Algèbre et géométrie vectorielle pour la Data Science

Cours, TD et TP

Quantum horaire :30H

Dr. Abdoulaye MANE

Calcul matriciel avancé

Dans ce cours, nous considérons les $\mathbb{R}$-espaces vectoriels $\mathbb{R}^n$ et $\mathbb{R}^m$ de bases respectives $\mathcal{B}_{\mathbb{R}^n}$ et $\mathcal{B}_{\mathbb{R}^m}$, pour tous $n, m \in \mathbb{N}^*$. Tout élément $X \in \mathbb{R}^n$, alors $X = (x_1, x_2, \cdots, x_n)$ est un vecteur. Soient $X, Y \in \mathbb{R}^n$ deux vecteurs et $\alpha \in \mathbb{R}$ un scalaire, alors $X + Y = (x_1 + y_1, x_2 + y_2, \cdots, x_n + y_n)$, $\alpha X = (\alpha x_1, \alpha x_2, \cdots, \alpha x_n)$ et $\alpha X + Y = (\alpha x_1 + y_1, \alpha x_2 + y_2, \cdots, \alpha x_n + y_n)$.
Les opérations élémentaires des matrices et la programmation en python notamment Sympy et Numpy sont supposées acquises, nous vous suggérons de réviser les cours des semestres passés.

1 Applications linéaires

Definition 1. *Soit* $f : \begin{array}{ccc} \mathbb{R}^n & \to & \mathbb{R}^m \\ X & \mapsto & f(X) \end{array}$ *une application.*
On dit que f *est linéaire si pour tous* $X, Y \in \mathbb{R}^n$ *et pour tout* $\alpha \in \mathbb{R}$*, alors :*

1. $f(X + Y) = f(X) + f(Y)$.
2. $f(\alpha X) = \alpha f(X)$.

Remark 1.

- *D'après 1. et 2. de la définition,* f *est linéaire si et seulement si* $f(\alpha X + Y) = \alpha f(X) + f(Y)$.
- *Pour tout vecteur* $X \in \mathbb{R}^n$*,* $f(X)$ *est un vecteur de* $\mathbb{R}^m$.
- *Si* $n = m$*,* f *est appelée endomorphisme.*

Example 1. $f : \begin{array}{ccc} \mathbb{R}^2 & \to & \mathbb{R}^2 \\ (x, y) & \mapsto & (x + y, 2x - y) \end{array}$ *est une application linéaire.*
En effet, soient $(x_1, y_1), (x_2, y_2) \in \mathbb{R}^2$ *et* $\alpha \in \mathbb{R}$*, alors :*

1. *Méthode 1:*

1.a

$$\begin{aligned} f((x_1,y_1)+(x_2,y_2)) &= f(x_1+x_2,\ y_1+y_2) \\ &= \big((x_1+x_2)+(y_1+y_2),\ 2(x_1+x_2)-(y_1+y_2)\big) \\ &= (x_1+y_1+x_2+y_2,\ 2x_1-y_1+2x_2-y_2) \\ &= (x_1+y_1,\ 2x_1-y_1)+(x_2+y_2,\ 2x_2-y_2) \\ &= f(x_1,y_1)+f(x_2,y_2). \end{aligned}$$

1.b

$$\begin{aligned} f(\alpha(x,y)) &= f(\alpha x,\ \alpha y) \\ &= (\alpha x+\alpha y,\ 2\alpha x-\alpha y) \\ &= \alpha(x+y,\ 2x-y) \\ &= \alpha f(x,y). \end{aligned}$$

2. *Méthode 2:*

$$\begin{aligned} f(\alpha(x_1,y_1)+(x_1',y_1')) &= f(\alpha x_1+x_1',\ \alpha y_1+y_1') \\ &= \big((\alpha x_1+x_1')+(\alpha y_1+y_1'),\ 2(\alpha x_1+x_1')-(\alpha y_1+y_1')\big) \\ &= \big(\alpha(x_1+y_1)+(x_1'+y_1'),\ \alpha(2x_1-y_1)+(2x_1'-y_1')\big) \\ &= \alpha(x_1+y_1,\ 2x_1-y_1)+(x_1'+y_1',\ 2x_1'-y_1') \\ &= \alpha f(x_1,y_1)+f(x_1',y_1'). \end{aligned}$$

Application avec python

[22]:
```python
import sympy as sp
# déclaration des variables
x1, x2, y1, y2, a = sp.symbols('x1 x2 y1 y2 a', real=True)

# définition de la fonction f(x,y)=(x+y, 2x-y)
def f(x,y): return sp.Matrix([x+y, 2*x-y])

# on définit af(x1,y1),  f(x2,y2) et f(ax1+x2, ay1+y2)
u=a*f(x1,y1)
v=f(x2,y2)
AddF=f(a*x1+x2,a*y1+y2)

# On va vérifier si af(x1,y1)+f(x2,y2) - f(ax1+x2, ay1+y2) = 0
sp.simplify(AddF-(u+v))==sp.Matrix([0,0])
```

[22]:
```
True
```

Figure 1: Teste de la linéarité avec python utilisant la méthode 2.

Proposition 1. *Soient $f : \mathbb{R}^n \to \mathbb{R}^m$ et $g : \mathbb{R}^m \to \mathbb{R}^p$ deux applications linéaires. Alors, $g \circ f : \mathbb{R}^n \to \mathbb{R}^p$ est une application linéaire.*

Proposition 2. *Soit $f : \mathbb{R}^n \to \mathbb{R}^m$ une application linéaire. Alors :*

1. *L'image de f, notée $Im(f) = f(\mathbb{R}^n) = \{f(X)|\forall X \in \mathbb{R}^n\}$, est un sous-espace vectoriel de $\mathbb{R}^m$.*
2. *Le noyau de f, noté $Ker(f) = f^{-1}(\{0_{\mathbb{R}^m}\}) = \{X \in \mathbb{R}^n | f(X) = 0_{\mathbb{R}^m}\}$, est un sous-espace vectoriel de $\mathbb{R}^n$.*

Theorem 1. *Soit $f : \mathbb{R}^n \to \mathbb{R}^m$ une application linéaire. Alors :*

1. *f est injective $\Leftrightarrow Ker(f) = \{0_{\mathbb{R}^n}\}$.*
2. *f est surjective $\Leftrightarrow Im(f) = \mathbb{R}^m$.*
3. *f est bijective si elle est, à la fois, injective et surjective.*

Remark 2. *Si f est un endomorphisme bijectif, f est appelé automorphisme.*

Definition 2. *Soit $f : \mathbb{R}^n \to \mathbb{R}^m$ une application linéaire. On appelle rang de f, noté $Rang(f)$, la dimension l'image de f.*

Theorem 2. *Soit $f : \mathbb{R}^n \to \mathbb{R}^m$ une application linéaire. Alors :*

1. *$Rang(f) = dim(Im(f))$,*
2. *$Rang(f) = n - dim(Ker(f))$.*

Méthodes

- *Déterminer $Im(f)$; le $Ran(f)$ est égale à la dimension d'une base de $Im(f)$.*
- *Calculer l'image de la base canonique de $\mathbb{R}^n$ par f, et le rang est égale au nombre de vecteurs libres.*

Example 2. *Considérons $f(x, y) = (x + y, 2x - y)$.*
Déterminer le rang de f et en déduire la dimension de son noyau.

Correction :

1. *On a, par définition, $Im(f) = \{(x + y, 2x - y)|(x, y) \in \mathbb{R}^2\}$.*
   Soit $X \in Im(f)$, alors $X = (x + y, 2x - y) = (x, 2x) + (y, -y) = x(1, 2) + y(1, -1)$.
   Donc $Im(f) = vect\{(1, 2), (1, -1)\}$; ainsi $Rang(f) = dim(Im(f)) = dim(vect\{(1, 2), (1, -1)\}) = 2$.
2. *On a $f : \mathbb{R}^2 \to \mathbb{R}^2$, alors $\{e_1 = (1, 0), e_2 = (0, 1)\}$ est la base canonique de $\mathbb{R}^2$.*

$$f(e_1) = f(1, 0) = (1 + 0, 2 \cdot 1 - 0) = (1, 2),$$
$$f(e_2) = f(0, 1) = (0 + 1, 2 \cdot 0 - 1) = (1, -1).$$

   Vérifions si $\{f(e_1), f(e_2)\}$ est libre.
   Soit $\alpha f(e_1) + \beta f(e_2) = (0, 0) \Rightarrow \alpha(1, 2) + \beta(1, -1) = (0, 0) \Rightarrow \begin{cases} \alpha + \beta = 0 \\ 2\alpha - \beta = 0 \end{cases}$. On trouve $\alpha = \beta = 0$,
   d'où $Rang(f) = 2$.

Application avec python

```
•[17]: import sympy as sp

       # Déclaration des variables
       x, y = sp.symbols('x y')
       def f(x,y) : return sp.Matrix([x + y, 2*x - y])

       # Coefficients de x et y
       f=f(x,y)
       A = f.jacobian([x, y])
```

Déterminons Im(f)

```
[27]: # Base de l'image
      base_im = A.columnspace()
      print("Base de l'image :")
      tuples = [tuple(v[i, 0] for i in range(v.rows)) for v in base_im]

      f"Im(f)=vec{tuples}"
```

```
Base de l'image :
[27]: 'Im(f)=vec[(1, 2), (1, -1)]'
```

Déterminons Rang(f)

```
[29]: rang = A.rank()

      f"Rang(f)={rang}"
```

```
[29]: 'Rang(f)=2'
```

Figure 2: Détermination de Im(f) et Rang(f).

Exercices (TD)

Exercice 1 :

Considérons les applications suivantes :

$f_1(x,y) = (2x, 3y)$, $f_4(x,y,z) = (x+y+z, x-y, y+z)$

$f_2(x,y) = (x+1, y)$, $f_5(x,y) = (x+y, y)$

$f_3(x,y) = (x-y, x+y)$, $f_6(x,y) = (2x, x+y)$.

1. *Lesquelles des applications ci-dessus sont linéaires ? Justifier.*
2. *Déterminer le noyau et l'image de chaque application linéaire parmi les applications ci-dessus.*
3. *En déduire le rang de chaque application linéaire parmi les applications ci-dessus.*
4. *Écrire un code python qui :*
   - *vérifie la linéarité de chacune des applications linéaires parmi les applications ci-dessus.*
   - *permet de déterminer l'image et le rang de chacune des applications linéaires parmi les applications ci-dessus.*

Exercice 2 :

Soient $f(x,y) = (x-y, 2x-3y, x+y)$ et $g(x,y,z) = (x-z, z-y)$ deux applications linéaires.

1. *Déterminer $g \circ f$.*
2. *Montrer que $g \circ f$ est linéaire et puis déterminer son rang.*
3. *Peut-on déterminer $f \circ g$? Justifier.*

2 Matrices d'une applications linéaires

2.1 Rappel-les Matrices

Definition 3. *Une matrice A est un tableau de données numériques, réelles ou complexes, notée par :*

$$A = (a_{ij})_{\substack{1 \le j \le m \\ 1 \le i \le n}} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix}.$$

Si $n = m$, A est dite matrice carrée et sinon, elle est dite rectangulaire.

Remark 3.

1. *La dimension de A est égale à $n \times m$.*
2. *L'ensemble des matrices à coefficients réels de dimension $n \times m$ est $\mathcal{M}_{n,m}(\mathbb{R})$.*

Example 3.

