Licence Sciences et Techniques
L1 PC & SI - Spécialité Renforcée

UNIVERSITÉ DE TOULON

GNU Octave

Matlab

R21 – Initiation au calcul matriciel avec Matlab/Octave

Recueil de 67 exercices corrigés et aide-mémoire.

https://moodle.univ-tln.fr/course/view.php?id=4875

Année 2023 – 2024

Dernière mise-à-jour : Vendredi 5 janvier 2024

Gloria Faccanoni

https://faccanoni.univ-tln.fr

Table des matières

Ce fascicule est un support pour le cours d'*initiation au calcul matriciel avec Matlab/Octave* de la première année d'une Licence Scientifique. Ce document donne une aperçue de certains outils qui constituent un socle de connaissances indispensables pour votre parcours.
Avec un souci de rigueur, mais sans insister sur les concepts les plus abstraits que ne rencontrera probablement pas un élève, le moindre calcul est détaillé et les difficultés apparaissent progressivement.
Actuellement il est impossible d'aborder ce sujet sans faire des simulations numériques et le langage Octave/Matlab a été choisi comme langage de programmation du cours.

- La documentation et les sources d'Octave peuvent être téléchargées à l'adresse https://www.gnu.org/software/octave/.
  La version en ligne d'Octave est disponible ici https://octave-online.net/.
- L'université de Toulon propose aux étudiants la possibilité de le télécharger et de l'installer sur leur poste MATLAB. Toutes les informations sont ici http://dsiun.univ-tln.fr/MATLAB.html
  Par ailleurs, la version on line de MATLAB est disponible ici https://fr.mathworks.com/products/matlab-online.html Les étudiants et enseignants peuvent s'y connecter avec leurs paramètres universitaires.

R21 – Calcul matriciel		
CM-TP	12h	4 séances de 3h

Séance 1 : Éléments d'analyse matricielle

Séance 2 : Introduction à Matlab/Octave

Séance 3 : TP

Séance 4 : TP

Page moodle du cours (les visiteurs anonymes peuvent y accéder) :
https://moodle.univ-tln.fr/course/view.php?id=4875

Gloria FACCANONI

IMATH Bâtiment M-117
Université de Toulon
Avenue de l'université
83957 LA GARDE - FRANCE

☎ 0033 (0)4 83 16 66 72

✉ gloria.faccanoni@univ-tln.fr
🛈 http://faccanoni.univ-tln.fr

CHAPITRE 1

Éléments d'analyse matricielle

Dans ce chapitre

1.1 Généralité

On appelle **MATRICE** $m \times n$ (ou d'ordre $m \times n$) à coefficients dans $\mathbb{K}$ tout tableau de m lignes et n colonnes d'éléments de $\mathbb{K}$.
L'ensemble des matrices $m \times n$ à coefficients dans $\mathbb{K}$ est noté $\mathscr{M}_{m,n}(\mathbb{K})$.
On convient de noter a_{ij} l'élément de la matrice situé sur la i-ème ligne et j-ème colonne ($1 \leq i \leq m$ et $1 \leq j \leq n$).
Une matrice $\mathbb{A}$ est représentée entre deux parenthèses ou deux crochets :

$$\mathbb{A} = \begin{pmatrix} a_{11} & \dots & a_{1j} & \dots & a_{1n} \\ \vdots & & \vdots & & \vdots \\ a_{i1} & \dots & a_{ij} & \dots & a_{in} \\ \vdots & & \vdots & & \vdots \\ a_{m1} & \dots & a_{mj} & \dots & a_{mn} \end{pmatrix} \quad \text{ou} \quad \mathbb{A} = \begin{bmatrix} a_{11} & \dots & a_{1j} & \dots & a_{1n} \\ \vdots & & \vdots & & \vdots \\ a_{i1} & \dots & a_{ij} & \dots & a_{in} \\ \vdots & & \vdots & & \vdots \\ a_{m1} & \dots & a_{mj} & \dots & a_{mn} \end{bmatrix}$$

ou encore

$$\mathbb{A} = (a_{ij})_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} \quad \text{ou} \quad \mathbb{A} = [a_{ij}]_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}}$$

- ⋆ Si $m = n$ on dit qu'on a une **MATRICE CARRÉE**. L'ensemble des matrices carrées d'ordre n à coefficients dans $\mathbb{K}$ est noté $\mathscr{M}_n(\mathbb{K})$.
- ⋆ Une matrice $m \times 1$ est appelée **VECTEUR-COLONNE** et une matrice $1 \times n$ est appelée **VECTEUR-LIGNE**.
- ⋆ La **MATRICE NULLE**, notée $\mathbb{O}_{m,n}$, est la matrice dont tous les éléments sont nuls : $a_{ij} = 0$ pour tout $i = 1, \dots, m$ et tout $j = 1, \dots, n$.
- ⋆ On appelle **MATRICE DIAGONALE** toute matrice carrée $\mathbb{D} = (d_{ij})_{1 \leq i,j \leq n}$ telle que $i \neq j \implies d_{ij} = 0$. Si on note $d_i \equiv d_{ii}$, une matrice diagonale est de la forme

$$\mathbb{D}_n = \begin{pmatrix} d_1 & 0 & \dots & 0 & 0 \\ 0 & d_2 & \dots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \dots & d_{n-1} & 0 \\ 0 & 0 & \dots & 0 & d_n \end{pmatrix}.$$

On la note $\mathrm{Diag}(d_1, d_2, \dots, d_n)$.

- ⋆ La **MATRICE IDENTITÉ** d'ordre n, notée $\mathbb{I}_n$, est la matrice diagonale $\mathrm{Diag}(\underbrace{1,1,\ldots,1}_{n \text{ fois}})$.
- ⋆ On dit qu'une matrice carrée $\mathbb{A} = (a_{ij})_{1\leq i,j\leq n}$ est
  - ⋆ **TRIANGULAIRE SUPÉRIEURE** si $i > j \implies a_{ij} = 0$,
  - ⋆ **TRIANGULAIRE INFÉRIEURE** si $i < j \implies a_{ij} = 0$.
- ⋆ On appelle matrice **TRANSPOSÉE** de $\mathbb{A}$, notée $\mathbb{A}^T$, la matrice $\mathbb{A} = (a_{ji})_{\substack{1\leq j\leq n\\1\leq i\leq m}}$. C'est donc une matrice de $\mathscr{M}_{n,m}(\mathbb{R})$ obtenue en échangeant lignes et colonnes de la matrice initiale.
- ⋆ Une matrice $\mathbb{A}$ est dite **SYMÉTRIQUE** si $\mathbb{A}^T = \mathbb{A}$, *i.e.* si $a_{ij} = a_{ji}$ pour tout $i \neq j$.
- ⋆ Si $\mathbb{A}$ est une matrice carrée d'ordre n, on définit la **TRACE** de $\mathbb{A}$ comme la somme des éléments de la diagonale principale : $\mathrm{tr}(\mathbb{A}) \equiv \sum_{i=1}^{n} a_{ii}$. Par conséquent $\mathrm{tr}(\mathbb{A}^T) = \mathrm{tr}(\mathbb{A})$.

On remarque qu'une matrice DIAGONALE est triangulaire supérieure et inférieure (*i.e.* $i \neq j \implies a_{ij} = 0$).

EXEMPLE
- ⋆ La matrice $\mathbb{A} = \begin{pmatrix} -1 & 4 & 2 \\ 0 & 1 & -3 \\ 4 & 1 & 5 \end{pmatrix}$ est carrée et d'ordre 3 à coefficients dans $\mathbb{Z}$.
- ⋆ La matrice $\mathbb{U} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & 5 & 6 & 7 \\ 0 & 0 & 2 & -1 \\ 0 & 0 & 0 & -5 \end{pmatrix}$ est une matrice triangulaire supérieure.
- ⋆ La matrice $\mathbb{L} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 \\ 5 & -1 & 2 & 0 \\ 7 & 9 & 15 & 4 \end{pmatrix}$ est une matrice triangulaire inférieure.
- ⋆ La matrice $\mathbb{D} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -8 & 0 & 0 \\ 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ est une matrice diagonale.
- ⋆ La matrice $\mathbb{I}_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ est la matrice identité d'ordre 4.
- ⋆ La matrice $\mathbb{B} = \begin{pmatrix} 7 & 0 & 8 & 2 \end{pmatrix}$ est une matrice ligne (= vecteur ligne) d'ordre 4.
- ⋆ La matrice $\mathbb{C} = \begin{pmatrix} 7 \\ 0 \\ 9 \end{pmatrix}$ est une matrice colonne (= vecteur colonne) d'ordre 3.
- ⋆ La matrice $\mathbb{A} = \begin{pmatrix} 1 & 5 & -9 \\ 5 & 4 & 0 \\ -9 & 0 & 7 \end{pmatrix}$ est symétrique.
- ⋆ Si $\mathbb{A} = \begin{pmatrix} 1 & -1 & 5 \\ 3 & 0 & 7 \end{pmatrix}$ alors $\mathbb{A}^T = \begin{pmatrix} 1 & 3 \\ -1 & 0 \\ 5 & 7 \end{pmatrix}$.
- ⋆ La trace de la matrice $\mathbb{A} = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 2 & 3 \\ 0 & -1 & -2 \end{pmatrix}$ est $\mathrm{tr}(\mathbb{A}) = a_{11} + a_{22} + a_{33} = 1 + 2 + (-2) = 1$.

