 0 & 0.66 & -4.33 & 2.33 \\ 0 & 4.33 & 2.33 & 1.16 \end{pmatrix}, Y = \begin{pmatrix} 16 \\ 0.33 \\ -3.33 \\ 8.33 \end{pmatrix}$$

Etape 3: Mettre des 0 sur le reste de la première colonne

La procédure continue avec la matrice de dimension juste inférieure

$$A = \begin{pmatrix} 0.33 & -1.66 & 3.66 \\ 0.66 & -4.33 & 2.33 \\ 4.33 & 2.33 & 1.16 \end{pmatrix}, Y = \begin{pmatrix} 0.33 \\ -3.33 \\ 8.33 \end{pmatrix}$$

Le résultat de l'étape de mise sous forme triangulaire est alors:

$$A = \begin{pmatrix} 6 & 4 & 4 & 5 \\ & & & \\ & & & \\ & & & \end{pmatrix}, Y = \begin{pmatrix} 16 \\ 8.33 \\ -4.61 \\ 1.5 \end{pmatrix}$$

Etape 3: Mettre des 0 sur le reste de la première colonne

La procédure continue avec la matrice de dimension juste inférieure

$$A = \begin{pmatrix} 0.33 & -1.66 & 3.66 \\ 0.66 & -4.33 & 2.33 \\ 4.33 & 2.33 & 1.16 \end{pmatrix}, Y = \begin{pmatrix} 0.33 \\ -3.33 \\ 8.33 \end{pmatrix}$$

Le résultat de l'étape de mise sous forme triangulaire est alors:

$$A = \begin{pmatrix} 6 & 4 & 4 & 5 \\ 0 & 4.33 & 2.33 & 1.16 \\ 0 & 0 & -4.69 & 2.15 \\ 0 & 0 & 0 & 2.73 \end{pmatrix}, Y = \begin{pmatrix} 16 \\ 8.33 \\ -4.61 \\ 1.5 \end{pmatrix}$$

Il ne reste plus qu'à utiliser la méthode décrite précédemment afin de déterminer le résultat.

Le résultat final pour ces valeurs est alors: $X = \begin{pmatrix} 0.642642642643 \\ 1.10810810811 \\ 1.23723723724 \\ 0.552552552553 \end{pmatrix}$

Etude 1: Code Python

Recherche du pivot et permutation des lignes:

Etude 1: Code Python

Recherche du pivot et permutation des lignes:

```
def recherche_pivot(A, j0):
    n = len(A)
    imax = j0
    for i in range(j0+1, n):
        if abs(A[i][j0]) > abs(A[imax][j0]):
            imax = i
    return imax
```

```
def permutation(A,i,j):
    t = A[i]
    A[i] = A[j]
    A[j] = t
```

Etude 2: Code Python

Transvection:

Etude 2: Code Python

Transvection:

```
def transvection(A,i,j,x):
    # si vecteur
    if type(A[0]) != list:
        A[i] = A[i] + x*A[j]
    else:
        n = len(A[0])
        for k in range(n):
            A[i][k] = A[i][k] + x*A[j][k]
```

Etude 3: Code Python

Résolution du système:

Etude 3: Code Python

Résolution du système:

```
def resolution_systeme(A,b):
    n = len(A)

    for i in range(n-1):
        i0 = recherche_pivot(A,i)
        permutation(A,i,i0)
        permutation(b,i,i0)

        for k in range(i+1,n):
            x = A[k][i]/A[i][i]
            transvection(A,k,i,-x)
            transvection(b,k,i,-x)

    return triangle(A,b)
```