Produit	Région d'origine	Quantité vendue (kg)	Prix unitaire (FCFA)	Chiffre d'affaires (FCFA)
Arachides	Kaolack	250	500	125000
Mangues	Casamance	180	800	144000
Poisson frais	Dakar	300	1000	300000
Riz local	Saint-Louis	400	600	240000
Bissap (fleurs séchées	Thiès	120	700	84000

Figure 3: Tableau de vente hebdomadaire de produits locaux.

Interprétation du tableau

Matrice des données associées au tableau : $A = \begin{pmatrix} 250 & 500 & 125000 \\ 180 & 800 & 144000 \\ 300 & 1000 & 300000 \\ 400 & 600 & 240000 \\ 120 & 700 & 84000 \end{pmatrix}$.

A ***est un élément du*** $\mathbb{R}$***-espace vectoriel*** $\mathcal{M}_{5,3}(\mathbb{R})$, ***l'ensemble des matrices réelles de dimension*** $5 \times 3 = 15$.

Data et extraction de la matrice des data numériques python

```
[7]: import pandas as pd
import numpy as np
import sympy as sp
# ---Création du tableau de données ---
data = {
    "Produit": ["Arachides", "Mangues", "Poisson frais", "Riz local", "Bissap (fleurs séchées)"],
    "Région d'origine": ["Kaolack", "Casamance", "Dakar", "Saint-Louis", "Thiès"],
    "Quantité vendue (kg)": [250, 180, 300, 400, 120],
    "Prix unitaire (FCFA)": [500, 800, 1000, 600, 700],
    "Chiffre d'affaires (FCFA)": [125000, 144000, 300000, 240000, 84000]
}
# Tableau des données brutes
TableOfData = pd.DataFrame(data)

#print(TableOfData)
# ---Conversion en matrice numérique ---
# On ne garde que les colonnes quantitatives
X = TableOfData[["Quantité vendue (kg)", "Prix unitaire (FCFA)", "Chiffre d'affaires (FCFA)"]].to_numpy()
# Affichage de la matrice numérique
print("\nMatrice X des données numériques\n")
print(X)
```

```
Matrice X des données numériques

[[   250    500 125000]
 [   180    800 144000]
 [   300   1000 300000]
 [   400    600 240000]
 [   120    700  84000]]
```

Figure 4: Matrice des données numériques d'un data.

Definition 4. *Une matrice élémentaire $E_{ij} \in \mathcal{M}_{nm}$ dont tous les coefficients sont nuls, sauf celui de la i^{ieme} ligne et de la j^{ieme} colonne qui est égal à* 1.

Example 4. $E_{23} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \in \mathcal{M}_{3,4}(\mathbb{R})$.

Proposition 3. *Soit* $A = (a_{ij})_{\substack{1 \leq j \leq m \\ 1 \leq i \leq n}} \in \mathcal{M}_{n,m}(\mathbb{R})$. *Alors,* $A = \sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij} E_{ij}$.

Example 5. $\begin{pmatrix} 2 & 3 \\ -1 & 4 \end{pmatrix} = 2E_{11} + 3E_{12} - E_{21} + 4E_{44}$.

Remark 4. *La famille* $\{E_{ij}\}_{\substack{1 \leq j \leq m \\ 1 \leq i \leq n}}$ *est une base du* $\mathbb{R}$*-espace vectoriel* $\mathcal{M}_{n,m}(\mathbb{R})$.

Definition 5. *Soit* $A \in \mathcal{M}_{n,m}(\mathbb{R})$ *une matrice. On appelle (matrice) transposée de* A *la matrice obtenue en échangeant les lignes (colonnes) de* A *en colonnes (en lignes).*

Example 6.

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \quad \Rightarrow \quad A^T = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

Proposition 4. *Soient $A, B \in \mathcal{M}_{n,m}(\mathbb{R})$ deux matrices et $\lambda \in \mathbb{R}$ un réel. Alors :*

$$1. \quad (A^T)^T = A \tag{1}$$

$$2. \quad (A+B)^T = A^T + B^T \tag{2}$$

$$3. \quad (\lambda A)^T = \lambda A^T \tag{3}$$

Proposition 5. *Soient $A \in \mathcal{M}_{n,m}(\mathbb{R})$ et $B \in \mathcal{M}_{m,p}(\mathbb{R})$ deux matrices réelle. Alors, $(AB)^T = B^T A^T$.*

Definition 6. *On appelle matrice diagonale toute matrice $D \in \mathcal{M}_n(\mathbb{R})$ telle que $a_{ij} = 0$, si $i \neq j$. On note* $D = \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix}$.

Definition 7. *On appelle matrice identité I_n la matrice définie par $a_{ij} = 0$, si $i \neq j$ et $a_{ii} = 1$, si $i = j$.*

Definition 8. *Soit $A \in \mathcal{M}_n(\mathbb{R})$ une matrice carrée.*

1. *A est une matrice triangulaire supérieure si* $A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix}$, *c'est à dire $a_{ij} = 0$ si $j < i$.*
2. *A est une matrice triangulaire inférieure si* $A = \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$, *c'est à dire $a_{ij} = 0$ si $j > i$.*
3. *A est une matrice symétrique si $A^T = A$.*
4. *A est une matrice antisymétrique si $A^T = -A$.*
5. *A est une matrice orthogonale si $A^T A = I_n$.*

Remark 5.

1. *Pour toute matrice carrée $A \in \mathcal{M}_n(\mathbb{R})$, $I_n A = A I_n = A$.*
2. *L'ensemble des matrices symétriques réelles est noté $\mathcal{S}_n(\mathbb{R})$.*
3. *L'ensemble des matrices orthogonales réelles est noté $\mathcal{O}_n(\mathbb{R})$.*

Example 7.

1. $U = \begin{pmatrix} 2 & 5 & -1 \\ 0 & 3 & 4 \\ 0 & 0 & 7 \end{pmatrix}$ *est une matrice triangulaire supérieure.*
2. $L = \begin{pmatrix} 4 & 0 & 0 \\ -2 & 1 & 0 \\ 3 & 6 & 5 \end{pmatrix}$ *est une matrice triangulaire inférieure.*

3. $D = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 6 \end{pmatrix}$ *est une matrice diagonale.*

4. $C == \begin{pmatrix} 1 & 2 & 0 \\ 3 & 4 & 5 \\ 0 & 6 & 7 \end{pmatrix}$ *est ni triangulaire et ni diagonale.*

5. $I_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ *est la matrice identité de* $\mathcal{M}_3(\mathbb{R})$.

2.2 Matrices et Applications linéaires

Proposition 6. *Soit* $f : \mathbb{R}^n \to \mathbb{R}^m$ *une application linéaire. Alors, il existe une matrice* $M_f = M_{\mathcal{B}_{\mathbb{R}^m}, \mathcal{B}_{\mathbb{R}^n}}(f) \in \mathcal{M}_{m,n}(\mathbb{R})$ *telle que* $f(X) = M_f X$ *où* $X \in \mathbb{R}^n$.

Remark 6.

1. *La matrice* M_f *est appelée matrice associée à* f.
2. f *est appelée application linéaire associée à* M_f.
3. $f^T(X) = X^T M_f^T$ *où* $X \in \mathbb{R}^n$

Proposition 7. *Soient* $f : \mathbb{R}^n \to \mathbb{R}^m$ *et* $g : \mathbb{R}^m \to \mathbb{R}^p$ *deux applications linéaires. Alors, la matrice associée à* $g \circ f$ *est* $M_g \times M_f \in \mathcal{M}_{pn}(\mathbb{R})$.

Déterminer la matrice d'une application linéaire avec python

la fonction ***jacobien*** *de* ***sympy*** *permet de trouver la matrice d'une application linéaire dans la base canonique.*

Example 8. *Déterminer la matrice de l'application linéaire* $f(x, y) = (2x + y, x - 3y)$ *dans la base canonique de* $\mathbb{R}^2$.

Matrice d'une application linéaire Python

[28]:
```
import sympy as sp

# Variables symboliques
x, y = sp.symbols('x y')

# Définition du vecteur image
def f(x,y) : return sp.Matrix([2*x + y, x - 3*y])

# Jacobien (matrice des dérivées partielles)
Mf = f(x,y).jacobian([x, y])
print(Mf)
```

```
Matrix([[2, 1], [1, -3]])
```

Example 9. *Déterminer l'application linéaire de* $A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 2 & 1 \\ -2 & -1 & -1 \end{pmatrix}$.

Soit $f(x,y,z) = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 2 & 1 \\ -2 & -1 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = (3x + y + 2z,\ x + 2y + z,\ -2x - y - z).$

Application linéaire d'une matrice Python

```
[43]: import sympy as sp
      #Déclation des variable et la matrice
      x, y, z =sp.symbols('x y z', real=True)
      T=sp.Matrix([x, y, z])
      A=sp.Matrix([[3, 1, 2],
                   [1, 2, 1],
                   [-2, -1, -1]])
      print('f(x,y,z)=')
      sp.pprint(A@T)
```

```
f(x,y,z)=
⎡3⋅x + y + 2⋅z⎤
⎢            ⎥
⎢ x + 2⋅y + z ⎥
⎢            ⎥
⎣-2⋅x - y - z ⎦
```

Figure 5: Application linéaire d'une matrice

Exercices (TD-TP)

Exercice 1 :

Considérons les matrices $A = \begin{pmatrix} 0 & -3 & 2 \\ 3 & 0 & -1 \\ -2 & 1 & 0 \end{pmatrix}$ *et* $B = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

1. *Vérifier si les matrices* A *et* B *sont symétriques, antisymétriques ou orthogonales.*
2. *Déterminer les applications linéaires associées à* A^T *et* B^T *ainsi que leurs rangs.*
3. *Déterminer les applications linéaires associées à* A^T *et* B^T *ainsi que leurs rangs avec python.*

Exercice 2 :

Soit l'application linéaire $f(x,y) = (2x + y, x - 3y)$ *définie de* $\mathbb{R}^2$ *muni de la base* $\{u = (1,-1), v = (2,1)\}$ *dans* $\mathbb{R}^2$ *muni de la base* $\{u' = (1,-1), v' = (-2,1)\}$.

1. *Déterminer la matrice* M_f *de* f *par rapport aux deux bases.*
2. *Déterminer la matrice transposée de la matrice* M_f *de* f *par rapport aux deux bases.*

3 Calcul du déterminant et de l'inverse d'une matrice

Pour le calcul de déterminant, nous considérons A une matrice carrée réelle.

Definition 9. *On appelle déterminant de A le réel défini et noté par :*

$$det(A) = |A| = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix}.$$

Definition 10. *On appelle mineur de l'élément a_{ij}, noté A_{ij} le déterminant obtenue après le retrait de la ligne i et de la colonne j de $det(A)$.*

Exemple

$$A_{11} = \begin{vmatrix} a_{22} & a_{23} & \cdots & a_{2n} \\ a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n2} & a_{n3} & \cdots & a_{nn} \end{vmatrix}.$$

Definition 11. *On appelle cofacteur de l'élément a_{ij} le déterminant défini et noté par $C_{ij} = (-1)^{i+j} A_{ij}$.*

<u>Méthode de Calcul du Déterminant :</u>

Déterminant D'une matrice Carré d'ordre 2

Posons $n = 2$, alors $det(A) = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{21}a_{12}$.