1.2 Calcul matriciel élémentaire

1.2.1 Addition de matrices

Si $\mathbb{A} = (a_{ij})_{\substack{1\leq i\leq m\\1\leq j\leq n}}$ et $\mathbb{B} = (b_{ij})_{\substack{1\leq i\leq m\\1\leq j\leq n}}$ sont deux matrices $m \times n$, on définit l'**ADDITION** des matrices par

$$\mathbb{A} + \mathbb{B} = (a_{ij} + b_{ij})_{\substack{1\leq i\leq m\\1\leq j\leq n}}.$$

La MATRICE OPPOSÉE D'UNE MATRICE $\mathbb{A}$ est notée $-\mathbb{A}$. Si $\mathbb{A} = (a_{ij})_{\substack{1\leq i\leq m\\1\leq j\leq n}}$ alors $-\mathbb{A} = (-a_{ij})_{\substack{1\leq i\leq m\\1\leq j\leq n}}$.

EXEMPLE
Soient les matrices 2×3 suivantes :

$$\mathbb{A} = \begin{pmatrix} 3 & 4 & 2 \\ 1 & 3 & 5 \end{pmatrix}, \qquad \mathbb{B} = \begin{pmatrix} 6 & 1 & 9 \\ 2 & 0 & 3 \end{pmatrix}.$$

La somme de $\mathbb{A}$ et $\mathbb{B}$ est la matrice 2×3 suivante :

$$\mathbb{A} + \mathbb{B} = \begin{pmatrix} 3+6 & 4+1 & 2+9 \\ 1+2 & 3+0 & 5+3 \end{pmatrix} = \begin{pmatrix} 9 & 5 & 11 \\ 3 & 3 & 8 \end{pmatrix}.$$

ATTENTION
La somme de deux matrices d'ordres différents n'est pas définie.

Si $\mathbb{A}$, $\mathbb{B}$ et $\mathbb{C}$ sont des matrices de même ordre, alors nous avons

1. $\mathbb{A}+\mathbb{B}=\mathbb{B}+\mathbb{A}$ (commutativité),
2. $\mathbb{A}+(\mathbb{B}+\mathbb{C})=(\mathbb{A}+\mathbb{B})+\mathbb{C}$ (associativité),
3. $(\mathbb{A}+\mathbb{B})^T=\mathbb{A}^T+\mathbb{B}^T$
4. $\mathrm{tr}(\mathbb{A}+\mathbb{B})=\mathrm{tr}(\mathbb{A})+\mathrm{tr}(\mathbb{B})$.

EXEMPLE
Soient les matrices 2×2 suivantes :

$$\mathbb{A}=\begin{pmatrix}1 & -1\\ 3 & 0\end{pmatrix},\qquad \mathbb{B}=\begin{pmatrix}6 & -5\\ 2 & 1\end{pmatrix},\qquad \mathbb{C}=\begin{pmatrix}0 & 2\\ 2 & 4\end{pmatrix}.$$

On a alors

$$\mathbb{A}+\mathbb{B}=\begin{pmatrix}1+6 & -1-5\\ 3+2 & 0+1\end{pmatrix}=\begin{pmatrix}7 & -6\\ 5 & 1\end{pmatrix},\quad \mathbb{B}+\mathbb{A}=\begin{pmatrix}6+1 & -5-1\\ 2+3 & 1+0\end{pmatrix}=\begin{pmatrix}7 & -6\\ 5 & 1\end{pmatrix},\quad \mathbb{B}+\mathbb{C}=\begin{pmatrix}6+0 & -5+2\\ 2+2 & 1+4\end{pmatrix}=\begin{pmatrix}6 & -3\\ 4 & 5\end{pmatrix}.$$

De plus,

$$(\mathbb{A}+\mathbb{B})+\mathbb{C}=\begin{pmatrix}7 & -4\\ 7 & 5\end{pmatrix},\qquad \mathbb{A}+(\mathbb{B}+\mathbb{C})=\begin{pmatrix}7 & -4\\ 7 & 5\end{pmatrix}.$$

1.2.2 Produit d'une matrice par un scalaire

Si $\mathbb{A}=(a_{ij})_{\substack{1\le i\le m\\ 1\le j\le n}}$ est une matrice $m\times n$ et si $\alpha\in\mathbb{K}$, on définit le **PRODUIT D'UNE MATRICE PAR UN SCALAIRE** comme la matrice

$$\alpha\cdot\mathbb{A}=(\alpha\cdot a_{ij})_{\substack{1\le i\le m\\ 1\le j\le n}}$$

Si $\mathbb{A}$ et $\mathbb{B}$ sont deux matrices de même ordre et $\alpha\in\mathbb{K}$ un scalaire, alors $\alpha\cdot(\mathbb{A}+\mathbb{B})=\alpha\cdot\mathbb{A}+\alpha\cdot\mathbb{B}$ (distributivité).

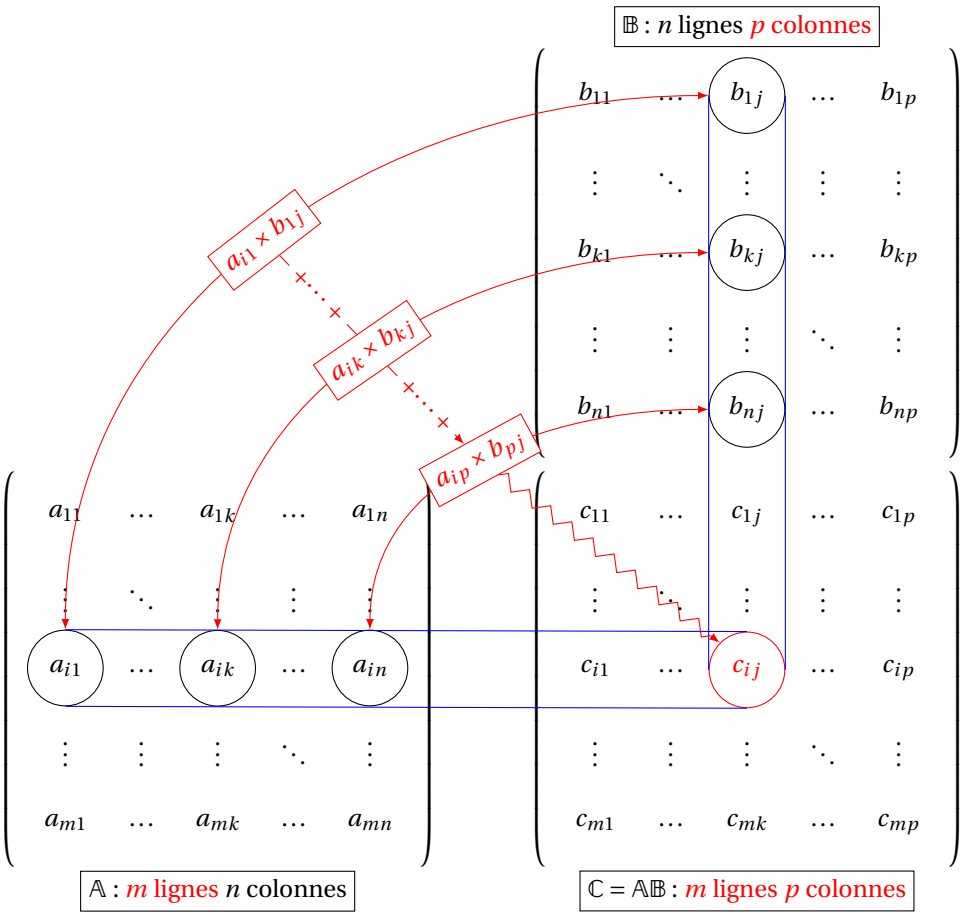
De plus, $(\alpha\mathbb{A})^T=\alpha\mathbb{A}^T$.

EXEMPLE
Si $\alpha=\frac{1}{2}$ et $\mathbb{A}=\begin{pmatrix}3 & 4 & 2\\ 1 & 3 & 5\end{pmatrix}$ alors $\alpha\cdot\mathbb{A}=\begin{pmatrix}3/2 & 2 & 1\\ 1/2 & 3/2 & 5/2\end{pmatrix}$.

