ALGÈBRE LINÉAIRE

Projet 2 : Calcul de A^{-1}

Auteurs
Brenton Ethan

March 6, 2024

Contents

1 Probabilité d'inversibilité d'une matrice $\mathrm{M}_{10}(\llbracket 0, 5 \rrbracket)$

L'idée principal est de générer des matrices aléatoires de taille 10×10 avec des valeurs entre 0 et 5. Ensuite, on vérifie si le déterminant de la matrice est non-nul. Si oui, on incrémente le compteur. On répète cette procédure 10000 fois.

Cette partie va prendre beaucoup de temps, approximativement 80s.

```python
A = sp.Matrix(randint(0, 6, size=(10, 10))) # On prendre des valeurs entre 0 et 5,
    de taille 10 x 10

time_init = time.time()
MAX_FOIS = 10000
count = 0

for fois in range(MAX_FOIS+1):
    A = sp.Matrix(randint(0, 5, size=(10, 10)))
    if A.det() != 0:
        count += 1

    time_end = time.time()

    if fois % 1000 == 0:
        print(f"Temps: {time_end - time_init}s, on a {count} matrices inversibles
            sur {fois+1}")
```

Cette partie de code nous donne le résultat suivant :

```
Temps: 79.15702891349792s, on a 10001 matrices inversibles sur 10001
```

En général, presque toutes les matrices sont inversibles. On peut donc conclure statistiquement que :

Inversibilité d'une matrice de $\mathrm{M}_{10}(\llbracket 0, 5 \rrbracket)$

La probabilité d'inversibilité d'une matrice $\mathrm{M}_{10}(\llbracket 0, 5 \rrbracket)$ est très proche de 1.

2 Comparaison de 4 méthodes pour calculer l'inverse d'une matrice

2.1 Utilisation de la formule

$$A^{-1} = \frac{1}{\det(A)} \cdot^t \operatorname{Com}(A)$$

Globalement, dans tous les questions, on définit C le nombre d'opérations totales, initialement $C = 0$. Spécialement, C est le nombre maximale d'opérations.

On va disiver cette question en deux parties : calculation de $\det(A)$ et calculation de ${}^t\operatorname{Com}(A)$.

2.1.1 Calculation du $\det(A)$, Méthode de Gauss

Dans cette partie, on utilise la **Méthode de Gauss** qui peut transform A en une matrice triangulaire, ensuite multiplie les éléments sur le diagonal.
Méthode de Gauss

Nous allons procéder ligne par ligne, en vérifiant d'abord si le pivot est 0, puis en traitant les autres lignes ensuite. On répète ce processus pour chaque ligne. Donc, pour chaque ligne $i \in [\![1, 10]\!]$, on fait :

- On vérifie si le pivot $a_{i,i}$ est 0.
  Si oui, on échange la ligne avec une autre ligne qui a un élément non-nul dans la même colonne. On fait la même chose pour les autres lignes.

  Notice

  Lorsque on échange deux lignes, le signe du déterminant change. Il faut donc compter le nombre de changement de signe.
  Le maximum de changement de signe est 9 fois. Mais, en général, il est plus petit que 9. Donc, nous le séparons dans nos calculs.
  (Lorsque le premier ligne est

  $$[0, 0, 0, 0, 0, 0, 0, 0, 0, N]$$

  et en même temps la matrice est inversible, il doit être tombé de la ligne 1 à la ligne 10, où N est un nombre entier $\in [\![1, 5]\!]$)

  $$C := C + \underbrace{9}_{\text{changement de signe}}$$

- On est assuré que $a_{i,i} \neq 0$. Donc, en conservant le pivot de la ligne i, on calcul le valeur $R_j = a_{j,i}/a_{i,i}$ pour tout $a_{j,i},\ j \in [\![i + 1, 10]\!]$.

  $$C := C + (10 - i)$$

- On multiplie chaque ligne par R_j. Pour tout $a_{j,k},\ j \in [\![i+1, 10]\!],\ k \in [\![1, 10]\!]$, on fait $a_{j,k} = a_{j,k} - R_j \cdot a_{i,k}$.
$$C := C + 2 \times (10 - i) \times (11 - i)$$