Déterminant D'une matrice Carré d'ordre $n > 2$

1. *Déterminant par rapport à la ligne i:* $det(A) = \sum_{j=1}^{n} a_{ij}C_{ij}$.
2. *Déterminant par rapport à la colonne j :* $det(A) = \sum_{i=1}^{n} a_{ij}C_{ij}$.

<u>Attention</u>

1. *Le déterminant d'une matrice est nul si :*
   (a) *deux colonnes ou lignes adjacentes sont liées.*
   (b) *une colonne ou ligne est nulle.*
2. *Soit $\lambda \in K$ un scalaire, alors :*

$$det(A) = \begin{vmatrix} \lambda a_{11} & \lambda a_{12} & \cdots & \lambda a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix} = \begin{vmatrix} \lambda a_{11} & a_{12} & \cdots & a_{1n} \\ \lambda a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix} = \lambda \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix}$$

3. *Si l'on échange entre elles deux colonnes ou lignes; le déterminant change de signe.*

Example 10. *Soit* $A = \begin{pmatrix} 3 & 0 & -1 \\ 2 & 2 & -2 \\ 1 & 0 & 1 \end{pmatrix}$. *Alors :*

$$det(A) = \begin{vmatrix} 3 & 0 & -1 \\ 2 & 2 & -2 \\ 1 & 0 & 1 \end{vmatrix} = 2 \begin{vmatrix} 3 & -1 \\ 1 & 1 \end{vmatrix} = 2(3 \times 1 - 1 \times (-1)) = 2(3+1) = 2 \times 4 = 8.$$

Calcul de déterminant avec python

```
[5]: import sympy as sp
     # Définition de la matrice A
     A = sp.Matrix([
         [3, 0, -1],
         [2, 2, -2],
         [1, 0, 1]
     ])
     # Calcul du déterminant
     det_A = A.det()
     print("La matrice A est :\n")
     sp.pprint(A)
     f"det(A)={det_A}"

     La matrice A est :

     ⎡3  0  -1⎤
     ⎢        ⎥
     ⎢2  2  -2⎥
     ⎢        ⎥
     ⎣1  0  1 ⎦
[5]: 'det(A)=8'
```

Figure 6: Déterminant d'une matrice

Proposition 8. *Soit* $A \in \mathcal{M}_n(\mathbb{R})$ *une matrice carrée réelle. Alors,* A *est inversible si et seulement* $det(A) \neq 0$. *L'inverse de* A *est notée* A^{-1}.

Remark 7. *Soit* A^{-1} *l'inverse d'une matrice* A, *alors* $A^{-1} \times A = A \times A^{-1} = I_n$.

Méthode :

Une des méthodes de calcul de l'inverse d'une matrice carrée consiste à faire des opérations élémentaires sur ligne [1] *pour obtenir la transformation du graphe suivant :*

$$(A \,|\, I_n) \xrightarrow[\text{opérations sur les lignes}]{} (I_n \,|\, A^{-1}).$$

1. *les opérations élémentaires sur les lignes :*

[1] Méthode du tableau augmenté

- ***Permutation de lignes :*** $L_i \leftrightarrow L_j$;
- ***Multiplication d'une ligne par un scalaire non nul :*** $L_i \leftarrow kL_i$, *avec* $k \neq 0$;
- ***Ajout d'un multiple d'une ligne à une autre :*** $L_i \leftarrow L_i + kL_j$.

Example 11. *Soit la matrice :* $A = \begin{pmatrix} 2 & 1 \\ 5 & 3 \end{pmatrix}$.

Nous cherchons à calculer son inverse A^{-1} *à l'aide de la méthode du tableau augmenté :*

mettre le pivot à 1
$\Downarrow$

$$(A \mid I_2) = \left(\begin{array}{cc|cc} \boxed{2} & 1 & 1 & 0 \\ 5 & 3 & 0 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{cc|cc} 1 & \frac{1}{2} & \frac{1}{2} & 0 \\ 5 & 3 & 0 & 1 \end{array}\right) L_1 \leftarrow \tfrac{1}{2}L_1$$

$$\Rightarrow \left(\begin{array}{cc|cc} 1 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \boxed{\frac{1}{2}} & -\frac{5}{2} & 1 \end{array}\right) L_2 \leftarrow L_2 - 5L_1$$

$$\Rightarrow \left(\begin{array}{cc|cc} 1 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 1 & -5 & 2 \end{array}\right) L_2 \leftarrow 2L_2$$

$$\Rightarrow \left(\begin{array}{cc|cc} 1 & 0 & 3 & -1 \\ 0 & 1 & -5 & 2 \end{array}\right) L_1 \leftarrow L_1 - \tfrac{1}{2}L_2.$$

Alors, $A^{-1} = \begin{pmatrix} 3 & -1 \\ -5 & 2 \end{pmatrix}$.

Example 12. *Soit la matrice* $A = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 1 & -1 \\ 3 & 1 & 0 \end{pmatrix}$ *alors :*

$$(A \mid I_3) = \left(\begin{array}{ccc|ccc} 1 & 2 & 1 & 1 & 0 & 0 \\ 2 & 1 & -1 & 0 & 1 & 0 \\ 3 & 1 & 0 & 0 & 0 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|ccc} 1 & 2 & 1 & 1 & 0 & 0 \\ 0 & -3 & -3 & -2 & 1 & 0 \\ 0 & -5 & -3 & -3 & 0 & 1 \end{array}\right) \begin{array}{l} L_2 \leftarrow L_2 - 2L_1 \\ L_3 \leftarrow L_3 - 3L_1 \end{array}$$

$$\Rightarrow \left(\begin{array}{ccc|ccc} 1 & 2 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & \frac{2}{3} & -\frac{1}{3} & 0 \\ 0 & -5 & -3 & -3 & 0 & 1 \end{array}\right) L_2 \leftarrow -\tfrac{1}{3}L_2$$

$$\Rightarrow \left(\begin{array}{ccc|ccc} 1 & 0 & -1 & -\frac{1}{3} & \frac{2}{3} & 0 \\ 0 & 1 & 1 & \frac{2}{3} & -\frac{1}{3} & 0 \\ 0 & 0 & 2 & \frac{1}{3} & -\frac{5}{3} & 1 \end{array}\right) \begin{array}{l} L_1 \leftarrow L_1 - 2L_2 \\ \\ L_3 \leftarrow L_3 + 5L_2 \end{array}$$

$$\Rightarrow \left(\begin{array}{ccc|ccc} 1 & 0 & -1 & -\frac{1}{3} & \frac{2}{3} & 0 \\ 0 & 1 & 1 & \frac{2}{3} & -\frac{1}{3} & 0 \\ 0 & 0 & 1 & \frac{1}{6} & -\frac{5}{6} & \frac{1}{2} \end{array}\right) L_3 \leftarrow \tfrac{1}{2}L_3$$

$$\Rightarrow \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{1}{6} & -\frac{1}{6} & \frac{1}{2} \\ 0 & 1 & 0 & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ 0 & 0 & 1 & \frac{1}{6} & -\frac{5}{6} & \frac{1}{2} \end{array}\right) \begin{array}{l} L_1 \leftarrow L_1 + L_3 \\ L_2 \leftarrow L_2 - L_3. \\ \end{array}$$

Ainsi, on obtient : $A^{-1} = \begin{pmatrix} -\frac{1}{6} & -\frac{1}{6} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{6} & -\frac{5}{6} & \frac{1}{2} \end{pmatrix}$.

Inverse d'une matrice Python

[8]:
```python
import sympy as sp
# Définition de la matrice A
A = sp.Matrix([
    [1, 2, 1],
    [2, 1, -1],
    [3, 1, 0]
])
# Calcul de l'inverse
A_inv = A.inv()
# Affichage
print("Matrice A :\n")
sp.pprint(A)
print("\nInverse de A :\n")
sp.pprint(A_inv)
# Vérification (A * A_inv = I)
print("\nVérification A * A_inv =\n")
sp.pprint(A * A_inv)
```

```
Matrice A :

⎡1  2  1 ⎤
⎢        ⎥
⎢2  1  -1⎥
⎢        ⎥
⎣3  1  0 ⎦

Inverse de A :

⎡-1/6  -1/6  1/2 ⎤
⎢                ⎥
⎢1/2   1/2   -1/2⎥
⎢                ⎥
⎣1/6   -5/6  1/2 ⎦

Vérification A * A_inv =

⎡1  0  0⎤
⎢       ⎥
⎢0  1  0⎥
⎢       ⎥
⎣0  0  1⎦
```

Figure 7: Calcul de l'inverse d'une matrice

Exercices (TD-TP)

Exercice 1 :

Considérons les matrices suivantes :

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 0 & 0 & 0 \end{pmatrix}; B = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 7 & 8 & 9 \end{pmatrix}; C = \begin{pmatrix} 2 & 1 & 3 \\ 4 & -2 & 5 \\ 1 & 3 & -1 \end{pmatrix}; D = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 9 & 12 & 15 \end{pmatrix}; E = \begin{pmatrix} 2 & 3 & 1 \\ 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

1. *Calculer le déterminant de chacune des matrices ci-dessus.*
2. *En déduire celles qui sont inversible et calculer leurs inverses.*
3. *Calculer le déterminant de chacune des matrices ci-dessus et calculer les inverses de celles qui sont inversibles avec python.*

4 Décomposition en valeurs singulières

4.1 Produit scalaire euclidien

Definition 12. *Soient $x, y \in \mathbb{R}^n$. Le **produit scalaire euclidien** (ou usuel) de x et y est défini par :*

$$\langle x, y \rangle = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n.$$

Remark 8. *Soit $x \in \mathbb{R}^n$. Alors, $\langle x, x \rangle = \|x\|^2 = x_1^2 + x_2^2 + \cdots + x_n^2$.*

4.2 Norme et distance euclidiennes

Definition 13. *Soit $x \in \mathbb{R}^n$. La **norme euclidienne** (ou **usuelle**) est définie par le réel $\|x\| = \sqrt{x_1^2 + \cdots + x_n^2}$.*

Proposition 9. *La norme euclidienne satisfait aux propriétés suivantes :*

- $\|x\| \geq 0$ *pour tout $x \in \mathbb{R}^n$, et $\|x\| = 0$ si et seulement si $x = 0$.*
- $\|\lambda x\| = |\lambda| \|x\|$ *pour tous $x \in \mathbb{R}^n$, $\lambda \in \mathbb{R}$.*
- $\|x + y\| \leq \|x\| + \|y\|$ *pour tous $x, y \in \mathbb{R}^n$ (inégalité triangulaire).*

Definition 14. *On dit que $x \in \mathbb{R}^n$ est **unitaire** (ou **normalisé**) s'il est tel que $\|x\| = 1$.*

Definition 15. *Soit $v \in \mathbb{R}^n$ un vecteur non nul, $D = Vect(v)$ la droite vectorielle engendrée par v. On appelle la projection du vecteur $u \in \mathbb{R}^n$ sur la droite le vecteur $\mathbf{proj_D}(\mathbf{u}) = \frac{\langle \mathbf{u}, \mathbf{v} \rangle}{\|\mathbf{v}\|^2} \mathbf{v}$.*

Remark 9.
La projection orthogonale d'un vecteur $\mathbf{v}$ sur une droite vectorielle $D = \mathrm{Vect}(\mathbf{u})$ consiste à remplacer $\mathbf{v}$ par son ombre portée sur la droite D. Autrement dit, on cherche le vecteur de D qui est le plus proche possible de $\mathbf{v}$.