1.2.3 Produit de matrices

Si $\mathbb{A}=(a_{ik})_{\substack{1\le i\le m\\ 1\le k\le n}}$ est une matrice $m\times n$ et $\mathbb{B}=(b_{kj})_{\substack{1\le k\le n\\ 1\le j\le p}}$ une matrice $n\times p$, on définit $\mathbb{C}=\mathbb{A}\mathbb{B}$ le **PRODUIT DES MATRICES** $\mathbb{A}$ et $\mathbb{B}$ (dans l'ordre) comme la matrice de dimension $m\times p$ telle que l'élément c_{ij} est le produit scalaire de la ligne i de $\mathbb{A}$ et de la colonne j de $\mathbb{B}$. par

$$c_{ij}=\sum_{k=1}^{n}a_{ik}b_{kj}=a_{i1}b1j+a_{i2}b2j+\cdots+a_{in}bnj$$



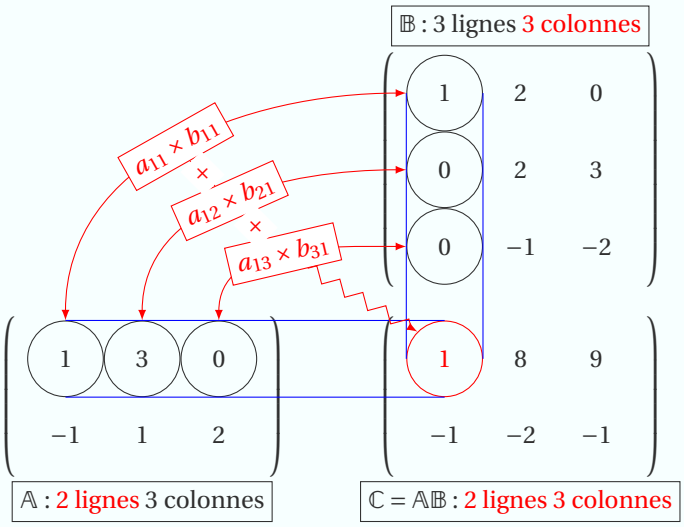
EXEMPLE

Soient les deux matrices

$$\mathbb{A} = \begin{pmatrix} 1 & 3 & 0 \\ -1 & 1 & 2 \end{pmatrix} \quad \text{et} \quad \mathbb{B} = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 2 & 3 \\ 0 & -1 & -2 \end{pmatrix}.$$

La matrice $\mathbb{A}$ est d'ordre 2×3, la matrice $\mathbb{B}$ est d'ordre 3×3, donc la matrice produit $\mathbb{AB}$ est une matrice d'ordre 2×3 :

$$\mathbb{AB} = \begin{pmatrix} 1\times1+3\times0+0\times0 & 1\times2+3\times2+0\times(-1) & 1\times0+3\times3+0\times(-2) \\ -1\times1+1\times0+2\times0 & -1\times2+1\times2+2\times(-1) & -1\times0+1\times3+2\times(-2) \end{pmatrix} = \begin{pmatrix} 1 & 8 & 9 \\ -1 & -2 & -1 \end{pmatrix}.$$



EXEMPLE
Une société commerciale possède deux magasins dont l'aménagement du parc informatique est le suivant :
- ⋆ Magasin 1 : 12 PC, 5 tablettes et 10 smartphones,
- ⋆ Magasin 2 : 17 PC, 6 tablettes et 14 smartphones.

On peut associer à cet équipement la matrice $\mathbb{M} = \left(\begin{smallmatrix} 12 & 5 & 10 \\ 17 & 6 & 14 \end{smallmatrix}\right)$.
La société souhaite améliorer son équipement de la manière suivante :
- ⋆ Magasin 1 : +3 PC, +2 tablettes et +2 smartphones,
- ⋆ Magasin 2 : +5 PC, +3 tablettes et +4 smartphones.

Ce nouvel équipement peut être associé à la matrice $\mathbb{N} = \left(\begin{smallmatrix} 3 & 2 & 2 \\ 5 & 3 & 4 \end{smallmatrix}\right)$.
Le répartition du nouvel aménagement du parc informatique des deux magasins sera donc

$$\mathbb{M}+\mathbb{N} = \begin{pmatrix} 12 & 5 & 10 \\ 17 & 6 & 14 \end{pmatrix} + \begin{pmatrix} 3 & 2 & 2 \\ 5 & 3 & 4 \end{pmatrix} = \begin{pmatrix} 15 & 7 & 12 \\ 22 & 9 & 18 \end{pmatrix}$$

Pour acheter le nouvel équipement, la société commerciale a le choix entre deux fournisseurs :
- ⋆ Fournisseur 1 : 600 e le PC, 180 e la tablette et 60 e le smartphone,
- ⋆ Fournisseur 2 : 550 e le PC, 200 e la tablette et 50 e le smartphone.

On peut associer ces prix à la matrice $\mathbb{P} = \left(\begin{smallmatrix} 600 & 550 \\ 180 & 200 \\ 60 & 50 \end{smallmatrix}\right)$.
On obtient les prix du nouvel aménagement selon les magasins et selon les fournisseurs en calculant

$$\mathbb{NP} = \begin{pmatrix} 3 & 2 & 2 \\ 5 & 3 & 4 \end{pmatrix} \begin{pmatrix} 600 & 550 \\ 180 & 200 \\ 60 & 50 \end{pmatrix} = \begin{pmatrix} 3\times 600+2\times 180+2\times 60 & 3\times 550+2\times 200+2\times 50 \\ 5\times 600+3\times 180+4\times 60 & 5\times 550+3\times 200+4\times 50 \end{pmatrix} = \begin{pmatrix} 2280 & 2150 \\ 3780 & 3550 \end{pmatrix}$$

Par exemple, le prix de l'investissement pour le magasin 1 est de 2280 e avec le fournisseur 1 et de 2150 e avec le fournisseur 2.

ATTENTION
$\mathbb{AB} \neq \mathbb{BA}$ en général (non commutativité).
Prenons le cas général avec $\mathbb{A}$ d'ordre $m \times p$ et $\mathbb{B}$ d'ordre $p \times n$. Le produit $\mathbb{AB}$ est défini, c'est une matrice d'ordre $m \times n$. Qu'en est-il du produit $\mathbb{BA}$? Il faut distinguer trois cas :
- ⋆ si $m \neq n$ le produit $\mathbb{BA}$ n'est pas défini ;
- ⋆ si $m = n$ mais $p \neq n$, le produit $\mathbb{AB}$ est défini et c'est une matrice d'ordre $m \times n$ tandis que le produit $\mathbb{BA}$ est défini mais c'est une matrice d'ordre $p \times p$ donc $\mathbb{AB} \neq \mathbb{BA}$;
- ⋆ si $m = n = p$, $\mathbb{A}$ et $\mathbb{B}$ sont deux matrices carrées d'ordre m. Les produits $\mathbb{AB}$ et $\mathbb{BA}$ sont aussi carrés et d'ordre m mais là encore, en général, $\mathbb{AB} \neq \mathbb{BA}$;

EXEMPLE
Soient les matrices

$$\mathbb{A} = \begin{pmatrix} 1 & -1 \\ 3 & 0 \end{pmatrix}, \qquad \mathbb{B} = \begin{pmatrix} 6 & -5 \\ 2 & 1 \end{pmatrix}.$$

On obtient

$$\mathbb{AB} = \begin{pmatrix} 4 & -6 \\ 18 & -15 \end{pmatrix} \quad \text{et} \quad \mathbb{BA} = \begin{pmatrix} -9 & -6 \\ 5 & -2 \end{pmatrix}.$$

Si les dimensions sont compatibles, on a les propriétés suivantes :

1. $\mathbb{A}(\mathbb{BC}) = (\mathbb{AB})\mathbb{C}$ (associativité)
2. $\mathbb{A}(\mathbb{B}+\mathbb{C}) = \mathbb{AB}+\mathbb{AC}$ (distributivité)
3. $\mathbb{AI}_n = \mathbb{I}_n\mathbb{A} = \mathbb{A}$
4. $(\mathbb{AB})^T = \mathbb{B}^T\mathbb{A}^T$
5. $\mathrm{tr}(\mathbb{AB}) = \mathrm{tr}(\mathbb{BA})$

1.2.4 Puissance d'une matrice

Si $\mathbb{A}$ est une matrice carrée, on note $\mathbb{B} = \mathbb{A}^q$ (pour $q \geq 2$) la matrice définie par

$$\mathbb{B} \equiv \underbrace{\mathbb{A} \times \mathbb{A} \times \cdots \times \mathbb{A}}_{\text{q fois}}.$$

Il s'agit du produit matriciel de $\mathbb{A}$ par elle même q fois par conséquent, en générale, $b_{ij} \neq (a_{ij})^q$.
Si la matrice est diagonale, *i.e.* si $a_{ij} = 0$ pour $i \neq j$, alors $b_{ij} = (a_{ij})^q$.

1.2.5 Produit matriciel d'Hadamard (produit "pointé")

Le produit matriciel de HADAMARD est une opération qui pour deux matrices **de mêmes dimensions**, associe une autre matrice, de même dimension, où chaque coefficient est le produit terme à terme des deux matrices. Dans ce polycopié on le notera ".∗" :

$$\mathbb{H} \overset{\text{def}}{=} \mathbb{A}.*\mathbb{B} \quad \Longleftrightarrow \quad h_{ij} = a_{ij}b_{ij}.$$

Il est associatif, distributif et, contrairement au produit matriciel classique, commutatif.

1.2.6 Inverse d'une matrice carrée et systèmes linéaires carrés

Une matrice carrée $\mathbb{A} \in \mathscr{M}_n(\mathbb{K})$ est dite **INVERSIBLE** (ou régulière) s'il existe une matrice $\mathbb{B} \in \mathscr{M}_n(\mathbb{K})$ telle que

$$\mathbb{AB} = \mathbb{BA} = \mathbb{I}_n.$$

Si une telle matrice existe, alors elle est unique, on la note $\mathbb{A}^{-1}$ et on l'appelle matrice **INVERSE** de $\mathbb{A}$.

- ⋆ Si une matrice est non inversible (*i.e.* il n'existe pas $\mathbb{A}^{-1}$), on dit qu'elle est SINGULIÈRE.
- ⋆ Une matrice carrée $\mathbb{A}$ est dite ORTHOGONALE si elle est inversible et $\mathbb{A}^T\mathbb{A} = \mathbb{A}\mathbb{A}^T = \mathbb{I}_n$, *i.e.* si $\mathbb{A}^T = \mathbb{A}^{-1}$.