Dans cette partie, le nombre d'opération est donc :

$$\begin{aligned} C &:= C + \sum_{i=1}^{10} ((10-i) + 2 \times (10-i) \times (11-i)) + \underbrace{9}_{\text{changement de signe}} \\ &:= C + 705 + \underbrace{9}_{\text{changement de signe}} \end{aligned}$$

```python
sum = 0
for i in range(1, 11):
    sum += (10-i) + 2 * (10-i) * (11-i)

print(sum)
```

Le résultat est :

```
705
```

2.1.2 Calculation du $\det(A)$, Multiplication des éléments

Multiplication des éléments

Enfin, on multiplie tous les éléments de la diagonale de la matrice triangulaire.

$$C := C + 9$$

2.1.3 Calculation de ${}^t\mathbf{Com}(A)$

On utilise la formule suivante pour calculer ${}^t\mathrm{Com}(A)$:

$$\mathrm{Com}(A) = \begin{pmatrix} C_{1,1} & C_{1,2} & \cdots & C_{1,10} \\ C_{2,1} & C_{2,2} & \cdots & C_{2,10} \\ \vdots & \vdots & \ddots & \vdots \\ C_{10,1} & C_{10,2} & \cdots & C_{10,10} \end{pmatrix}$$

Où $C_{i,j}$ est le cofacteur de $a_{i,j}$, qui est égal à $(-1)^{i+j} \cdot \det(A_{i,j})$. En total il y a $10 \times 10 = 100$ matrices de taille 9×9. Donc,

$$C := C + 2$$

Le nombre d'opération peut être exprimée comme :

Dans cette partie, le nombre d'opération est :

$$
\begin{aligned}
C &:= C + 100 \times \left(\sum_{i=1}^{9} ((9-i) + 2 \times (9-i) \times (10-i)) + 8 + 2 + \underbrace{8}_{\text{changement de signe}} \right) \\
&:= C + 52600 + \underbrace{800}_{\text{changement de signe}}
\end{aligned}
$$

```python
sum = 0
for i in range(1, 10):
    sum = sum + (9-i) + 2 * (9-i) * (10-i)

sum += 10
sum
```

Le résultat est :

```
526
```

2.1.4 Division par $\det(A)$

Rappelons que on doit diviser chaque élément dans la transposée de la matrice des cofacteurs par $\det(A)$:

On doit ajouter 100 opérations pour la division.

$$C := C + 10 \times 10 = C + 100$$

2.1.5 Conclusion

Conclusion

Le nombre d'opération total est donc :

$$
\begin{aligned}
C &= \sum_{i=1}^{10} ((10-i) + 2 \times (10-i) \times (11-i)) + 9 + \underbrace{9}_{\text{changement de signe}} \\
&\quad + 100 \times \left(\sum_{i=1}^{9} ((9-i) + 2 \times (9-i) \times (10-i)) + 8 + 2 \right) + \underbrace{800}_{\text{changement de signe}} \\
&\quad + 100 \\
&= 53414 + \underbrace{809}_{\text{changement de signe}}
\end{aligned}
$$

2.2 Résolution par l'algorithme de Gauss

2.2.1 Augmentation de la matrice

Dans la **Méthode de Gauss**, on fera des opérations sur la matrice augmentée suivante:

$$\left(\begin{array}{cccc|cccc} a_{1,1} & a_{1,2} & ... & a_{1,10} & 1 & 0 & ... & 0 \\ a_{2,1} & a_{2,2} & ... & a_{2,10} & 0 & 1 & ... & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ a_{10,1} & a_{10,2} & ... & a_{10,10} & 0 & 0 & ... & 1 \end{array}\right)$$

On a besoin de transformer la partie gauche en une matrice unitée.

2.2.2 Transformation des parties

PARTIE A : Le nombre d'opération pour la matrice de la partie gauche.