Géométriquement, cela signifie que le vecteur

$$\mathrm{proj}_D(\mathbf{v})$$

est l'unique vecteur de D tel que la différence

$$\mathbf{v} - \mathrm{proj}_D(\mathbf{v})$$

soit orthogonale à $\mathbf{u}$, c'est-à-dire perpendiculaire à la droite D.
Cette projection correspond donc au pied de la perpendiculaire abaissée depuis l'extrémité du vecteur $\mathbf{v}$ jusqu'à la droite D. Si tu veux, je peux aussi te donner

4.3 Orthonormalisation de Gram-Schmidt

Soit $\mathcal{B} = \{w_1, \cdots, w_n\}$ une base de $\mathbb{R}^n$. L'orthonormalisation de $\mathcal{B}$ consiste à normaliser et orthogonaliser les éléments de $\mathcal{B}$. L'une des méthodes connues est l'orthonormalisation de Gram-Schmidt décrite par :
premier point consiste à orthogonaliser :

$$\begin{aligned} v_1 &= w_1, \\ v_2 &= w_2 - \frac{\langle w_2, v_1\rangle}{\|v_1\|^2} v_1, \\ v_3 &= w_3 - \sum_{i=1}^{2} \frac{\langle w_3, v_i\rangle}{\|v_i\|^2} v_i, \\ &\vdots \\ v_n &= w_n - \sum_{i=1}^{n-1} \frac{\langle w_n, v_i\rangle}{\|v_i\|^2} v_i. \end{aligned}$$

La famille $\mathcal{B}' = \{v_1, v_2, \cdots, v_n\}$ est une base orthogonale de $\mathbb{R}^n$.
Le deuxième point consiste à normaliser chaque vecteur de $\mathcal{B}'$ par : $\epsilon_i = \frac{v_i}{\|v_i\|}$.
La $\{\epsilon_i\}$ forme une base orthonormale de $\mathbb{R}^n$.

Example 13. *Orthonormalisons la base $\{u_1 = (1, -1), u_2 = (2, 1)\}$.*

1. *Orthogonalisation :*
   Posons $v_1 = u_1$.
   Calculons $\langle u_1, u_2\rangle = 1 \times 2 + (-1) \times 1 = 2 - 1 = 1$.
   Calculons $\|v_1\|^2 = 1^2 + (-1)^2 = 1 + 1 = 2$.
   Alors :$v_2 = u_2 - \frac{\langle u_1, u_2\rangle}{\|v_1\|^2} v_1 = (2, 1) - \frac{1}{2}(1, -1) = (\frac{3}{2}, \frac{3}{2})$
2. *Normalisation :*
   $\Rightarrow v_1' = (\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$,
   $\Rightarrow \|v_2\| = \sqrt{\frac{9}{4} + \frac{9}{4}} = \sqrt{\frac{9}{2}} = \frac{3}{\sqrt{2}}$, *alors* $v_2' = \frac{1}{\frac{3\sqrt{2}}{2}}(\frac{3}{2}, \frac{3}{2}) = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$.

Alors la famille $\{v_1', v_2'\}$ forme une base orthonormée de $\mathbb{R}^2$.

Orthonormalisation avec Python

```
[6]: import sympy as sp
     #GramSchmidt
     # les vecteurs
     v=sp.Matrix([1,-1])
     u=sp.Matrix([2,1])
     bas_orthonorm=sp.GramSchmidt([v,u],orthonormal=True)
     sp.pprint(bas_orthonorm)
```

```
⎡⎡ √2 ⎤  ⎡√2⎤⎤
⎢⎢ ── ⎥  ⎢──⎥⎥
⎢⎢ 2  ⎥  ⎢2 ⎥⎥
⎢⎢    ⎥, ⎢  ⎥⎥
⎢⎢-√2 ⎥  ⎢√2⎥⎥
⎢⎢────⎥  ⎢──⎥⎥
⎣⎣ 2  ⎦  ⎣2 ⎦⎦
```

Figure 8: Orthonormalisation

Exercices (TD-TP)

Exercice 1 :

Considérons les familles $B_1 = \left\{ \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right\}$ *et* $B_2 = \left\{ \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \right\}$.

1. *Orthonormaliser* B_1 *et* B_2.
2. *Orthonormaliser* B_1 *et* B_2 *avec python.*

4.4 Valeurs et vecteurs propres

Definition 16. *Soit* $A \in \mathcal{M}_n(\mathbb{R})$ *la matrice d'un endomorphisme* f *de* $\mathbb{R}^n$. *Pour tout* $\lambda \in \mathbb{R}$, *s'il existe* $v \in \mathbb{R}^n$ *non nul tel que* $\mathbf{Av} = \lambda \mathbf{v}$, *alors :*

1. λ *est appelé* ***valeur propre*** *associée au vecteur* v.
2. v *est appelé* ***vecteur propre*** *associé à* λ.

Definition 17. *Soit* λ *une valeur propre d'une matrice* $A \in \mathbb{M}_n(\mathbb{R})$. *On appelle sous espace propre de* λ *l'ensemble défini et noté par* $E_\lambda = \{v \in \mathbb{R}^n | (A - \lambda I_n)v = 0\}$.

Proposition 10.

$$\begin{aligned} \lambda \text{ est une valeur propre de } A &\Leftrightarrow \ker(f - \lambda \operatorname{Id}_{\mathbb{R}^n}) \neq \{0\} \\ &\Leftrightarrow E_\lambda \neq \{0\} \\ &\Leftrightarrow f \text{ n'est pas injectif} \\ &\Leftrightarrow f \text{ n'est pas bijectif} \\ &\Leftrightarrow \det(f - \lambda \operatorname{Id}_{\mathbb{R}^n}) = 0 \\ &\Leftrightarrow \det(A - \lambda I_n) = 0 \end{aligned}$$

Example 14. *Soit* $A = \begin{pmatrix} 2 & 1 & 1 \\ -1 & 0 & -1 \\ 1 & 1 & 2 \end{pmatrix}$ *une matrice.*
Calculer les valeurs et vecteurs propres de A.

1. *Calcul des valeurs propres*

   Les valeurs propres sont les racines du polynôme caractéristique $\det(A - \lambda I) = 0$.

$$A - \lambda I = \begin{pmatrix} 2-\lambda & 1 & 1 \\ -1 & -\lambda & -1 \\ 1 & 1 & 2-\lambda \end{pmatrix}$$

Calculons le déterminant :

$$\begin{aligned}
\det(A-\lambda I) &= (2-\lambda)\begin{vmatrix}-\lambda & -1\\ 1 & 2-\lambda\end{vmatrix} - 1\cdot\begin{vmatrix}-1 & -1\\ 1 & 2-\lambda\end{vmatrix} + 1\cdot\begin{vmatrix}-1 & -\lambda\\ 1 & 1\end{vmatrix}\\
&= (2-\lambda)[(-\lambda)(2-\lambda)-(-1)(1)] - 1[(-1)(2-\lambda)-(-1)(1)] + 1[(-1)(1)-(-\lambda)(1)]\\
&= (2-\lambda)[-\lambda(2-\lambda)+1] - 1[-(2-\lambda)+1] + 1[-1+\lambda]\\
&= (2-\lambda)[-2\lambda+\lambda^2+1] - 1[-2+\lambda+1] + (\lambda-1)\\
&= (2-\lambda)(\lambda^2-2\lambda+1) - 1(\lambda-1) + (\lambda-1)\\
&= (2-\lambda)(\lambda-1)^2 - (\lambda-1) + (\lambda-1)\\
&= (2-\lambda)(\lambda-1)^2
\end{aligned}$$

Le polynôme caractéristique est donc $(2-\lambda)(\lambda-1)^2=0$.

Valeurs propres : $\lambda_1=2$ *(valeur propre simple),* $\lambda_2=1$ *(valeur propre double).*

2. *Calcul des vecteurs propres*
   Pour $\lambda=2$

   On résout $(A-2I)X=0$ *:* $A-2I=\begin{pmatrix}0 & 1 & 1\\ -1 & -2 & -1\\ 1 & 1 & 0\end{pmatrix}$

   Système d'équations : $(A-2I)X=\begin{pmatrix}0 & 1 & 1\\ -1 & -2 & -1\\ 1 & 1 & 0\end{pmatrix}\begin{pmatrix}x\\ y\\ z\end{pmatrix}\Rightarrow\begin{cases}x_2+x_3=0,\\ -x_1-2x_2-x_3=0,\\ x_1+x_2=0.\end{cases}$

   De la première équation : $x_3=-x_2$.
   De la troisième équation : $x_1=-x_2$.
   Vérifions la deuxième : $-(-x_2)-2x_2-(-x_2)=x_2-2x_2+x_2=0$.

   Donc $X=\begin{pmatrix}-x_2\\ x_2\\ -x_2\end{pmatrix}=x_2\begin{pmatrix}-1\\ 1\\ -1\end{pmatrix}$.

   Vecteur propre pour $\lambda=2$ ***:*** $v_1=\begin{pmatrix}1\\ -1\\ 1\end{pmatrix}$.

   Pour $\lambda=1$ *(multiplicité 2)*

   On résout $(A-I)X=0$ *:* $A-I=\begin{pmatrix}1 & 1 & 1\\ -1 & -1 & -1\\ 1 & 1 & 1\end{pmatrix}\begin{pmatrix}x_1\\ x_2\\ x_3\end{pmatrix}=\begin{pmatrix}0\\ 0\\ 0\end{pmatrix}\Rightarrow x_1+x_2+x_3=0$. *Alors,*

   $x_1=-x_2-x_3\Rightarrow X=\begin{pmatrix}-x_2-x_3\\ x_2\\ x_3\end{pmatrix}\Rightarrow v_2=\begin{pmatrix}-1\\ 1\\ 0\end{pmatrix},\qquad v_3=\begin{pmatrix}-1\\ 0\\ 1\end{pmatrix}$.

Valeurs et vecteurs propres avec Python

[12]:
```python
import sympy as sp
# Définir la matrice
A = sp.Matrix([    [2, 1, 1],[-1, 0, -1],[1, 1, 2]])
# Affichage de la matrice
# --- Valeurs propres ---
eigenvals = A.eigenvals()
# --- Vecteurs propres ---
eigenvects = A.eigenvects()
```

[13]:
```python
print("\nValeurs propres :")
for val, mult in eigenvals.items():
    print(f"λ = {val}, multiplicité = {mult}")

print("\nVecteurs propres :")
for val, mult, vects in eigenvects:
    print(f"\nPour λ = {val} (multiplicité {mult}) :")
    for v in vects:
        sp.pprint(v)
```

```
Valeurs propres :
λ = 2, multiplicité = 1
λ = 1, multiplicité = 2

Vecteurs propres :

Pour λ = 1 (multiplicité 2) :
Matrix([[-1], [1], [0]])
Matrix([[-1], [0], [1]])

Pour λ = 2 (multiplicité 1) :
Matrix([[1], [-1], [1]])
```

Figure 9: Valeurs et vecteurs propres

Exercices (TD-TP)

Exercice 1 :

Considérons les matrices suivantes $A = \begin{pmatrix} 0 & 2 & 1 \\ 1 & 1 & 0 \\ -1 & -2 & -1 \end{pmatrix}$ *et* $B = \begin{pmatrix} 3 & 0 & -1 \\ 2 & 2 & -2 \\ 1 & 0 & 1 \end{pmatrix}$.