Soit $\mathbb{A}$ et $\mathbb{B}$ deux matrices inversibles, alors

- ⋆ $\mathbb{A}^{-1}$ l'est aussi et $\left(\mathbb{A}^{-1}\right)^{-1} = \mathbb{A}$,
- ⋆ $\mathbb{A}^T$ l'est aussi et $\left(\mathbb{A}^T\right)^{-1} = \left(\mathbb{A}^{-1}\right)^T$,
- ⋆ $\mathbb{AB}$ l'est aussi et $(\mathbb{AB})^{-1} = \mathbb{B}^{-1}\mathbb{A}^{-1}$.

EXEMPLE
Considérons les matrices

$$\mathbb{A} = \begin{pmatrix} 1 & 1 \\ 2 & 4 \end{pmatrix} \qquad \mathbb{B} = \begin{pmatrix} 2 & -\frac{1}{2} \\ -1 & \frac{1}{2} \end{pmatrix}.$$

On a

$$\mathbb{AB} = \begin{pmatrix} 1 & 1 \\ 2 & 4 \end{pmatrix}\begin{pmatrix} 2 & -\frac{1}{2} \\ -1 & \frac{1}{2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbb{I}_2 \qquad \mathbb{BA} = \begin{pmatrix} 2 & -\frac{1}{2} \\ -1 & \frac{1}{2} \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 2 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbb{I}_2$$

On dit que $\mathbb{B}$ est la matrice inverse de $\mathbb{A}$ et réciproquement.

Remarque
Matrice inverse et systèmes linéaires Il est fréquent, dans toutes les disciplines scientifiques, de devoir résoudre des systèmes linéaires.
Tout système linéaire de n équations à n inconnues peut s'écrire sous la forme matricielle $\mathbb{A}\mathbf{x} = \mathbf{b}$; $\mathbb{A}$ est une matrice carrée de dimension n et $\mathbf{x}$ et $\mathbf{b}$ sont des vecteurs colonnes de dimension n, où $\mathbf{x}$ est l'inconnue et $\mathbf{b}$ un vecteur donné.
Si $\mathbb{A}$ est inversible alors ce système possède une unique solution $\mathbf{x}$ donnée par $\mathbf{x} = \mathbb{A}^{-1}\mathbf{b}$ car

$$\mathbb{A}\mathbf{x} = \mathbf{b} \quad \Longleftrightarrow \quad \mathbb{A}^{-1}\mathbb{A}\mathbf{x} = \mathbb{A}^{-1}\mathbf{b} \quad \Longleftrightarrow \quad \mathbf{x} = \mathbb{A}^{-1}\mathbf{b}.$$

EXEMPLE
Considérons le système linéaire

$$\begin{cases} 2x + 3y = 15 \\ 3x + 4y = 12 \end{cases}$$

Si on pose $\mathbb{A} = \left(\begin{smallmatrix} 2 & 3 \\ 3 & 4 \end{smallmatrix}\right)$, $\mathbf{x} = \left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right)$ et $\mathbf{b} = \left(\begin{smallmatrix} 15 \\ 12 \end{smallmatrix}\right)$, alors le produit matriciel $\mathbb{A}\mathbf{x}$ donne le vecteur colonne

$$\mathbb{A}\mathbf{x} = \begin{pmatrix} 2 & 3 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 2x + 3y \\ 3x + 4y \end{pmatrix}$$

ainsi le système peut s'écrire sous forme matricielle $\mathbb{A}\mathbf{x} = \mathbf{b}$.
On cherche donc à calculer $\mathbb{A}^{-1}$, *i.e.* on cherche a, b, c, d tels que

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} 2 & 3 \\ 3 & 4 \end{pmatrix} = \begin{pmatrix} 2 & 3 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

On a

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} 2 & 3 \\ 3 & 4 \end{pmatrix} = \begin{pmatrix} 2a+3b & 3a+4b \\ 2c+3d & 3c+4d \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$
$$\begin{pmatrix} 2 & 3 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 2a+3c & 2b+3d \\ 3a+4c & 3b+4d \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

si et seulement si $a = -4$, $b = c = 3$ et $d = -2$ ainsi

$$\mathbf{x} = \mathbb{A}^{-1}\mathbf{b} = \begin{pmatrix} -4 & 3 \\ 3 & -2 \end{pmatrix}\begin{pmatrix} 15 \\ 12 \end{pmatrix} = \begin{pmatrix} -24 \\ 21 \end{pmatrix}$$

Cet exemple montre le lien entre résolution d'un système linéaire et calcul d'une matrice inverse. Cependant, pour calculer la solution du système initiale de 2 équations à 2 inconnues, on doit calculer les 4 coefficients de $\mathbb{A}^{-1}$ et pour cela on doit résoudre un système linéaire de 8 équations et 4 inconnues… ce n'est pas la bonne stratégie!

1.3 Définition et calcul pratique du déterminant

1.3.1 Déterminant d'une matrice d'ordre n (règle de Laplace)

Soit $\mathbb{A}$ une matrice carrée d'ordre n.
Étant donné un couple (i, j) d'entiers, $1 \leq i, j \leq n$, on note $\mathbb{A}_{ij}$ la matrice carrée d'ordre $n-1$ obtenue en supprimant la i-ème ligne et la j-ème colonne de $\mathbb{A}$.
Le DÉTERMINANT de $\mathbb{A}$, noté $\det(\mathbb{A})$ ou $|\mathbb{A}|$, est défini par récurrence sur l'ordre de la matrice $\mathbb{A}$:

- ⋆ si $n = 1$: le déterminant de $\mathbb{A}$ est le nombre
$$\det(\mathbb{A}) \equiv a_{11},$$
- ⋆ si $n > 1$: le déterminant de $\mathbb{A}$ est le nombre
$$\det(\mathbb{A}) \equiv \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det(\mathbb{A}_{ij}) \qquad \text{quelque soit la ligne } i \text{ fixée, } 1 \leq i \leq n,$$
ou, de manière équivalente, le nombre
$$\det(\mathbb{A}) \equiv \sum_{i=1}^{n} (-1)^{i+j} a_{ij} \det(\mathbb{A}_{ij}) \qquad \text{quelque soit la colonne } j \text{ fixée, } 1 \leq j \leq n.$$

Astuce
Pour se souvenir des signes de ces deux formules, on peut remarquer que la distribution des signes + et − avec la formule $(-1)^{i+j}$ est analogue à la distribution des cases noirs et blanches sur un damier :

$$\begin{vmatrix} + & - & + & - & \dots \\ - & + & - & + & \dots \\ + & - & + & - & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{vmatrix}$$

EXEMPLE (DÉTERMINANT D'UNE MATRICE D'ORDRE 2 — MÉTHODE DE LAPLACE)
Soit la matrice

$$\mathbb{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

alors

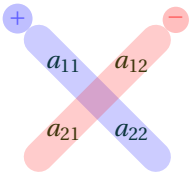
$$\det(\mathbb{A}_{11}) = a_{22}, \qquad \det(\mathbb{A}_{12}) = a_{21}, \qquad \det(\mathbb{A}_{21}) = a_{12}, \qquad \det(\mathbb{A}_{22}) = a_{11}.$$

On peut calculer $\det(\mathbb{A})$ par l'une des formules suivantes :

- $a_{11}\det(\mathbb{A}_{11}) - a_{12}\det(\mathbb{A}_{12}) = a_{11}a_{22} - a_{12}a_{21}$ (développement suivant la ligne $i = 1$)
- $-a_{21}\det(\mathbb{A}_{21}) + a_{22}\det(\mathbb{A}_{22}) = -a_{21}a_{12} + a_{22}a_{11}$ (développement suivant la ligne $i = 2$)
- $a_{11}\det(\mathbb{A}_{11}) - a_{21}\det(\mathbb{A}_{21}) = a_{11}a_{22} - a_{21}a_{12}$ (développement suivant la colonne $j = 1$)
- $-a_{12}\det(\mathbb{A}_{12}) + a_{22}\det(\mathbb{A}_{22}) = -a_{12}a_{21} + a_{22}a_{11}$ (développement suivant la colonne $j = 2$)

Astuce (Déterminant d'une matrice d'ordre 2 — méthode pratique)
Soit $\mathbb{A}$ une matrice carrée d'ordre $n = 2$. Sans appliquer la méthode de Laplace, nous pouvons nous rappeler du déterminant par le schéma suivant :

$$\det(\mathbb{A}) = \det\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = a_{11}a_{22} - a_{12}a_{21}.$$



EXEMPLE

$$\det\begin{pmatrix} 5 & 7 \\ 4 & 3 \end{pmatrix} = 5 \times 3 - 7 \times 4 = 15 - 28 = -13.$$