- Premièrement, on transforme la partie gauche en une matrix triangulaire supérieure utilisant la méthode de Gauss.
  On a vu dans la partie 2.1.1, pour une 10×10 matrice, on pourra faire au plus

(On ne compte pas la signe ici, car on ne utilisera pas cette fonction pour calculer le déterminant)

$$C := C + \sum_{i=1}^{10} [(10-i) + 2 \times (10-i) \times (11-i)] = C + 705$$

PARTIE B : Le nombre d'opération pour la matrice de la partie droite. On note la partie droite comme

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} = \begin{pmatrix} b_{1,1} & b_{1,2} & \dots & b_{1,10} \\ b_{2,1} & b_{2,2} & \dots & b_{2,10} \\ \vdots & \vdots & \ddots & \vdots \\ b_{10,1} & b_{10,2} & \dots & b_{10,10} \end{pmatrix}$$

On prend l'exemple suivant pour expliquer la transformation de la partie droite : Supposons que l'on a fait la transformation $L_i \leftarrow L_i - L_1 \times R_i$ pour tous les i lignes.

- Pour la première *colonne à droite*, il deviendra :

$$\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \implies \begin{bmatrix} 1 \\ 0 - 1 \times \frac{a_{2,1}}{a_{1,1}} \\ \vdots \\ 0 - 1 \times \frac{a_{10,1}}{a_{1,1}} \end{bmatrix} = \begin{bmatrix} 1 \\ -\frac{a_{2,1}}{a_{1,1}} \\ \vdots \\ -\frac{a_{10,1}}{a_{1,1}} \end{bmatrix} = \begin{bmatrix} b_{1,1} \\ b_{2,1} \\ \vdots \\ b_{10,1} \end{bmatrix}$$

Alors,

$$C := C + 9 \times 2 = C + 18$$

Pour la deuxième *colonne à droite*, on fait la même chose, mais on a besoin de faire une opération supplémentaire, c'est-à-dire, pour chaque ligne $i \in [\![3, n]\!]$, on fait une opération $L_i \to L_i - L_2 \times R_i$, il deviendra :

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \implies \begin{bmatrix} 0 \\ 1 \\ 0 - 1 \times \frac{a_{3,2}}{a_{2,2}} \\ \vdots \\ 0 - 1 \times \frac{a_{10,2}}{a_{2,2}} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ -\frac{a_{3,2}}{a_{2,2}} \\ \vdots \\ -\frac{a_{10,2}}{a_{2,2}} \end{bmatrix} = \begin{bmatrix} 0 \\ b_{2,2} \\ b_{3,2} \\ \vdots \\ b_{10,2} \end{bmatrix}$$

Alors,

$$C := C + 8 \times 2 = C + 16$$

- En résumé, quand on fait une transformation à la première colonne de la matrice à gauche, le nombre d'opération s'écrit comme :

$$C := C + \sum_{i=1}^{10} (10 - i) \times 2$$

- Cela implique, lorsque l'on a fini la transformation du triangulaire supérièure à la matrice à gauche, en totale, le nombre d'opération vaut :

$$\begin{aligned} C &:= C + \sum_{i=1}^{10} (10 - i) \times 2 + \sum_{i=1}^{9} (9 - i) \times 2 + \ldots + 2 \\ &:= C + \sum_{n=1}^{10} \sum_{i=1}^{n} 2 \times (n - i) = C + 330 \end{aligned}$$

```python
count = 0
for n in range(1, 10+1):
    for i in range(1, n+1):
        count += 2*(n-i)
count # Rsultat : 330
```

Enfin, on obtient la matrice augmentée suivante :

$$\left(\begin{array}{cccc|cccc} a_{1,1} & a_{1,2} & \ldots & a_{1,10} & b_{1,1} & 0 & \ldots & 0 \\ 0 & a_{2,2} & \ldots & a_{2,10} & b_{2,1} & b_{2,2} & \ldots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \ldots & a_{10,10} & b_{10,1} & b_{10,2} & \ldots & b_{10,10} \end{array}\right)$$

2.2.3 Transformation de la partie gauche en une matrice d'identité

- On souhaite de transformer la matrice à la gauche d'une matrice triangulaire supérieure diagonale à une matrice diagonale. C'est l'inverse de ce que l'on a fait dans la partie B pour la matrice de la partie droite, voir 2.2.2.