1. *Déterminer les valeurs et vecteurs propres de* A *et de* B.
2. *Déterminer les valeurs et vecteurs propres de* A *et de* B *avec python.*

4.5 Décomposition en valeurs singulières

Definition 18. *Soit $A \in \mathcal{M}_{mn}(\mathbb{R})$ une matrice d'une application linéaire $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$. On appelle décomposition en valeurs singulières [2] de A la décomposition définie par :*

$$A = U\Sigma V^T$$

où

1. *$U \in \mathcal{M}_m(\mathbb{R})$ orthogonale : $UU^T = U^TU = I_m$;*
2. *$\Sigma \in \mathcal{M}_{mn}(\mathbb{R})$ diagonale : $\Sigma_{ij} = 0$, si $i \neq j$ et de plus $\Sigma_{ii} \geq 0$;*
3. *$V \in \mathcal{M}_n(\mathbb{R})$ orthogonale : $VV^T = V^TV = I_n$.*

Proposition 11. *Pour $A \in \mathcal{M}_{mn}(\mathbb{R})$ une matrice. Alors :*

1. *les matrices AA^T et A^TA sont symétriques.*
2. *λ est une valeur propre de $AA^T \Leftrightarrow \lambda$ est une valeur propre de A^TA.*
3. *pour toute valeur propre λ de AA^T, alors $\lambda \geq 0$.*
4. *les matrices AA^T et A^TA admettent au plus $min\{n, m\}$ valeurs propres non nulles en commun.*

Méthode de la SVD

SVD1 : *Calculer les valeurs propres $\lambda_1 \geq \cdots \geq \lambda_n \geq 0$ de $A^TA \in \mathbb{M}_{n\times n}(\mathbb{R})$ (chacune est répétée autant de fois que sa multiplicité algébrique [3]), en distinguant ses valeurs propres positives $\lambda_1 \geq \cdots \geq \lambda_\ell > 0$, où $\ell = rang(A) = rang(A^TA)$.*
Les valeurs singulières de A sont $\sigma_j := \sqrt{\lambda_j}, \quad j = 1, 2, \ldots, \ell$.

La matrice $\Sigma \in \mathbb{M}_{m\times n}(\mathbb{R})$ s'obtient alors $\Sigma := \begin{bmatrix} \sigma_1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & 0 \\ 0 & \cdots & \sigma_\ell & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \cdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix}$.

SVD2 : *Une base orthonormée $\{v_1, \ldots, v_n\}$ de vecteurs propres de A^TA, où v_i est le vecteur propre de valeur propre λ_i, donnent la matrice $V := [v_1 \cdots v_n] \in \mathbb{M}_{n\times n}(\mathbb{R})$, les $n - \ell$ dernières colonnes étant associées à la valeur propre nulle (si besoin est).*

SVD3 : *On définit d'abord $u_j := \frac{1}{\sqrt{\lambda_j}} Av_j, \quad j = 1, 2, \ldots, \ell$. Si $\ell = m$, on pose $U := [u_1 \cdots u_m] \in \mathbb{M}_{m\times m}(\mathbb{R})$.*

Si $\ell < m$, on utilise une entre les deux méthodes ci-dessous :

- ***(SVD.3.i)** on complète $\{u_1, \ldots, u_\ell\}$ en une base $\{u_1, \ldots, u_\ell, w_{\ell+1}, \ldots, w_m\}$ de $\mathbb{R}^m$, et on applique le procédé d'orthonormalisation de Gram-Schmidt pour obtenir une base orthonormée $\{u_1, \ldots, u_\ell, u_{\ell+1}, \ldots, u_m\}$ de $\mathbb{R}^m$.*
- ***(SVD.3.ii)** on calcule une base $\{w_{\ell+1}, \ldots, w_m\}$ du noyau de la matrice $A^T \in \mathbb{M}_{n\times m}(\mathbb{R})$ (ou $AA^T \in \mathbb{M}_{m\times m}(\mathbb{R})$), et on applique le procédé d'orthonormalisation de Gram-Schmidt pour obtenir une base orthonormée $\{u_{\ell+1}, \ldots, u_m\}$ de $\mathrm{Ker}(A^T) = \mathrm{Ker}(AA^T) \subseteq \mathbb{R}^m$.*

[2] SVD : Singular Value Decomposition

[3] La multiplicité algébrique d'une valeur propre λ est la dimension de son sous-espace propre E_λ.

On pose finalement

$$U := [u_1 \cdots, u_\ell \;\; u_{\ell+1} \cdots u_m] \in \mathbb{M}_{m\times m}(\mathbb{R}).$$

Remark 10.

1. *Si le rang de A est ℓ, $Rang(A) = \ell$, alors les $m-\ell$ dernières lignes et $n-\ell$ dernières colonnes de Σ sont nulles.*
2.

Example 15. *Calculons la décomposition en valeurs singulières de*

$$A = \begin{pmatrix} \frac{3}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} \\ \frac{9}{10\sqrt{2}} & \frac{13}{10\sqrt{2}} \end{pmatrix}$$

1.Calcul de A^TA :

$$A^TA = \begin{pmatrix} \frac{3}{2\sqrt{2}} & \frac{9}{10\sqrt{2}} \\ \frac{1}{2\sqrt{2}} & \frac{13}{10\sqrt{2}} \end{pmatrix} \begin{pmatrix} \frac{3}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} \\ \frac{9}{10\sqrt{2}} & \frac{13}{10\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{153}{100} & \frac{24}{25} \\ \frac{24}{25} & \frac{97}{100} \end{pmatrix}.$$

2. Valeurs propres de A^TA :

On résout $\det(A^TA - \lambda I) = 0$ *:*

$$\begin{vmatrix} \frac{153}{100} - \lambda & \frac{24}{25} \\ \frac{24}{25} & \frac{97}{100} - \lambda \end{vmatrix} = \lambda^2 - \left(\tfrac{153}{100} + \tfrac{97}{100}\right)\lambda + \tfrac{153}{100} \times \tfrac{97}{100} - \left(\tfrac{24}{25}\right)^2 = 0$$

$$= \lambda^2 - \tfrac{250}{100}\lambda + \tfrac{153\times 97}{100\times 100} - \tfrac{576}{625} = 0$$

$$= \lambda^2 - \tfrac{5}{2}\lambda + \tfrac{9}{16} = 0, \textit{ après simplification.}$$

$\Rightarrow \lambda_1 = \frac{9}{4}, \quad \lambda_2 = \frac{1}{4}$.

Les valeurs singulières sont : $\sigma_1 = \sqrt{\lambda_1} = \frac{3}{2}, \quad \sigma_2 = \sqrt{\lambda_2} = \frac{1}{2}$.

3. Vecteurs propres de A^TA

Pour $\lambda_1 = \frac{9}{4}$ *:*

$$A^TA - \tfrac{9}{4}I_2 = \begin{pmatrix} \frac{153}{100} & \frac{24}{25} \\ \frac{24}{25} & \frac{97}{100} \end{pmatrix} - \tfrac{9}{4}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{153}{100} - \frac{9}{4} & \frac{24}{25} \\ \frac{24}{25} & \frac{97}{100} - \frac{9}{4} \end{pmatrix} = \begin{pmatrix} \frac{153-9\times 25}{100} & \frac{24}{25} \\ \frac{24}{25} & \frac{97-9\times 25}{100} \end{pmatrix} = \begin{pmatrix} -\frac{18}{25} & \frac{24}{25} \\ \frac{24}{25} & -\frac{32}{25} \end{pmatrix}.$$

Alors, $(A^TA - \frac{9}{4}I_2)u = \begin{pmatrix} -\frac{18}{25} & \frac{24}{25} \\ \frac{24}{25} & -\frac{32}{25} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \Rightarrow \begin{cases} -18x + 24y = 0, \\ 24x - 32y = 0. \end{cases} \Rightarrow -3x + 4y = 0 \Rightarrow x = \frac{4}{3}y.$

Donc, $v = (x, y) = (\frac{4}{3}y, y) = \frac{y}{3}(4, 3)$*, ainsi* $E_{\frac{9}{4}} = vect(4, 3)$.

On fait de même pour $\lambda_2 = \frac{1}{4}$*,* $E_{\frac{1}{4}} = vect(-3, 4)$.

Ainsi, après normalisation $V = \begin{pmatrix} \frac{4}{5} & -\frac{3}{5} \\ \frac{3}{5} & \frac{4}{5} \end{pmatrix}$.

4. Calcul de U par $u_i = \frac{1}{\sigma_i}Av_i$

$$u_1 = \frac{1}{\sigma_1}Av_1 = \frac{2}{3}A\begin{pmatrix} \frac{4}{5} \\ \frac{3}{5} \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad u_2 = \frac{1}{\sigma_2}Av_2 = 2A\begin{pmatrix} -\frac{3}{5} \\ \frac{4}{5} \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}.$$

$$\text{Donc } U = \begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}.$$

5. Matrice des valeurs singulières

$$\Sigma = \begin{pmatrix} \frac{3}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}.$$

6. Résultat de la SVD

$$\boxed{A = U\,\Sigma\,V^T = \begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} \begin{pmatrix} \frac{3}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{4}{5} & \frac{3}{5} \\ -\frac{3}{5} & \frac{4}{5} \end{pmatrix}.}$$

SVD d'une matrice Python

[17]:
```python
import sympy as sp
# Définition exacte de A (avec les bons préfixes)
A = sp.Matrix([
    [sp.Rational(3, 2)/sp.sqrt(2), sp.Rational(1, 2)/sp.sqrt(2)],
    [sp.Rational(9, 10)/sp.sqrt(2), sp.Rational(13, 10)/sp.sqrt(2)]
])

# Décomposition en valeurs singulières (SVD)
U, S, V = A.singular_value_decomposition()
```

[18]:
```python
# Affichage des résultats
print("U =")
sp.pprint(U)

print("\nS =")
sp.pprint(S)

print("\nV =")
sp.pprint(V)
```

```
U =
⎡-√2   √2⎤
⎢────  ──⎥
⎢ 2    2 ⎥
⎢        ⎥
⎢ √2   √2⎥
⎢ ──   ──⎥
⎣ 2    2 ⎦

S =
⎡1/2   0 ⎤
⎢        ⎥
⎣ 0   3/2⎦

V =
⎡-3/5  4/5⎤
⎢         ⎥
⎣4/5   3/5⎦
```

Figure 10: SVD d'une matrice

Remark 11. *Lors qu'une valeur propre d'une matrice A est nulle, alors :*

1. *le rang de A (resp. f l'application linéaire de A) est $Rang(A) = Rang(f) = n - 1$, où n l'ordre de A,*

2. *On complète la base orthonormée pour construire U en utilisant le théorème de la base incomplète (voir exercice 1) en utilisant le produit vectoriel ou calculer les vecteurs propres associés à la valeur propre nulle par rapport à AA^T (ou A^TA).*

Exercices (TD-TP)

Exercice 1 :

Considérons la matrice $A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \end{pmatrix}$.