EXEMPLE (DÉTERMINANT D'UNE MATRICE D'ORDRE 3 — MÉTHODE DE LAPLACE)
Soit la matrice

$$\mathbb{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

alors

$$\det(\mathbb{A}_{11}) = \det\begin{pmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{pmatrix} = a_{22}a_{33} - a_{23}a_{32}, \qquad \det(\mathbb{A}_{12}) = \det\begin{pmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{pmatrix} = a_{21}a_{33} - a_{23}a_{31},$$

$$\det(\mathbb{A}_{13}) = \det\begin{pmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{pmatrix} = a_{21}a_{32} - a_{22}a_{31}, \qquad \det(\mathbb{A}_{21}) = \det\begin{pmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{pmatrix} = a_{12}a_{33} - a_{13}a_{32},$$

$$\det(\mathbb{A}_{22}) = \det\begin{pmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{pmatrix} = a_{11}a_{33} - a_{13}a_{31}, \qquad \det(\mathbb{A}_{23}) = \det\begin{pmatrix} a_{11} & a_{12} \\ a_{31} & a_{32} \end{pmatrix} = a_{11}a_{32} - a_{12}a_{31},$$

$$\det(\mathbb{A}_{31}) = \det\begin{pmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{pmatrix} = a_{12}a_{23} - a_{13}a_{22}, \qquad \det(\mathbb{A}_{32}) = \det\begin{pmatrix} a_{11} & a_{13} \\ a_{21} & a_{23} \end{pmatrix} = a_{11}a_{23} - a_{13}a_{21},$$

$$\det(\mathbb{A}_{33}) = \det\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = a_{11}a_{22} - a_{12}a_{21},$$

donc on peut calculer $\det(\mathbb{A})$ *par l'une des formules suivantes* :

- $a_{11}\det(\mathbb{A}_{11}) - a_{12}\det(\mathbb{A}_{12}) + a_{13}\det(\mathbb{A}_{13})$ (développement suivant la ligne $i = 1$)
- $-a_{21}\det(\mathbb{A}_{21}) + a_{22}\det(\mathbb{A}_{22}) - a_{23}\det(\mathbb{A}_{23})$ (développement suivant la ligne $i = 2$)
- $a_{31}\det(\mathbb{A}_{31}) - a_{32}\det(\mathbb{A}_{32}) + a_{33}\det(\mathbb{A}_{33})$ (développement suivant la ligne $i = 3$)
- $-a_{11}\det(\mathbb{A}_{11}) + a_{21}\det(\mathbb{A}_{21}) - a_{31}\det(\mathbb{A}_{31})$ (développement suivant la colonne $j = 1$)
- $a_{12}\det(\mathbb{A}_{12}) - a_{22}\det(\mathbb{A}_{22}) + a_{32}\det(\mathbb{A}_{32})$ (développement suivant la colonne $j = 2$)

⋆ $-a_{13}\det(\mathbb{A}_{13}) + a_{23}\det(\mathbb{A}_{23}) - a_{33}\det(\mathbb{A}_{33})$ (développement suivant la colonne $j=3$)

Quelques calculs montrent que ces formules donnent bien le même résultat.

EXEMPLE

Soit la matrice

$$\mathbb{A} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 3 & 5 \end{pmatrix}$$

alors

$$\det(\mathbb{A}_{11}) = \det\begin{pmatrix} 2 & 0 \\ 3 & 5 \end{pmatrix} = 10, \quad \det(\mathbb{A}_{12}) = \det\begin{pmatrix} 0 & 0 \\ 0 & 5 \end{pmatrix} = 0, \quad \det(\mathbb{A}_{13}) = \det\begin{pmatrix} 0 & 2 \\ 0 & 3 \end{pmatrix} = 0,$$

$$\det(\mathbb{A}_{21}) = \det\begin{pmatrix} 0 & 1 \\ 3 & 5 \end{pmatrix} = -3, \quad \det(\mathbb{A}_{22}) = \det\begin{pmatrix} 1 & 1 \\ 0 & 5 \end{pmatrix} = 5, \quad \det(\mathbb{A}_{23}) = \det\begin{pmatrix} 1 & 0 \\ 0 & 3 \end{pmatrix} = 3,$$

$$\det(\mathbb{A}_{31}) = \det\begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix} = -2, \quad \det(\mathbb{A}_{32}) = \det\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} = 0, \quad \det(\mathbb{A}_{33}) = \det\begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} = 2,$$

donc on peut calculer $\det(\mathbb{A})$ par l'une des formules suivantes :

⋆ $1\det(\mathbb{A}_{11}) + 0\det(\mathbb{A}_{12}) + 1\det(\mathbb{A}_{13}) = 10+0+0 = 10$

⋆ $0\det(\mathbb{A}_{21}) + 2\det(\mathbb{A}_{22}) + 0\det(\mathbb{A}_{23}) = 0+2\times 5+0 = 10$ ←-- formule pratique car il n'y a qu'un déterminant à calculer

⋆ $0\det(\mathbb{A}_{31}) + 3\det(\mathbb{A}_{32}) + 5\det(\mathbb{A}_{33}) = 0+0+5\times 2 = 10$

⋆ $1\det(\mathbb{A}_{11}) + 0\det(\mathbb{A}_{21}) + 0\det(\mathbb{A}_{31}) = 10+0+0 = 10$ ←-- formule pratique car il n'y a qu'un déterminant à calculer

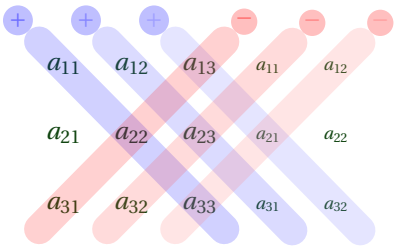
⋆ $0\det(\mathbb{A}_{12}) + 2\det(\mathbb{A}_{22}) + 3\det(\mathbb{A}_{32}) = 0+2\times 5+0 = 10$

⋆ $1\det(\mathbb{A}_{13}) + 0\det(\mathbb{A}_{23}) + 5\det(\mathbb{A}_{33}) = 0+0+5\times 2 = 10$

Astuce (Déterminant d'une matrice d'ordre 3 — méthode pratique (règle de SARRUS))

Soit $\mathbb{A}$ une matrice carrée d'ordre $n=3$. Sans appliquer la méthode de Laplace, nous pouvons nous rappeler du déterminant par le schéma suivant :

$$\det(\mathbb{A}) = \det\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \left(a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32}\right) - \left(a_{13}a_{22}a_{31} + a_{11}a_{23}a_{32} + a_{12}a_{21}a_{33}\right)$$

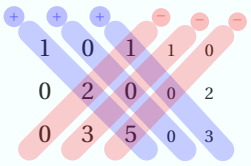


EXEMPLE

Soit la matrice

$$\mathbb{A} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 3 & 5 \end{pmatrix}$$

alors avec la règle de SARRUS



$$\det(\mathbb{A}) = (1\times 2\times 5 + 0\times 0\times 0 + 1\times 0\times 3) - (1\times 2\times 0 + 1\times 0\times 3 + 0\times 0\times 5) = 10.$$

Si on utilise la définition (règle de LAPLACE), en développant selon la première colonne on obtient

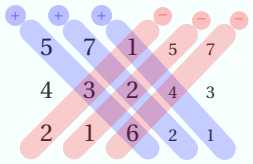
$$\det(\mathbb{A}) = 1\times \det\begin{pmatrix} 2 & 0 \\ 3 & 5 \end{pmatrix} = 2\times 5 - 0\times 3 = 10.$$

EXEMPLE

Soit la matrice

$$\mathbb{A} = \begin{pmatrix} 5 & 7 & 1 \\ 4 & 3 & 2 \\ 2 & 1 & 6 \end{pmatrix}$$

alors



$$\det(\mathbb{A}) = (5\times 3\times 6 + 7\times 2\times 2 + 1\times 4\times 1) - (1\times 3\times 2 + 5\times 2\times 1 + 7\times 4\times 6) = -62.$$

ATTENTION

La règle de SARRUS ne s'applique qu'à des matrices d'ordre 3.

EXEMPLE

Soit la matrice d'ordre 4 suivante :

$$\mathbb{A} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 2 & 0 & 1 & 0 \\ 1 & 2 & 0 & 4 \\ 1 & 2 & 3 & 0 \end{pmatrix}$$

Alors

$$\det(\mathbb{A}) = \det(\mathbb{A}_{11}) - \det(\mathbb{A}_{14}) = \det\begin{pmatrix} 0 & 1 & 0 \\ 2 & 0 & 4 \\ 2 & 3 & 0 \end{pmatrix} - \det\begin{pmatrix} 2 & 0 & 1 \\ 1 & 2 & 0 \\ 1 & 2 & 3 \end{pmatrix} = -\det\begin{pmatrix} 2 & 4 \\ 2 & 0 \end{pmatrix} - \big(12+0+2-2-0-0\big) = -(-8) - 12 = -4.$$

Si on essaye de «généraliser» la règle de SARRUS on n'obtient pas le bon résultat :

$$\big(1\times 0\times 0\times 0 + 0\times 1\times 4\times 1 + 0\times 0\times 1\times 2 + 1\times 2\times 2\times 3\big) - \big(1\times 1\times 2\times 1 + 1\times 0\times 0\times 2 + 0\times 2\times 4\times 3 + 0\times 0\times 1\times 0\big) = 10.$$

On a les propriétés suivantes :

1. $\mathbb{A}$ est inversible si et seulement si $\det(\mathbb{A}) \neq 0$,
2. $\det(\mathbb{A}^{-1}) = \dfrac{1}{\det(\mathbb{A})}$,
3. $\det(\mathbb{A}^T) = \det(\mathbb{A})$,
4. $\det(\mathbb{A}\mathbb{B}) = \det(\mathbb{A})\cdot\det(\mathbb{B})$
5. le déterminant d'une matrice triangulaire est égal au produit des éléments diagonaux,
6. le déterminant d'une matrice orthogonale est égal à 1.