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,10} \\ 0 & a_{2,2} & \dots & a_{2,10} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & a_{10,10} \end{pmatrix} \Longrightarrow \begin{pmatrix} a'_{1,1} & 0 & \dots & 0 \\ 0 & a'_{2,2} & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & a'_{10,10} \end{pmatrix}$$

$$C := C + \sum_{n=1}^{10} \sum_{i=1}^{n} 2(n-i) = C + 330$$

- Puis, on va calculer les opérations fait pour la matrice à la droite. Chaque élément $b_{i,j} \leftarrow b_{i,j} - b_{10,j} \times R_i$. Cette méthode est la réverse de la **Méthode de Gauss**, cette fois-ci, on a transformé une matrice triangulaire inférieur en une matrice de taille 10×10. On obtient immédiatement :

$$\begin{pmatrix} b_{1,1} & 0 & \dots & 0 \\ b_{2,1} & b_{2,2} & \dots & 0 \\ \vdots & \vdots & & \vdots \\ b_{10,1} & b_{10,2} & \dots & b_{10,10} \end{pmatrix} \Longrightarrow \begin{pmatrix} b'_{1,1} & b'_{1,2} & \dots & b'_{1,10} \\ b'_{2,1} & b'_{2,2} & \dots & b'_{2,10} \\ \vdots & \vdots & & \vdots \\ b'_{10,1} & b'_{10,2} & \dots & b'_{10,10} \end{pmatrix}$$

$$C := C + \sum_{i=1}^{10} ((10-i) + 2 \times (10-i) \times (11-i)) = C + 705$$

- Finalement, on divise la matrice de la droite par la matrice de la gauche, c'est-à-dire :

$$L_i \leftarrow L_i / a_{i,i},\ \forall i \in [\![1, 10]\!]$$

$$C := C + 10 \times 10 = C + 100$$

2.2.4 Conclusion

En sommant les 3 parties,

Conclusion

Le nombre d'opération total est donc :

$$C = 705 + 330 + 705 + 330 + 100 = 2170$$

2.3 Trigonalisation

On résoud les équations linéaires suivantes :

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,10} \\ a_{2,1} & a_{2,2} & \dots & a_{2,10} \\ \vdots & \vdots & & \vdots \\ a_{10,1} & a_{10,2} & \dots & a_{10,10} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{10} \end{pmatrix} = I_{10} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}, \dots \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}$$

C'est similaire que la partie précédente parce que la nature du pivot de Gauss est la résolution de ces éequations.

2.3.1 Transformation de la matrice à gauche

Dans cette partie, on fera des opérations sur la matrice comme la méthode précédente (on utilisera encore la méthode de Gauss pour la transformer en une matrix triangulaire supérieure):

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,10} \\ a_{2,1} & a_{2,2} & \dots & a_{2,10} \\ \vdots & \vdots & & \vdots \\ a_{10,1} & a_{10,2} & \dots & a_{10,10} \end{pmatrix} \Longrightarrow \begin{pmatrix} a'_{1,1} & a'_{1,2} & \dots & a'_{1,10} \\ 0 & a'_{2,2} & \dots & a'_{2,10} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & a'_{10,10} \end{pmatrix}$$

Comme la partie 2.1.1, cela nous prend 705 étaps.

$$C := C + 705$$

2.3.2 Transformation de la matrice à droite

- On rappel que on va transformer la partie droite de la première équation:

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,10} \\ a_{2,1} & a_{2,2} & \dots & a_{2,10} \\ \vdots & \vdots & & \vdots \\ a_{10,1} & a_{10,2} & \dots & a_{10,10} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{10} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

  On a vu dans la partie 2.2.2 que quand on fait une transformation à la première colonne de la matrice,

$$\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \Longrightarrow \begin{pmatrix} 1 \\ 0 - 1 \times \frac{a_{2,1}}{a_{1,1}} \\ \vdots \\ 0 - 1 \times \frac{a_{10,1}}{a_{1,1}} \end{pmatrix} = \begin{pmatrix} 1 \\ -\frac{a_{2,1}}{a_{1,1}} \\ \vdots \\ -\frac{a_{10,1}}{a_{1,1}} \end{pmatrix}$$

  Il y aura $2 \times 9 = 18$ opération.