1. *Calculer A^TA et AA^T.*

2. *Calculer les valeurs propres de A^TA.*

3. *Calculer le vecteur propre associé à chaque valeur propre de A^TA.*

4. *Déterminer la décomposition en valeurs singulières de A.*

5. *Procéder à la décomposition en valeurs singulières de A avec python.*

Exercice 2 :

Considérons les matrices $A = \begin{pmatrix} 3 & 1 & 1 \\ -1 & 3 & 1 \end{pmatrix}$, $B = \begin{pmatrix} 2 & 0 & 1 \\ -1 & 1 & 2 \end{pmatrix}$.

1. *Déterminer la décomposition en valeurs singulières de A et B .*

2. *Procéder à la décomposition en valeurs singulières de A et B avec python.*

5 Décomposition LU et résolution de système d'équations

5.1 Décomposition LU

Definition 19. *Soit $A \in \mathcal{M}_{mn}(\mathbb{R})$ une matrice. On appelle décomposition LU de A la décomposition définie par $A = LU$, où :*

1. *$L \in \mathcal{M}_m(\mathbb{R})$ triangulaire inférieure : $l_{ij} = 0$, si $i < j$,*

2. *$U \in \mathcal{M}_{mn}(\mathbb{R})$ triangulaire supérieure : $u_{ij} = 0$, si $i > j$.*

Méthode de la décomposition LU

Une des méthodes de décomposer $A = LU$ d'effectuer la transformation suivante :

$$(I_m \,|\, A) \xrightarrow[\text{Algorithme LU sur les lignes}]{} (L \,|\, U).$$

Algorithme LU sur les lignes :

1. *Sur A, du graphe, on utilise les opérations élémentaires par élimination de Gauss afin d'avoir une matrice triangulaire supérieure.*
   (a) *Noter le pivot.*
   (b) *Noter l'indice du coefficient ij à annuler avec le pivot.*
   (c) *Trouver le facteur q_{ij} du pivot pour mettre nul le coefficient ij de A avec les opérations éléments.*
2. *Et mettre q_{ij} à la position ij dans I_m.*

Remark 12.

- *Dans le cas d'une matrice $m \times n$ avec $m > n$, la dernière ligne de U peut être nulle ou partiellement remplie, mais ce qui importe est que les éléments sous les pivots $U_{11}, U_{22}, \ldots, U_{nn}$ soient nuls.*
- *L'algorithme de décomposition LU ne cherche donc des pivots que jusqu'à l'indice $k = \min(m, n)$, et l'élimination s'arrête lorsque toutes les colonnes ou toutes les lignes pertinentes ont été traitées.*
- *Dans tous les cas, si certains pivots sont nuls, il est parfois nécessaire d'introduire une **permutation de lignes** (pivot de Gauss), c'est-à-dire écrire :*
$$PA = LU,$$
où P est une matrice de permutation.

Example 16. *Procéder à la décomposition LU de la matrice A définie par* $A = \begin{pmatrix} 2 & 1 \\ 4 & 3 \\ 6 & 5 \end{pmatrix}$.

On a :

$$\left(\begin{array}{ccc|cc} 1 & 0 & 0 & 2 & 1 \\ 0 & 1 & 0 & 4 & 3 \\ 0 & 0 & 1 & 6 & 5 \end{array}\right) \begin{array}{l} q_{21} = 2, q_{31} = 3 \\ L_2 \leftarrow L_2 - 2L_1 \\ L_3 \leftarrow L_3 - 3L_1 \end{array} \Rightarrow \left(\begin{array}{ccc|cc} 1 & 0 & 0 & 2 & 1 \\ \boxed{2} & 1 & 0 & 0 & 1 \\ \boxed{3} & 0 & 1 & 0 & 2 \end{array}\right) \begin{array}{l} q_{32} = 2 \\ L_3 \leftarrow L_3 - 2L_2 \end{array} \Rightarrow \left(\begin{array}{ccc|cc} 1 & 0 & 0 & 2 & 1 \\ 2 & 1 & 0 & 0 & 1 \\ 3 & \boxed{2} & 1 & 0 & 0 \end{array}\right).$$

Alors $L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix}$, $U = \begin{pmatrix} 2 & 1 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$ *et* $A = LU$.

LU d'une matrice Python

[4]:
```python
import sympy as sp

A = sp.Matrix([[2, 1],
               [4, 3],
               [6, 5]])

L, U, _ = A.LUdecomposition()
print("L ="); sp.pprint(L)
print("U ="); sp.pprint(U)
```

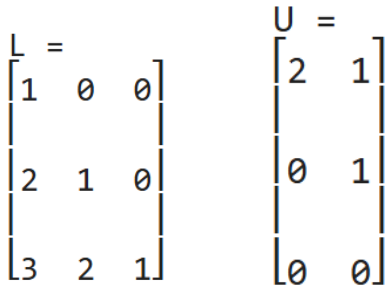


Figure 11: Décomposition LU d'une matrice

5.2 Résolution des systèmes linéaires avec la décomposition LU

En entreprise, on résout souvent le système d'équation $Ax = b$. Imaginer que $A \in \mathcal{M}_{12003\times 53008}(\mathbb{R})$, on aura à effectuer $12003 \times 53008 = 636\,255\,024$ opérations sur b. La décomposition LU permet d'éviter une bonne partie de ces opérations en subdivisant le processus en deux étapes :

1. *résoudre $Ly = b$: on résout ce système par substitution avant,*
2. *résoudre $Ux = y$: on résout ce système par substitution arrière.*

Example 17. *Résoudre le système* $\begin{cases} 2x_1 + 3x_2 + x_3 = 1 \\ 4x_1 + 7x_2 + 3x_3 = 0 \\ 4x_1 + 7x_2 + 4x_3 = -1 \end{cases}$.

On a la matrice associée au système : $A = \begin{pmatrix} 2 & 3 & 1 \\ 4 & 7 & 3 \\ 4 & 7 & 4 \end{pmatrix}$.

Résolution

On considère $A = \begin{pmatrix} 2 & 3 & 1 \\ 4 & 7 & 3 \\ 4 & 7 & 4 \end{pmatrix}$, $\quad b = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$.

Calcul de la décomposition LU :

$$\left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 2 & 3 & 1 \\ 0 & 1 & 0 & 4 & 7 & 3 \\ 0 & 0 & 1 & 4 & 7 & 4 \end{array}\right) \begin{array}{l} q_{21} = 2, q_{31} = 2 \\ L_2 \leftarrow L_2 - 2L_1 \\ L_3 \leftarrow L_3 - 2L_1 \end{array} \Rightarrow \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 2 & 3 & 1 \\ 2 & 1 & 0 & 0 & 1 & 1 \\ 2 & 0 & 1 & 0 & 1 & 2 \end{array}\right) \begin{array}{l} q_{32} = 1 \\ \\ L_3 \leftarrow L_3 - L_2 \end{array} \Rightarrow \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 2 & 3 & 1 \\ 2 & 1 & 0 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 0 & 1 \end{array}\right).$$

Résolution de $Ly = b$ ***(substitution avant) :***

Écrivons les équations composantes :

$$\begin{cases} y_1 = 1, \\ 2y_1 + y_2 = 0, \\ 2y_1 + y_2 + y_3 = -1. \end{cases}$$

Alors $y_1 = 1, \qquad y_2 = -2, \qquad y_3 = -1$. *Donc* $y = \begin{pmatrix} 1 \\ -2 \\ -1 \end{pmatrix}$.

Résolution de $Ux = y$ ***(substitution arrière) :***

Le système $Ux = y$ *s'écrit :*

$$\begin{cases} 2x_1 + 3x_2 + x_3 = 1, \\ x_2 + x_3 = -2, \\ x_3 = -1. \end{cases}$$

En remontant, on a : $x_3 = -1, \qquad x_2 + (-1) = -2 \Rightarrow x_2 = -1$, *et* $2x_1 + 3(-1) + (-1) = 1 \Rightarrow 2x_1 - 4 = 1 \Rightarrow x_1 = \frac{5}{2}$. *La solution est donc*

$$\boxed{x = \begin{pmatrix} \frac{5}{2} \\ -1 \\ -1 \end{pmatrix}}.$$

Vérification :*(non obligatoire)*

Calculons Ax *pour vérifier :* $Ax = \begin{pmatrix} 2 & 3 & 1 \\ 4 & 7 & 3 \\ 4 & 7 & 4 \end{pmatrix} \begin{pmatrix} \frac{5}{2} \\ -1 \\ -1 \end{pmatrix} = \begin{pmatrix} 2 \cdot \frac{5}{2} + 3 \cdot (-1) + 1 \cdot (-1) \\ 4 \cdot \frac{5}{2} + 7 \cdot (-1) + 3 \cdot (-1) \\ 4 \cdot \frac{5}{2} + 7 \cdot (-1) + 4 \cdot (-1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} = b.$

Ainsi la solution du système $Ax = b$ *obtenue par la décomposition* LU *est* $x = \begin{pmatrix} \frac{5}{2} \\ -1 \\ -1 \end{pmatrix}$.

Résolution de systèmes d'équations par LU Python

[6]:
```python
import sympy as sp
# Définition des symboles
x1, x2, x3 = sp.symbols('x1 x2 x3')

# Matrice A et vecteur b
A = sp.Matrix([[2, 3, 1], [4, 7, 3], [4, 7, 4]])
b = sp.Matrix([1, 0, -1])

# Décomposition LU
L, U, perm = A.LUdecomposition()
```

[12]:
```python
# affichage de L et U
print("\nDécomposition LU :")
print("L =")
sp.pprint(L)
print("\nU =")
sp.pprint(U)
# Étape 1 : Résolution de Ly = b (substitution avant)
y = L.solve(b)
print("\nSolution de Ly = b : y =")
sp.pprint(y)
# Étape 2 : Résolution de Ux = y (substitution arrière)
x = U.solve(y)
print("\nSolution finale de Ux = y : x =")
sp.pprint(x)
```

```
Décomposition LU :
L =
⎡1  0  0⎤
⎢       ⎥
⎢2  1  0⎥
⎢       ⎥
⎣2  1  1⎦

U =
⎡2  3  1⎤
⎢       ⎥
⎢0  1  1⎥
⎢       ⎥
⎣0  0  1⎦

Solution de Ly = b : y =
⎡1 ⎤
⎢  ⎥
⎢-2⎥
⎢  ⎥
⎣-1⎦

Solution finale de Ux = y : x =
⎡5/2⎤
⎢   ⎥
⎢-1 ⎥
⎢   ⎥
⎣-1 ⎦
```

Figure 12: Système d'équations et LU

Exercices (TD-TP)

Exercice 1 :

Considérons les matrices $A = \begin{pmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{pmatrix}$, $B = \begin{pmatrix} 1 & 2 & 1 \\ 3 & 8 & 1 \\ 0 & 4 & 1 \end{pmatrix}$, $C = \begin{pmatrix} 2 & 1 & 1 \\ 4 & 3 & 2 \\ 6 & 5 & 4 \\ 8 & 7 & 5 \end{pmatrix}$.