Astuce

Il convient d'utiliser la définition de déterminant après avoir fait apparaître sur une même rangée le plus possible de zéro sachant que

- si deux colonnes (resp. deux lignes) sont identiques ou proportionnelles, alors $\det(\mathbb{A}) = 0$;

- si on multiplie une colonne (resp. une ligne) par un scalaire $\alpha \neq 0$, alors le déterminant est multiplié par α ;
- si on échange deux colonnes (resp. deux lignes), alors le déterminant est changé en son opposé (*i.e.*, le déterminant change de signe) ;
- on ne change pas un déterminant si on ajoute à une colonne (resp. une ligne) une combinaison linéaire des autres colonnes (resp. lignes), *i.e.*

$$C_i \leftarrow C_i + \alpha C_j, \qquad L_i \leftarrow L_i + \alpha L_j,$$

avec $j \neq i$ et $\alpha \neq 0$.

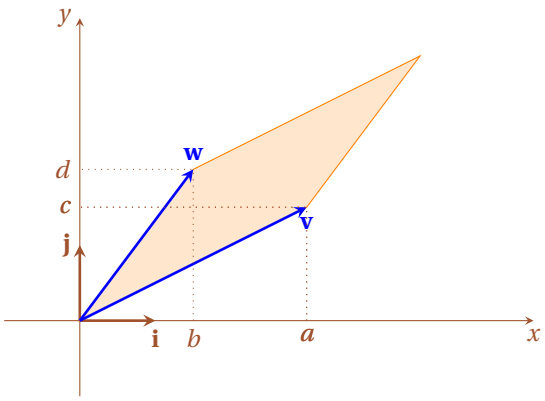
EXEMPLE
Soit la matrice

$$\mathbb{A} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 3 & 5 \end{pmatrix}$$

On fait apparaître encore plus de zéros dans la matrice jusqu'à obtenir une matrice triangulaire :

$$\det(\mathbb{A}) = \det \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 3 & 5 \end{pmatrix} \overset{L_3 \leftarrow L_3 - \frac{3}{2} L_2}{=} \det \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 5 \end{pmatrix} = 1 \times 2 \times 5 = 10.$$

En dimension 2 les déterminants correspondent à des aires et en dimension 3 à des volumes.
Considérons deux vecteurs $\mathbf{v} = \binom{a}{c}$ et $\mathbf{w} = \binom{b}{d}$ du plan $\mathbb{R}^2$. Ces deux vecteurs déterminent un parallélogramme :



L'aire du parallélogramme est donnée par la valeur absolue du déterminant de la matrice dont les colonnes sont **v** et **w** :

$$\text{Aire} = \left| \det \begin{pmatrix} a & c \\ b & d \end{pmatrix} \right|$$

1.3.2 Rang d'une matrice

Le RANG d'une matrice quelconque $\mathbb{A} \in \mathcal{M}_{m,n}$, noté $\text{rg}(\mathbb{A})$, est égal au plus grand entier s tel que l'on puisse extraire de $\mathbb{A}$ une matrice carrée d'ordre s inversible, c'est-à-dire de déterminant non nul. Il représente le nombre maximum de vecteurs colonnes de $\mathbb{A}$ linéairement indépendants (ou, ce qui est équivalent, le nombre maximum de vecteurs lignes linéairement indépendants).

Remarque
Soit une matrice $\mathbb{A} \in \mathcal{M}_{m,n}$. Alors

$$0 \leq \text{rg}(\mathbb{A}) \leq \min(m, n)$$

et $\text{rg}(\mathbb{A}) = 0$ si et seulement si tous les éléments de $\mathbb{A}$ sont nuls.

EXEMPLE
Soit $\mathbb{A}$ la matrice suivante

$$\mathbb{A} = \begin{pmatrix} 1 & 3 & 2 \\ 1 & 3 & 1 \end{pmatrix}.$$

Le rang de $\mathbb{A}$ est 2 car
- $\mathbb{A}$ est d'ordre 2×3 donc $s \leq \min\{2,3\}$ soit encore $s = 0$, 1 ou 2 ;
- il existe au moins un élément de $\mathbb{A}$ différent de zéro, donc $s \neq 0$ soit encore $s = 1$ ou 2 ; pour qu'il soit 2 il faut trouver une sous-matrice de dimension 2 inversible :
  - comme le déterminant de la sous-matrice composée de la première et de la deuxième colonne est nul, on ne peut pas conclure car je peux encore trouver une autre sous-matrice de dimension 2 inversible ;
  - comme le déterminant de la sous-matrice composée de la première et de la troisième colonne est non nul, alors $s = 2$.

EXEMPLE
Soit $\mathbb{A}$ la matrice suivante

$$\mathbb{A} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 5 & -1 \\ -1 & 0 & -1 \end{pmatrix}.$$

Le rang de $\mathbb{A}$ est 2 car
- $\mathbb{A}$ est d'ordre 3×3 donc $s \leq 3$, *i.e.* $s = 0$, 1, 2 ou 3 ;
- il existe au moins un élément de $\mathbb{A}$ différent de zéro, donc $s \neq 0$;
- le déterminant de $\mathbb{A}$ est 0 (car $L_1 = -L_3$) donc $s \neq 3$;
- le déterminant de la sous-matrice $\left(\begin{smallmatrix} 1 & 0 \\ 0 & 5 \end{smallmatrix}\right)$ est non nul, donc $s = 2$.

1.4 Systèmes linéaires et calcul pratique de la matrice inverse

Soit $n, p \geq 1$ des entiers. Un SYSTÈME LINÉAIRE $n \times p$ est un ensemble de n équations linéaires à p inconnues de la forme

$$(S) \quad \begin{cases} a_{11}x_1 & + & \dots & + & a_{1p}x_p & = & b_1, \\ \vdots & & & & \vdots & & \vdots \\ a_{n1}x_1 & + & \dots & + & a_{np}x_p & = & b_n. \end{cases}$$

- Les COEFFICIENTS a_{ij} et les SECONDES MEMBRES b_i sont des éléments donnés de $\mathbb{K}$.
- Les INCONNUES $x_1, x_2, \dots, x_p$ sont à chercher dans $\mathbb{K}$.
- Une SOLUTION de (S) est un p-uplet $(x_1, x_2, \dots, x_p)$ qui vérifie simultanément les n équations de (S). Résoudre (S) signifie chercher toutes les solutions.
- Un système est IMPOSSIBLE, ou incompatible, s'il n'admet pas de solution.
  Un système est POSSIBLE, ou compatible, s'il admet une ou plusieurs solutions.
- Deux systèmes sont ÉQUIVALENTS s'ils admettent les mêmes solutions.
- Le SYSTÈME HOMOGÈNE associé à (S) est le système obtenu en remplaçant les b_i par 0.
- Un système est CARRÉ si $n = p$.

Si on note

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \qquad \mathbb{A} = \begin{pmatrix} a_{11} & \dots & a_{1p} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{np} \end{pmatrix}$$

le système (S) est équivalent à l'écriture matricielle $\mathbb{A}\mathbf{x} = \mathbf{b}$.
Si on ajoute le vecteur-colonne des seconds membres $\mathbf{b}$ à la matrice des coefficients $\mathbb{A}$, on obtient ce qu'on appelle la matrice augmentée que l'on note $[\mathbb{A}|\mathbf{b}]$.

1.4.1 Système échelonné (ou triangulaire supérieur)

Un système (S) est EN ESCALIER, ou ÉCHELONNÉ, si le nombre de premiers coefficients nuls successifs de chaque équation est strictement croissant. Autrement dit, un système est échelonné si les coefficients non nuls des équations se présentent avec une sorte d'escalier à marches de longueurs variables marquant la séparation entre une zone composée uniquement de zéros et une zone où les lignes situées à droite de l'escalier commencent par des termes non nuls, comme dans l'exemple suivant de 5 équations à 6 inconnues :

$$\begin{cases} 5x_1 - x_2 - x_3 + 2x_4 + x_6 = b_1 \\ 3x_3 - x_4 + 2x_5 = b_2 \\ -x_5 + x_6 = b_3 \\ 5x_6 = b_4 \\ 0 = b_5 \end{cases}$$

La résolution d'un système linéaire $\mathbb{A}\mathbf{x} = \mathbf{b}$ échelonné est simple car, la matrice lui associée étant triangulaire supérieure, on utilise la relation de récurrence (dite *par remontée*)

$$\begin{cases} x_n = \dfrac{b_n}{a_{nn}}, \\ x_i = \dfrac{1}{a_{ii}}\left(b_i - \displaystyle\sum_{j=i+1}^{n} a_{ij}x_j\right), \quad \text{pour } i = n-1, n-2\ldots, 1 \end{cases}$$

EXEMPLE
Résolution du système triangulaire supérieur : $\begin{cases} x_1+x_2+x_3=6, \\ x_2+x_3=5, \\ x_3=3. \end{cases}$

$$\begin{cases} x_3 = \frac{b_3}{a_{33}} = \frac{3}{1}, \\ x_2 = x_i = \frac{1}{a_{22}}(b_2 - a_{23}x_3) = \frac{1}{1}(5 - x_3) = 2 \\ x_1 = x_i = \frac{1}{a_{11}}(b_1 - a_{12}x_2 - a_{13}x_3) = \frac{1}{1}(6 - x_2 - x_3) = 1. \end{cases}$$

Quand un système contient une équation du type

$$0x_1 + \cdots + 0x_p = b,$$

- ⋆ si $b \neq 0$ le système est impossible,
- ⋆ si $b = 0$, on peut supprimer cette équation, ce qui conduit à un système équivalent à (S) dit SYSTÈME RÉDUIT.