Lorsque on fait des transformations aux autre colonnes de la matrice, il ya aura $\sum_{n=1}^{8} 2n = 72$ opérations.
Donc en transformant la partie droite, on fera $\sum_{n=1}^{9} 2n = 90$ opérations en totale.

$$C := C + \sum_{n=1}^{9} 2n = C + 90$$

On obtient

$$\begin{pmatrix} b_{1,1} \\ b_{2,1} \\ \vdots \\ b_{10,1} \end{pmatrix}$$

En suite, on résoud l'équation:

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,10} \\ 0 & a_{2,2} & \dots & a_{2,10} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & a_{10,10} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{10} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{10} \end{pmatrix}$$

.
Donc, on a

$$\begin{aligned} x_{10} &= \frac{b_{10}}{a_{10,10}} \\ x_9 &= \frac{b_9 - a_{9,10} b_{10}}{t_{9,9}} \\ &\vdots \\ x_1 &= \frac{b_1 - \sum_{n=2}^{10} a_{1,n} x_n}{a_{1,1}} \end{aligned}$$

$$C := C + \sum_{n=1}^{10} 2n - 1 = C + 100$$

Donc pour la première équation, on a :

$$\boxed{C := C + \sum_{n=1}^{9} (2n) + \sum_{n=1}^{10} (2n - 1) = C + 90 + 100 = C + 190}$$

- Ensuite, on transforme la partie droite de la deuxième équation :

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,10} \\ a_{2,1} & a_{2,2} & \dots & a_{2,10} \\ \vdots & \vdots & & \vdots \\ a_{10,1} & a_{10,2} & \dots & a_{10,10} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{10} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}$$

Comme l'élément de la premire ligne de $\begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}$ est toujours 0 dans la processus des opérations, alors on doit faire seulement $\sum_{n=1}^{9} 2n = 72$ opérations. De plus, pour résoudre cette équation, on prendra encore 100 étaps comme la cas précédent.
Donc

$$C := C + 72 + 100 = C + 172$$

- Similairement, on peut résoudre les autre 8 équations.

2.3.3 Conclusion

En totale, on a :

Conclusion

La nombre d'opérations :

$$C := C + 705 + 100 \times 10 + \sum_{n=1}^{10} n(n-1) = C + 2035$$

pour résoudre chaque équation.

Ici, $\sum_{m=1}^{n} 2m = n(n-1)$, en effet, chaque solution est une colonne de la matrix A^{-1}.

2.3.4 Autre méthode de trigonalisation

La trigonalisation d'une matrice de taille 10×10, strictement dite, doit être faite par les étapes suivantes :

- On fait la transformation pour A :

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,10} \\ a_{2,1} & a_{2,2} & \dots & a_{2,10} \\ \vdots & \vdots & & \vdots \\ a_{10,1} & a_{10,2} & \dots & a_{10,10} \end{pmatrix} \Longrightarrow \begin{pmatrix} a_{1,1} - \lambda & a_{1,2} & \dots & a_{1,10} \\ a_{2,1} & a_{2,2} - \lambda & \dots & a_{2,10} \\ \vdots & \vdots & & \vdots \\ a_{10,1} & a_{10,2} & \dots & a_{10,10} - \lambda \end{pmatrix}$$

- Par récurrence, à partir de la matrice de taille 2×2, on utilise la formule :

$$\det(A) = \sum_{i=1}^{n} a_{i,j} \text{Cofacteur}_{i,j}(A)$$

pour calculer le déterminant de la matrice $A - \lambda I_n$. Nous trouverons une expression en fonction de λ et de degré 10.

- On trouve un des racines de l'équation caractéristique, et on obtient les valeurs propres de la matrice A. Mais, ce n'est pas facile de trouver une valeur exacte d'une polynome de degré 10, mais on peut trouver les solutions très proche de leur valeurs réels, en utilisant les méthodes par exemple la Bissection.

  On note ce valeur propre λ_1.