1. *Calculer les décompositions LU de A, B et C.*
2. *Calculer les décompositions LU de A, B et C avec python.*

Exercice 2:

On considère les matrices $A = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 2 & 1 & 1 \end{pmatrix}$ *et* $b = \begin{pmatrix} 3 \\ 0 \\ 1 \\ 4 \end{pmatrix}$; $A' = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 3 & 2 \\ 1 & 0 & 4 \end{pmatrix}$ *et* $b' = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$.

1. *Calculer les décompositions LU de* A *et* A'.
2. *Résoudre les systèmes d'équations* $Ax = b$ *et* $A'x = b'$.
3. *Résoudre les systèmes d'équations* $Ax = b$ *et* $A'x = b'$ *par la décomposition LU avec python.*

6 Décomposition QR et résolution de système d'équations

6.1 Décomposition QR

Definition 20. *Soit* $A \in \mathcal{M}_{mn}(\mathbb{R})$ *une matrice telle que* $m \geqslant n$. *On appelle décomposition QR de* A *la décomposition définie par* $A = QR$, *où :*

1. $Q \in \mathcal{M}_{mn}(\mathbb{R})$ *orthogonale :* $Q^TQ = I$,
2. $R \in \mathcal{M}_n(\mathbb{R})$ *triangulaire supérieure :* $r_{ij} = 0$, *si* $i > j$.

Proposition 12. *Soit* $A \in \mathcal{M}_{mn}(\mathbb{R})$ *une matrice. Il existe* Q *et* R *telles que* $A = QR$.

Remark 13. *Si* $m < n$, *on fait la décomposition de* $A^T = QR$ *en appliquant la définition 20. Et puis* $A = \tilde{Q}\tilde{R}$ *où* $\tilde{R} = R^T$ *et* $\tilde{Q} = Q^T$ *(ou* Q*).*

Méthode de la décomposition QR

1. *Considérer la famille* $\{a_1, a_2, \cdots, a_n\}$ *composée par les colonnes de* $A \in \mathcal{M}_{mn}(\mathbb{R})$.
2. *Détermination de* Q *:*
   (a) *Construire* $\{u_1, u_2, \cdots, u_n\}$ *orthogonalisée par la méthode de Gram-Schmidt 4.3 de* $\{a_1, a_2, \cdots, a_n\}$.
   (b) *Construire* $\{e_1, e_2, \cdots, e_n\}$ *normalisée :* $e_i = \frac{u_i}{\|u_i\|}$.

   Alors, $Q = [e_1, e_2, \cdots, e_n]$ *composée en colonne par les* e_i.
3. *Détermination de* R *:*
   (a) $R = Q^TA = \begin{pmatrix} \langle \mathbf{e}_1, \mathbf{a}_1 \rangle & \langle \mathbf{e}_1, \mathbf{a}_2 \rangle & \langle \mathbf{e}_1, \mathbf{a}_3 \rangle & \cdots \\ 0 & \langle \mathbf{e}_2, \mathbf{a}_2 \rangle & \langle \mathbf{e}_2, \mathbf{a}_3 \rangle & \cdots \\ 0 & 0 & \langle \mathbf{e}_3, \mathbf{a}_3 \rangle & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$.

Example 18. *Procéder à la décomposition* QR *de la matrice* $\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$.

Correction :

Étape 1 : Vecteurs colonnes

$$\mathbf{a}_1 = \begin{bmatrix}1\\0\\1\end{bmatrix}, \quad \mathbf{a}_2 = \begin{bmatrix}1\\1\\0\end{bmatrix}$$

Étape 2 : Procédé de Gram-Schmidt

Premier vecteur orthogonal

$$\mathbf{q}_1 = \mathbf{a}_1 = \begin{bmatrix}1\\0\\1\end{bmatrix}$$

Deuxième vecteur orthogonal

$\mathbf{q}_2 = \mathbf{a}_2 - \frac{\langle \mathbf{a}_2, \mathbf{q}_1\rangle}{\langle \mathbf{q}_1, \mathbf{q}_1\rangle}\mathbf{q}_1, \quad \langle \mathbf{a}_2, \mathbf{q}_1\rangle = 1\times 1 + 1\times 0 + 0\times 1 = 1, \quad \langle \mathbf{q}_1, \mathbf{q}_1\rangle = \|\mathbf{q}_1\|^2 = 1^2 + 0^2 + 1^2 = 2.$

$$\mathbf{q}_2 = \begin{bmatrix}1\\1\\0\end{bmatrix} - \tfrac{1}{2}\begin{bmatrix}1\\0\\1\end{bmatrix} = \begin{bmatrix}1\\1\\0\end{bmatrix} + \begin{bmatrix}\frac{1}{2}\\0\\-\frac{1}{2}\end{bmatrix} = \begin{bmatrix}\frac{1}{2}\\1\\-\frac{1}{2}\end{bmatrix}.$$

Étape 3 : Vecteurs orthonormés

$$\Rightarrow \mathbf{u}_1 = \tfrac{\mathbf{q}_1}{\|\mathbf{q}_1\|} = \tfrac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\1\end{bmatrix} = \begin{bmatrix}\frac{1}{\sqrt{2}}\\0\\\frac{1}{\sqrt{2}}\end{bmatrix}.$$

$$\Rightarrow \|\mathbf{q}_2\| = \sqrt{\left(\tfrac{1}{2}\right)^2 + 1^2 + \left(-\tfrac{1}{2}\right)^2} = \sqrt{\tfrac{3}{2}}. \quad \mathbf{u}_2 = \tfrac{\mathbf{q}_2}{\|\mathbf{q}_2\|} = \tfrac{1}{\sqrt{3/2}}\begin{bmatrix}\frac{1}{2}\\1\\-\frac{1}{2}\end{bmatrix} = \sqrt{\tfrac{2}{3}}\begin{bmatrix}\frac{1}{2}\\1\\-\frac{1}{2}\end{bmatrix} = \begin{bmatrix}\frac{1}{\sqrt{6}}\\\frac{2}{\sqrt{6}}\\-\frac{1}{\sqrt{6}}\end{bmatrix}.$$

Étape 4 : Matrice Q (3×2)

$$Q = \begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}}\\0 & \frac{2}{\sqrt{6}}\\\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{6}}\end{bmatrix}.$$

Étape 5 : Matrice R (2×2)

$r_{11} = \langle \mathbf{a}_1, \mathbf{u}_1\rangle = \|\mathbf{q}_1\| = \sqrt{2}, \quad r_{12} = \langle \mathbf{a}_2, \mathbf{u}_1\rangle = \frac{\langle \mathbf{a}_2, \mathbf{q}_1\rangle}{\|\mathbf{q}_1\|} = \frac{1}{\sqrt{2}}, \quad r_{22} = \langle \mathbf{a}_2, \mathbf{u}_2\rangle = \|\mathbf{q}_2\| = \sqrt{\frac{3}{2}}.$

Alors, $R = \begin{bmatrix}\sqrt{2} & \frac{1}{\sqrt{2}}\\0 & \sqrt{\frac{3}{2}}\end{bmatrix}.$

Décomposition QR avec Python

[1]:
```python
import sympy as sp

# Définir la matrice A
A = sp.Matrix([[1, 1],[0, 1],[1, 0]])
# Décomposition QR
Q, R = A.QRdecomposition()
# Affichage
print("Q =")
sp.pprint(Q)
print("\nR =")
sp.pprint(R)
```

```
Q =
⎡√2   √6 ⎤
⎢──   ── ⎥
⎢2    6  ⎥
⎢        ⎥
⎢     √6 ⎥
⎢0    ── ⎥
⎢     3  ⎥
⎢        ⎥
⎢√2  -√6 ⎥
⎢──  ─── ⎥
⎣2    6  ⎦

R =
⎡      √2⎤
⎢      ──⎥
⎢√2    2 ⎥
⎢        ⎥
⎢      √6⎥
⎢0     ──⎥
⎣      2 ⎦
```

Figure 13: Décomposition QR

Example 19. Énoncé

Procéder à la décomposition QR de la matrice $\mathbf{A} = \begin{bmatrix} -4 & 5 & 1 \\ -2 & 7 & -1 \\ 4 & 4 & 2 \end{bmatrix}$.

Étape 1 : Vecteurs colonnes

$$\mathbf{a}_1 = \begin{bmatrix} -4 \\ -2 \\ 4 \end{bmatrix}, \quad \mathbf{a}_2 = \begin{bmatrix} 5 \\ 7 \\ 4 \end{bmatrix}, \quad \mathbf{a}_3 = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix}.$$

Étape 2 : Procédé de Gram-Schmidt

Premier vecteur orthogonal

$$\mathbf{q}_1 = \mathbf{a}_1 = \begin{bmatrix} -4 \\ -2 \\ 4 \end{bmatrix}, \quad \|\mathbf{q}_1\| = \sqrt{(-4)^2 + (-2)^2 + 4^2} = \sqrt{16 + 4 + 16} = \sqrt{36} = 6.$$

Deuxième vecteur orthogonal

$\mathbf{q}_2 = \mathbf{a}_2 - \frac{\langle \mathbf{a}_2, \mathbf{q}_1 \rangle}{\langle \mathbf{q}_1, \mathbf{q}_1 \rangle} \mathbf{q}_1,$

$\langle \mathbf{a}_2, \mathbf{q}_1 \rangle = 5 \times (-4) + 7 \times (-2) + 4 \times 4 = -20 - 14 + 16 = -18,$
$\langle \mathbf{q}_1, \mathbf{q}_1 \rangle = \|\mathbf{q}_1\|^2 = 36,$

$$\mathbf{q}_2 = \begin{bmatrix} 5 \\ 7 \\ 4 \end{bmatrix} - \tfrac{-18}{36} \begin{bmatrix} -4 \\ -2 \\ 4 \end{bmatrix} = \begin{bmatrix} 5 \\ 7 \\ 4 \end{bmatrix} + \tfrac{1}{2} \begin{bmatrix} -4 \\ -2 \\ 4 \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \\ 6 \end{bmatrix}.$$

Troisième vecteur orthogonal

$$\mathbf{q}_3 = \mathbf{a}_3 - \tfrac{\langle \mathbf{a}_3, \mathbf{q}_1 \rangle}{\langle \mathbf{q}_1, \mathbf{q}_1 \rangle} \mathbf{q}_1 - \tfrac{\langle \mathbf{a}_3, \mathbf{q}_2 \rangle}{\langle \mathbf{q}_2, \mathbf{q}_2 \rangle} \mathbf{q}_2,$$

$\langle \mathbf{a}_3, \mathbf{q}_1 \rangle = 1 \times (-4) + (-1) \times (-2) + 2 \times 4 = -4 + 2 + 8 = 6,$
$\langle \mathbf{a}_3, \mathbf{q}_2 \rangle = 1 \times 3 + (-1) \times 6 + 2 \times 6 = 3 - 6 + 12 = 9,$
$\langle \mathbf{q}_2, \mathbf{q}_2 \rangle = 81,$

$$\mathbf{q}_3 = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} - \tfrac{6}{36} \begin{bmatrix} -4 \\ -2 \\ 4 \end{bmatrix} - \tfrac{9}{81} \begin{bmatrix} 3 \\ 6 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} - \tfrac{1}{6} \begin{bmatrix} -4 \\ -2 \\ 4 \end{bmatrix} - \tfrac{1}{9} \begin{bmatrix} 3 \\ 6 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} - \begin{bmatrix} -\frac{2}{3} \\ -\frac{1}{3} \\ \frac{2}{3} \end{bmatrix} - \begin{bmatrix} \frac{1}{3} \\ \frac{2}{3} \\ \frac{2}{3} \end{bmatrix} = \begin{bmatrix} \frac{4}{3} \\ -\frac{4}{3} \\ \frac{2}{3} \end{bmatrix} = \tfrac{1}{3} \begin{bmatrix} 4 \\ -4 \\ 2 \end{bmatrix},$$