Par conséquent, un système échelonné permet d'établir si le système est possible ou impossible comme dans l'exemple suivant.

EXEMPLE
Établir si les trois systèmes linéaires suivantes sont impossibles ou possibles et, dans ce cas, calculer la/les solution(s).

$$(1)\begin{cases} x+y+z=6, \\ y+z=5, \\ z=3. \end{cases} \qquad (2)\begin{cases} x+y+z=6, \\ y+z=5, \\ 0=0. \end{cases} \qquad (3)\begin{cases} x+y+z=6, \\ y+z=5, \\ 0=3. \end{cases}$$

(1) Ce système est possible et admet une et une seule solution : en partant de la dernière ligne et en remontant, on obtient

$$\begin{aligned} z &= 3, \\ y &= 5 - z = 5 - 3 = 2, \\ x &= 6 - y - z = 6 - 2 - 3 = 1. \end{aligned}$$

(2) Ce système est possible et admet une infinité de solutions : en partant de la dernière ligne et en remontant, on obtient

$$z = \kappa \in \mathbb{R},$$

$$y = 5 - \kappa,$$
$$x = 6 - y - z = 1.$$

(3) Le système n'a pas de solution car aucune valeur de z permet de résoudre $0z = 3$.

1.4.2 Systèmes équivalents et opérations élémentaires

Deux systèmes sont équivalents s'ils ont les mêmes solutions. Les opérations suivantes donnent des systèmes équivalents :

- ⋆ remplacer une ligne par elle même ± un multiple d'une autre ligne

$$L_i \leftarrow L_i + \alpha L_j$$

comme par exemple

$$\begin{cases} x+y+ \quad z=6, \\ \quad y+ \quad z=5, \\ \quad y+2z=8, \end{cases} \xrightarrow{L_3 \leftarrow L_3 - L_2} \begin{cases} x+y+z=6, \\ \quad y+z=5, \\ \quad z=3. \end{cases}$$

- ⋆ échanger deux lignes,

$$L_i \leftrightarrow L_j$$

comme par exemple

$$\begin{cases} x+y+z=6, \\ \quad z=3, \\ \quad y+z=5, \end{cases} \xrightarrow{L_2 \leftrightarrow L_3} \begin{cases} x+y+z=6, \\ \quad y+z=5, \\ \quad z=3. \end{cases}$$

Ces transformations sont équivalentes à la multiplication à gauche (pré-multiplication) de la matrice $\mathbb{M} \in \mathscr{M}_{m,n}$ *par la matrice inversible obtenue en appliquant à la matrice identité* $\mathbb{I}_m$ *la transformation correspondante. Par exemple, la transformation qui échange les premières deux lignes de la matrice* $\mathbb{M} \in \mathscr{M}_{4,3}$ *suivante*

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \\ p & q & r \end{pmatrix} \xrightarrow{L_1 \leftrightarrow L_2} \begin{pmatrix} d & e & f \\ a & b & c \\ g & h & i \\ p & q & r \end{pmatrix}$$

équivaut à multiplier $\mathbb{M}$ *à gauche par la matrice obtenue en échangeant les premières deux lignes de la matrice identité* $\mathbb{I}_4$ *:*

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \\ p & q & r \end{pmatrix} = \begin{pmatrix} d & e & f \\ a & b & c \\ g & h & i \\ p & q & r \end{pmatrix}$$

1.4.3 Méthode de Gauss

La méthode de GAUSS transforme un système linéaire quelconque en un système *échelonné* équivalent.
Soit $\mathbb{A} = (a_{ij})_{\substack{1 \le i \le n \\ 1 \le j \le p}}$ la matrice des coefficients du système (S) et $[\mathbb{A}|\mathbf{b}]$ la matrice augmentée.
La méthode de GAUSS comporte $n-1$ étapes : à chaque étape j on fait apparaître des 0 sur la colonne j pour les lignes $i > j$ par des opérations élémentaires sur les lignes.

Étape j : en permutant éventuellement deux lignes de la matrice augmentée (*i.e.* deux équations du système linéaire), on peut supposer $a_{jj} \neq 0$ (appelé pivot de l'étape j). On transforme alors toutes les lignes L_i avec $i > j$ selon la règle :

$$L_i \leftarrow L_i - \frac{a_{ij}}{a_{jj}} L_j,$$

ainsi on fait apparaître des 0 sur la colonne j pour les lignes $i > j$ (*i.e.* on élimine l'inconnue x_j dans chaque lignes L_i du système linéaire).

En réitérant le procédé pour i de 1 à $n-1$, on aboutit à un système échelonné.

EXEMPLE
Soit le système linéaire

$$\begin{cases} x_1 +2x_2+3x_3+4x_4=1, \\ 2x_1+3x_2+4x_3+x_4 =2, \\ 3x_1+4x_2+x_3 +2x_4=3, \\ 4x_1+x_2 +2x_3+3x_4=4. \end{cases}$$

1. Résolution par la méthode du pivot de GAUSS :

$$\begin{cases} x_1+2x_2+3x_3+4x_4=1 \\ 2x_1+3x_2+4x_3 +x_4=2 \\ 3x_1+4x_2 +x_3+2x_4=3 \\ 4x_1 +x_2+2x_3+3x_4=4 \end{cases} \xrightarrow[\substack{L_3\leftarrow L_3-3L_1 \\ L_4\leftarrow L_4-4L_1}]{L_2\leftarrow L_2-2L_1} \text{Étape 1} \begin{cases} x_1+2x_2 +3x_3 +4x_4=1 \\ -x_2 -2x_3 -7x_4=0 \\ -2x_2 -8x_3-10x_4=0 \\ -7x_2-10x_3-13x_4=0 \end{cases}$$

$$\xrightarrow[L_4\leftarrow L_4-7L_2]{L_3\leftarrow L_3-2L_2} \text{Étape 2} \begin{cases} x_1+2x_2+3x_3 +4x_4=1 \\ -x_2-2x_3 -7x_4=0 \\ -4x_3 +4x_4=0 \\ 4x_3+36x_4=0 \end{cases} \xrightarrow{L_4\leftarrow L_4+L_3} \text{Étape 3} \begin{cases} x_1+2x_2+3x_3+4x_4=1 \\ -x_2-2x_3-7x_4=0 \\ -4x_3+4x_4=0 \\ 40x_4=0 \end{cases}$$

donc, en résolvant le système triangulaire supérieur obtenu, on obtient

$$x_4=0, \quad x_3=0, \quad x_2=0, \quad x_1=1.$$

2. Résolution par la méthode du pivot de GAUSS en écriture matricielle :

$$[\mathbb{A}|\mathbf{b}] = \left(\begin{array}{cccc|c} 1&2&3&4&1 \\ 2&3&4&1&2 \\ 3&4&1&2&3 \\ 4&1&2&3&4 \end{array}\right) \xrightarrow[\substack{L_3\leftarrow L_3-3L_1 \\ L_4\leftarrow L_4-4L_1}]{L_2\leftarrow L_2-2L_1} \text{Étape 1} \left(\begin{array}{cccc|c} 1&2&3&4&1 \\ \boxed{0}&-1&-2&-7&0 \\ \boxed{0}&-2&-8&-10&0 \\ \boxed{0}&-7&-10&-13&0 \end{array}\right)$$

$$\xrightarrow[L_4\leftarrow L_4-7L_2]{L_3\leftarrow L_3-2L_2} \text{Étape 2} \left(\begin{array}{cccc|c} 1&2&3&4&1 \\ 0&-1&-2&-7&0 \\ 0&\boxed{0}&-4&4&0 \\ 0&\boxed{0}&4&36&0 \end{array}\right) \xrightarrow{L_4\leftarrow L_4+L_3} \text{Étape 3} \left(\begin{array}{cccc|c} 1&2&3&4&1 \\ 0&-1&-2&-7&0 \\ 0&0&-4&4&0 \\ 0&0&\boxed{0}&40&0 \end{array}\right)$$

donc

$$x_4=0, \quad x_3=0, \quad x_2=0, \quad x_1=1.$$

EXEMPLE (SYSTÈME AVEC DES PARAMÈTRES)
Pour quelles valeurs de a et c le système linéaire suivant admet aucune, une seule ou une infinité de solutions ?

$$\begin{cases} x +5y+z=0, \\ x +6y-z=2, \\ 2x+ay+z=c. \end{cases}$$

Nous avons 3 équations donc il faut effectuer 2 étapes de la méthode de GAUSS :

$$\begin{cases} x +5y+z=0 \\ x +6y-z=2 \\ 2x+ay+z=c \end{cases} \xrightarrow[L_3\leftarrow L_3-2L_1]{L_2\leftarrow L_2-L_1} \text{Étape } j=1 \begin{cases} x+5y +z =0 \\ y -2z=2 \\ (a-10)y-z =c \end{cases} \xrightarrow{L_3\leftarrow L_3-(a-10)L_2} \text{Étape } j=2 \begin{cases} x+5y+z =0 \\ y -2z =2 \\ (2a-21)z=c-2(a-10) \end{cases}$$

Étudions la dernière équation :

$$(2a-21)z=(c-2a+20)$$

- Si $a\neq\frac{21}{2}$ alors $z=\frac{c-2a+20}{2a-21}$ et on trouve y puis x en remontant : il existe une et une seule solution ;
- si $a=\frac{21}{2}$ alors
  - si $c-2a+20=0$ (*i.e.* $c=1$), alors $z=\kappa\in\mathbb{R}$ et on trouve y puis x en remontant : il existe une infinité de solutions ;
  - si $c-2a+20\neq 0$ (*i.e.* $c\neq 1$), alors il n'y a aucune solution.