- On procède par récurrence, on fera la même chose pour la matrice B :

$$Q^{-1}AQ = \begin{pmatrix} \lambda_1 & * & \dots & * \\ 0 & & & \\ \vdots & & B & \end{pmatrix}$$

- À la fin, on trouve la matrice P telle que $A = P^{-1}TP$.
- On résoud l'équation : $AX = B \implies P^{-1}TPX = B$

Mais, cette méthode existe deux problèmes majeurs :

1. Il demande trop de calcul. En effet, pour calculer le déterminant de la matrice $A - \lambda I_n$ en considérant λ, on monte d'une matrice de taille 2×2 jusqu'à 9×9. On ne sait pas combien de fois que l'on trouve un valeur presque exacte lorsque on est en train de trouver une racine de l'équation caractéristique de degré 10.

2. Ce n'est pas possible de calculer P^{-1}. Notre but est de trouver l'inverse d'une matrice de taille 10×10, il est en conflit avec ce que l'on souhaite à faire.

   Alors, on n'utilise pas cette méthode.

2.4 Décomposition QR

D'après la proposition en cours, en vérifiant l'inversibilité de la matrice A, il existe un unique couple $(O, T) \in \mathrm{O}_p(\mathbb{R}) \times \mathrm{T}_p^+(\mathbb{R})$, T ayant tous ses termes diagonaux > 0, telles que

$$A = O.T$$

2.4.1 Décomposition en dix systèmes linéaires

L'idée principale est la même de la partie 2.3 :

$$\begin{aligned}
&A.B = I_n \text{ avec } B = A^{-1} \\
\iff\ & A.\begin{bmatrix} \vdots & \vdots & \dots & \vdots \\ B_1 & B_2 & \ddots & B_{10} \\ \vdots & \vdots & \dots & \vdots \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \\
\iff\ & A.\begin{bmatrix} \vdots \\ B_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix},\ A.\begin{bmatrix} \vdots \\ B_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix},\ \dots,\ A.\begin{bmatrix} \vdots \\ B_{10} \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \\
\iff\ & A.X_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix},\ \dots,\ A.X_{10} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}
\end{aligned}$$

Donc, on doit résoudre 10 système d'inconnue en totale.

$$C := C_{(S)} \times 10$$

Ici, $C_{(S)}$ représente la nombre d'opération pour chaque système (S).

2.4.2 Méthode de Gram-Schmidt

C'est un procéde qui transforme un système de n vecteurs en un système orthonormé en faisant respectivement des projections sur les axes. On note : $\mathcal{A} = (a_1, \dots, a_{10}) \implies \mathcal{E} = (e_1, \dots, e_{10})$

$$A = \begin{bmatrix} \vdots & \vdots & & \vdots \\ a_1 & a_2 & \dots & a_{10} \\ \vdots & \vdots & & \vdots \end{bmatrix}$$

On rappelle que la fonction de projection d'un vecteur $v \in E$ sur un autre vectueur $u \in E$ est définie comme :

$$p_u : v \mapsto \frac{< u, v >}{|| u ||^2} u$$

On procéde selon les formules :

$$
\begin{aligned}
b_1 &= a_1,\ e_1 = \frac{b_1}{\|\ b_1\ \|} \\
b_2 &= a_2 - p_{e_1}(a_2),\ e_2 = \frac{b_2}{\|\ b_2\ \|} \\
b_3 &= a_3 - p_{e_1}(a_3) - p_{e_2}(a_3),\ e_3 = \frac{b_3}{\|\ b_3\ \|} \\
&\vdots \\
b_{10} &= a_{10} - \sum_{n=1}^{9} p_{e_n}(a_{10}),\ e_{10} = \frac{b_{10}}{\|\ b_{10}\ \|}
\end{aligned}
$$

En résumé, en assurant que la norme de u vaut 1 dans chaque orthomalisation :

Process de l'orthomalisation de Gram-Schmidt

Les choses à faire pour chaque vecteur b_i :

$$
\begin{aligned}
b_1 &= a_1,\ e_1 = \frac{b_1}{\|\ b_1\ \|} \\
b_2 &= a_2 - < a_2, e_1 > .e_1,\ e_2 := \frac{b_2}{\|\ b_2\ \|} \\
b_3 &= a_3 - < a_3, e_1 > .e_1 - < a_3, e_2 > .e_2,\ e_3 := \frac{b_3}{\|\ b_3\ \|} \\
&\vdots \\
b_{10} &= a_{10} - \sum_{n=1}^{9} < a_{10}, e_n > .e_n,\ e_{10} := \frac{b_{10}}{\|\ b_{10}\ \|}
\end{aligned}
$$

- Pour chaque projection, par étape :
  - $< u, v >$: multiplication 10 + sommation 9 = 19 opérations
  - On multiplie la scalaire pour tous les éléments de u : 10 opérations
  - Chaque soustraction vaut 10 opération.