$\|\mathbf{q}_3\| = \frac{1}{3}\sqrt{4^2 + (-4)^2 + 2^2} = \frac{1}{3}\sqrt{16 + 16 + 4} = \frac{1}{3}\sqrt{36} = 2.$

Étape 3 : Vecteurs orthonormés

$$\mathbf{u}_1 = \tfrac{\mathbf{q}_1}{\|\mathbf{q}_1\|} = \tfrac{1}{6} \begin{bmatrix} -4 \\ -2 \\ 4 \end{bmatrix} = \begin{bmatrix} -\frac{2}{3} \\ -\frac{1}{3} \\ \frac{2}{3} \end{bmatrix}, \quad \mathbf{u}_2 = \tfrac{\mathbf{q}_2}{\|\mathbf{q}_2\|} = \tfrac{1}{9} \begin{bmatrix} 3 \\ 6 \\ 6 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ \frac{2}{3} \\ \frac{2}{3} \end{bmatrix}, \quad \mathbf{u}_3 = \tfrac{\mathbf{q}_3}{\|\mathbf{q}_3\|} = \tfrac{1}{2} \times \tfrac{1}{3} \begin{bmatrix} 4 \\ -4 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{2}{3} \\ -\frac{2}{3} \\ \frac{1}{3} \end{bmatrix}.$$

Étape 4 : Matrice Q

$$Q = \begin{bmatrix} -\frac{2}{3} & \frac{1}{3} & \frac{2}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{2}{3} \\ \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \end{bmatrix}.$$

Étape 5 : Matrice R

$$\textit{On a : } Q^T = \begin{bmatrix} -\frac{2}{3} & -\frac{1}{3} & \frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{2}{3} & \frac{1}{3} \end{bmatrix}, \textit{ alors } R = Q^T A = \begin{bmatrix} -\frac{2}{3} & -\frac{1}{3} & \frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{2}{3} & \frac{1}{3} \end{bmatrix} \times \begin{bmatrix} -4 & 5 & 1 \\ -2 & 7 & -1 \\ 4 & 4 & 2 \end{bmatrix} = \begin{bmatrix} 6 & -3 & 1 \\ 0 & 9 & 1 \\ 0 & 0 & 2 \end{bmatrix}.$$

Résultat final

$$A = QR = \begin{bmatrix} -\frac{2}{3} & \frac{1}{3} & \frac{2}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{2}{3} \\ \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} 6 & -3 & 1 \\ 0 & 9 & 1 \\ 0 & 0 & 2 \end{bmatrix} = \begin{bmatrix} -4 & 5 & 1 \\ -2 & 7 & -1 \\ 4 & 4 & 2 \end{bmatrix}.$$

Décomposition QR avec Python

[4]:
```python
import sympy as sp

# Définir la matrice A
A = sp.Matrix([[-4, 5, 1],[-2, 7,-1],[ 4, 4, 2]])
# Décomposition QR
Q, R = A.QRdecomposition()
# Affichage
print("Q =")
sp.pprint(sp.simplify(Q))
print("\nR =")
sp.pprint(R)
```

```
Q =
⎡-2/3   1/3   2/3 ⎤
⎢                 ⎥
⎢-1/3   2/3   -2/3⎥
⎢                 ⎥
⎣2/3    2/3   1/3 ⎦

R =
⎡6   -3   1⎤
⎢          ⎥
⎢0   9    1⎥
⎢          ⎥
⎣0   0    2⎦
```

Figure 14: Décomposition QR

6.2 Résolution des systèmes linéaires avec la décomposition QR

La décomposition QR est une des méthodes qui joue rôle fondamental en entreprise. Elle permet aussi, comme la décomposition LU, à la résolution du type $Ax = b$. Elle consiste :

1. *à décomposer $A = QR$,*
2. *à résoudre $Rx = Q^{-1}b$ par substitution arrière, où $Q^{-1} = Q^T$.*

Example 20. *Soit le système* $A\mathbf{x} = \mathbf{b}$ *avec* $A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}$, *et* $\mathbf{b} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$.

Décomposition QR par Gram-Schmidt

Vecteurs colonnes :

$$\mathbf{a}_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{a}_2 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}.$$

Orthogonalisation

$$v_1 = \mathbf{a}_1,\ \|v_1\| = \sqrt{2}\ \textit{donc}\ \mathbf{q}_1 = \frac{v_1}{\|v_1\|} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix}.$$

$$\mathbf{v}_2 = \mathbf{a}_2 - \frac{\langle \mathbf{a}_2, v_1 \rangle}{\|v_1\|^2} v_1, \quad \langle \mathbf{a}_2, v_1 \rangle = 1 \cdot 0 + 1 \cdot 1 + 0 \cdot 1 = 1.$$

Donc $\mathbf{v}_2 = \begin{bmatrix}0\\1\\1\end{bmatrix} - \frac{1}{2}\begin{bmatrix}1\\1\\0\end{bmatrix} = \begin{bmatrix}-\frac{1}{2}\\\frac{1}{2}\\1\end{bmatrix}$, $\|\mathbf{v}_2\| = \sqrt{\frac{3}{2}}$ *et* $\mathbf{q}_2 = \frac{\mathbf{v}_2}{\|\mathbf{v}_2\|} = \begin{bmatrix}-\frac{1}{\sqrt{6}}\\\frac{1}{\sqrt{6}}\\\frac{2}{\sqrt{6}}\end{bmatrix}$.

Calcul des matrices Q et R

$$Q = \begin{bmatrix}\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{6}}\\\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}}\\0 & \frac{2}{\sqrt{6}}\end{bmatrix}, \quad R = Q^T A = \begin{bmatrix}\sqrt{2} & \frac{1}{\sqrt{2}}\\0 & \sqrt{\frac{3}{2}}\end{bmatrix}$$

Résolution du système

On résout $R\mathbf{x} = Q^T\mathbf{b}$.

1. *Calculons* $Q^T\mathbf{b}$.
   $Q^T = \begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0\\-\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{2}{\sqrt{6}}\end{bmatrix}$.
   $Q^T\mathbf{b} = \begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0\\-\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{2}{\sqrt{6}}\end{bmatrix} \times \begin{bmatrix}-1\\0\\1\end{bmatrix} = \begin{bmatrix}\frac{1}{\sqrt{2}}\cdot(-1) + \frac{1}{\sqrt{2}}\cdot 0 + 0\cdot 1\\-\frac{1}{\sqrt{6}}\cdot(-1) + \frac{1}{\sqrt{6}}\cdot 0 + \frac{2}{\sqrt{6}}\cdot 1\end{bmatrix} = \begin{bmatrix}-\frac{1}{\sqrt{2}}\\\frac{3}{\sqrt{6}}\end{bmatrix} = \begin{bmatrix}-\frac{1}{\sqrt{2}}\\\frac{\sqrt{6}}{2}\end{bmatrix}$.
2. *Résolution de* $R\mathbf{x} = Q^T\mathbf{b}$

$$\begin{bmatrix}\sqrt{2} & \frac{1}{\sqrt{2}}\\0 & \sqrt{\frac{3}{2}}\end{bmatrix} \times \begin{bmatrix}x_1\\x_2\end{bmatrix} = \begin{bmatrix}-\frac{1}{\sqrt{2}}\\\frac{\sqrt{6}}{2}\end{bmatrix} \Rightarrow \begin{cases}\sqrt{\frac{3}{2}}\cdot x_2 = \frac{\sqrt{6}}{2}\\x_1\cdot\sqrt{2} + \frac{1}{\sqrt{2}}\cdot x_2 = -\frac{1}{\sqrt{2}}\end{cases} \Rightarrow \begin{cases}\sqrt{\frac{3}{2}}\cdot x_2 = \frac{\sqrt{6}}{2}\\x_1\cdot\sqrt{2} + \frac{1}{\sqrt{2}}\cdot 1 = -\frac{1}{\sqrt{2}}\end{cases}$$
$$\Rightarrow \begin{cases}x_2 = \frac{\frac{\sqrt{6}}{2}}{\sqrt{\frac{3}{2}}} = 1\\x_1\cdot\sqrt{2} = -\sqrt{2}\end{cases} \Rightarrow \begin{cases}x_2 = 1\\x_1 = -1\end{cases}.$$

Alors, $x = \begin{bmatrix}-1\\1\end{bmatrix}$.

Système d'équation et QR avec Python

[2]:
```python
import sympy as sp
# Matrices
A = sp.Matrix([[1, 0], [1, 1], [0, 1]])
b = sp.Matrix([-1, 0, 1])

# Décomposition QR
Q, R = A.QRdecomposition()
# Calcul de Q^T b
Qt_b = Q.T * b
# Comme R est triangulaire supérieure 2×2, on résout R x = Q^T b
x = R.solve(Qt_b)
```

[3]:
```python
# Affichage
print("Q ="); sp.pprint(Q)
print("\nR ="); sp.pprint(R)
print("\nQ^T b ="); sp.pprint(Qt_b)
print("\nSolution x ="); sp.pprint(x)
```

```
Q =
⎡√2   -√6 ⎤
⎢──   ────⎥
⎢2     6  ⎥
⎢          ⎥
⎢√2   √6  ⎥
⎢──   ──  ⎥
⎢2     6  ⎥
⎢          ⎥
⎢     √6   ⎥
⎢0    ──   ⎥
⎣     3    ⎦

R =
⎡     √2⎤
⎢√2   ──⎥
⎢     2 ⎥
⎢       ⎥
⎢     √6⎥
⎢0    ──⎥
⎣     2 ⎦

Q^T b =
⎡-√2 ⎤
⎢────⎥
⎢ 2  ⎥
⎢    ⎥
⎢ √6 ⎥
⎢ ── ⎥
⎣ 2  ⎦

Solution x =
⎡-1⎤
⎢  ⎥
⎣1 ⎦
```

Figure 15: Résolution de Ax= b par QR avec python

Exercices (TD-TP)

On considère les matrices $A = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 2 & 1 & 1 \end{pmatrix}$ *et* $b = \begin{pmatrix} 3 \\ 0 \\ 1 \\ 4 \end{pmatrix}$; $A' = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 3 & 2 \\ 1 & 0 & 4 \end{pmatrix}$ *et* $b' = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$.

1. *Calculer les décompositions QR de* A *et* A'.
2. *Résoudre les systèmes d'équations* $Ax = b$ *et* $A'x = b'$.
3. *Résoudre les systèmes d'équations* $Ax = b$ *et* $A'x = b'$ *par la décomposition LU avec python.*

Remark 14.
Les décomposition LU ***et*** QR ***permettent d'apporter facilement et efficacement une solution au problème de***

modélisation, exemple de type $Ax = b$***, plus que par élimination de*** $Gauss$***. La solution ainsi trouvée répond au problème d'optimisation*** $\underset{x \in \mathbb{R}^n}{min} \{\|Ax - b\|\}$ ***qui consiste à trouver l'ensemble des solution qui minimise*** $Ax = b$***.***