  En totale, la nombre d'opération de chaque projection est :

$$C := C + 19 + 10 + 10 = C + 39$$

- La normation d'un vecteur, par étape :
  - Calculer la norme : multiplication 10 + sommation 9 + $\sqrt{.}$ 1 = 20 opérations

$$\|\ u\ \| = \sqrt{< u, u >}$$

  - Diviser tous les éléments par la norme : 10 opérations

  En totale, la nombre d'opération de chaque normation est :

$$C := C + 20 + 10 = C + 30$$

Donc, cette étape demande en totale :

$$C := C + \sum_{n=1}^{9} 39n + 30 \times 10 = C + 2055$$

2.4.3 Construction de l'équation

De l'orthomalisation de Gram-Schmidt, on obtient :

$$\begin{aligned}
&a_1 = b_1 = \| b_1 \| .e_1 \\
&a_2 = b_2 + < a_2.e_1 > .e_1 = \| b_2 \| .e_2 + < a_2.e_1 > .e_1 \\
&\dots \\
&a_{10} = b_{10} + \sum_{n=1}^{9} < a_{10}, e_n > .e_n = \| b_{10} \| .e_{10} + \sum_{n=1}^{9} < a_{10}, e_n > .e_n
\end{aligned}$$

On peut aisément vérifie que :

$$A = O.T = \begin{bmatrix} \vdots & \vdots & & \vdots \\ a_1 & a_2 & \dots & a_{10} \\ \vdots & \vdots & & \vdots \end{bmatrix} = \begin{bmatrix} \vdots & \vdots & & \vdots \\ e_1 & e_2 & \dots & e_{10} \\ \vdots & \vdots & & \vdots \end{bmatrix} . \begin{bmatrix} \| b_1 \| & < a_2, e_1 > & < a_3, e_1 > & \dots & < a_{10}, e_1 > \\ 0 & \| b_2 \| & < a_3, e_2 > & \dots & < a_{10}, e_2 > \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \| b_{10} \| \end{bmatrix}$$

On veut résoudre $A.X = B$, et on a l'équation $A = O.T$. Donc, on a :

$$O.T.X = B \Leftrightarrow T.X = {}^tO.B$$

La multiplication des 2 matrices doit avoir

$$C := C + 10 \times 10 \times (10 \text{ multiplication} + 9 \text{ sommation}) = C + 1900$$

2.4.4 Résolution d'un système triangulaire

En notant ${}^tO.B = Y$, on veut résoudre $T.X = Y$. On a :

$$\begin{aligned}
x_{10} &= \frac{y_{10}}{t_{10,10}} \\
x_9 &= \frac{y_9 - t_{9,10}x_{10}}{t_{9,9}} \\
&\vdots \\
x_1 &= \frac{y_1 - \sum_{n=2}^{10} t_{1,n}x_n}{t_{1,1}}
\end{aligned}$$

Pour chaque étape, on a :

- Multiplication : $(10 - i)$ fois pour la i-ème équation

- Soustraction : 1 opération
- Calculer la division : 1 opération

Donc, le nombre d'opération pour chaque étape est :

$$C := C + \sum_{n=1}^{10}((n-1) \times 2 + 1) = C + 100$$

2.4.5 Conclusion

Le nombre d'opération total est donc :

Conclusion

La nombre d'opérations :

$$C = (2055 + 1900 + 100) \times 10 = 40550$$

3 Conclusion

d'après les questions précédentes, on a:

nombre maximal d'opération

$A^{-1} = \frac{1}{\det(A)} \cdot {}^{t}\operatorname{Com}(A)$: 54223
L'algorithme de Gauss: 2170
Trigonalisation: 2035
Décomposition QR:40550

Donc la méthode la plus faclie est trigonalisation (et l'algorithme de Gauss, les deux sont similaires).