1.4.4 Variante de Gauss-Jordan

Dans cette variante de la méthode de GAUSS, à chaque étape on fait apparaître des zéros à la fois au-dessus et en-dessous du pivot.

Soit $\mathbb{A} = (a_{ij})_{\substack{1\le i\le n\\ 1\le j\le p}}$ la matrice des coefficients du système (S) et $[\mathbb{A}|\mathbf{b}]$ la matrice augmentée.
La méthode de GAUSS-JORDAN comporte n étapes : à chaque étape j on fait apparaître des 0 sur la colonne j pour les lignes $i \neq j$ par des opérations élémentaires sur les lignes.

Étape j : en permutant éventuellement deux lignes de la matrice augmentée, on peut supposer $a_{jj} \neq 0$. On transforme alors toutes les lignes L_i avec $i \neq j$ selon la règle

$$L_i \leftarrow L_i - \frac{a_{ij}}{a_{jj}} L_j$$

ainsi on fait apparaître des 0 sur la colonne j pour les lignes $i \neq j$ (*i.e.* on élimine l'inconnue x_j dans chaque lignes L_i du système linéaire).

En réitérant le procédé pour i de 1 à n, on aboutit à un système diagonal.

EXEMPLE
Résoudre le système linéaire

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \\ 3 & 4 & 1 & 2 \\ 4 & 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$$

par la méthode de GAUSS-JORDAN.

$$[\mathbb{A}|\mathbf{b}] = \left(\begin{array}{cccc|c} 1 & 2 & 3 & 4 & 1 \\ 2 & 3 & 4 & 1 & 2 \\ 3 & 4 & 1 & 2 & 3 \\ 4 & 1 & 2 & 3 & 4 \end{array}\right) \xrightarrow[\substack{L_3\leftarrow L_3-3L_1\\ L_4\leftarrow L_4-4L_1}]{L_2\leftarrow L_2-2L_1} \text{Étape 1} \left(\begin{array}{cccc|c} 1 & 2 & 3 & 4 & 1 \\ 0 & -1 & -2 & -7 & 0 \\ 0 & -2 & -8 & -10 & 0 \\ 0 & -7 & -10 & -13 & 0 \end{array}\right) \xrightarrow[\substack{L_3\leftarrow L_3-2L_2\\ L_4\leftarrow L_4-7L_2}]{L_1\leftarrow L_1+2L_2} \text{Étape 2} \left(\begin{array}{cccc|c} 1 & 0 & -1 & -10 & 1 \\ 0 & -1 & -2 & -7 & 0 \\ 0 & 0 & -4 & 4 & 0 \\ 0 & 0 & 4 & 36 & 0 \end{array}\right)$$

$$\xrightarrow[\substack{L_2\leftarrow L_2-L_3/2\\ L_4\leftarrow L_4+L_3}]{L_1\leftarrow L_1-L_3/4} \text{Étape 3} \left(\begin{array}{cccc|c} 1 & 0 & 0 & 4 & 1 \\ 0 & -1 & 0 & -7 & 0 \\ 0 & 0 & -4 & 4 & 0 \\ 0 & 0 & 0 & 40 & 0 \end{array}\right) \xrightarrow[\substack{L_2\leftarrow L_2+9L_4/40\\ L_3\leftarrow L_3+4L_4/40}]{L_1\leftarrow L_1+11L_4/40} \text{Étape 4} \left(\begin{array}{cccc|c} 1 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -4 & 0 & 0 \\ 0 & 0 & 0 & 40 & 0 \end{array}\right)$$

donc

$$x_1 = 1, \quad x_2 = 0, \quad x_3 = 0, \quad x_4 = 0.$$

1.4.5 Calcul de la matrice inverse

$\mathbb{A}$ étant inversible, pour obtenir $\mathbb{A}^{-1}$ il suffit de résoudre le système $\mathbb{A}\mathbf{x} = \mathbf{b}$ qui admet pour solution $\mathbf{x} = \mathbb{A}^{-1}\mathbf{b}$.
Pour calculer $\mathbb{A}^{-1}$ il faut alors résoudre n systèmes linéaires de termes sources $(1,0,0,\dots,0)$, $(0,1,0,\dots,0)$, …, $(0,0,0,\dots,1)$.
La méthode suivante résout ces n systèmes linéaires simultanément : on effectue des opérations élémentaires sur la matrice $[\mathbb{A}|\mathbb{I}_n]$ jusqu'à obtenir $[\mathbb{I}_n|\mathbb{A}^{-1}]$:

$$[\mathbb{A}|\mathbb{I}_n] \xrightarrow{\text{Opérations élémentaires}} [\mathbb{I}_n|\mathbb{A}^{-1}].$$

La matrice $\mathbb{A}$ est inversible si et seulement si on obtient par opérations élémentaires sur les lignes de $\mathbb{A}$ une matrice triangulaire sans zéros sur la diagonale ; non inversible si et seulement si on obtient une matrice triangulaire avec un zéro sur la diagonale.

EXEMPLE
Soit $\mathbb{A} = \begin{pmatrix} 2 & 0 \\ 2 & 2 \end{pmatrix}$.

$$[\mathbb{A}|\mathbb{I}_2] = \left(\begin{array}{cc|cc} 2 & 0 & 1 & 0 \\ 2 & 2 & 0 & 1 \end{array}\right) \xrightarrow{L_2\leftarrow L_2-L_1} \left(\begin{array}{cc|cc} 2 & 0 & 1 & 0 \\ 0 & 2 & -1 & 1 \end{array}\right) \xrightarrow[L_2\leftarrow \frac{1}{2}L_2]{L_1\leftarrow \frac{1}{2}L_1} \left(\begin{array}{cc|cc} 1 & 0 & \frac{1}{2} & 0 \\ 0 & 1 & -\frac{1}{2} & \frac{1}{2} \end{array}\right)$$

EXEMPLE
Soit $\mathbb{A} = \begin{pmatrix} 2 & 1 \\ 2 & 2 \end{pmatrix}$.

$$[\mathbb{A}|\mathbb{I}_2] = \left(\begin{array}{cc|cc} 2 & 1 & 1 & 0 \\ 2 & 2 & 0 & 1 \end{array}\right) \xrightarrow{L_2 \leftarrow L_2 - L_1} \left(\begin{array}{cc|cc} 2 & 1 & 1 & 0 \\ 0 & 1 & -1 & 1 \end{array}\right) \xrightarrow{L_1 \leftarrow L_1 - L_2} \left(\begin{array}{cc|cc} 2 & 0 & 2 & -1 \\ 0 & 1 & -1 & 1 \end{array}\right) \xrightarrow{L_1 \leftarrow \frac{1}{2} L_1} \left(\begin{array}{cc|cc} 1 & 0 & 1 & -\frac{1}{2} \\ 0 & 1 & -1 & 1 \end{array}\right)$$

EXEMPLE
Soit $\mathbb{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ avec $\det(\mathbb{A}) = ad - bc \neq 0$.

$$[\mathbb{A}|\mathbb{I}_2] = \left(\begin{array}{cc|cc} a & b & 1 & 0 \\ c & d & 0 & 1 \end{array}\right) \xrightarrow{L_2 \leftarrow L_2 - \frac{c}{a} L_1} \left(\begin{array}{cc|cc} a & b & 1 & 0 \\ 0 & d - \frac{c}{a} b & -\frac{c}{a} & 1 \end{array}\right)$$

$$\xrightarrow{L_1 \leftarrow L_1 - \frac{b}{d - \frac{c}{a} b} L_2} \left(\begin{array}{cc|cc} a & 0 & 1 + \frac{bc}{ad-bc} & -\frac{ab}{ad-bc} \\ 0 & d - \frac{c}{a} b & -\frac{c}{a} & 1 \end{array}\right)$$

$$\xrightarrow[L_2 \leftarrow \frac{1}{d - \frac{c}{a} b} L_2 = \frac{a}{ad-cb} L_2]{L_1 \leftarrow \frac{1}{a} L_1} \left(\begin{array}{cc|cc} 1 & 0 & \frac{1}{a} + \frac{bc}{a(ad-bc)} & -\frac{ab}{ad-bc} \\ 0 & 1 & -\frac{c}{ad-cb} & \frac{a}{ad-cb} \end{array}\right) = \left(\begin{array}{cc|cc} 1 & 0 & \frac{d}{ad-bc} & -\frac{b}{ad-bc} \\ 0 & 1 & -\frac{c}{ad-cb} & \frac{a}{ad-cb} \end{array}\right)$$

EXEMPLE
Calculer l'inverse de la matrice

$$\mathbb{A} = \begin{pmatrix} 1 & 1 & -1 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \end{pmatrix}.$$

$$[\mathbb{A}|\mathbb{I}_3] = \left(\begin{array}{ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 \\ -1 & 1 & 1 & 0 & 1 & 0 \\ 1 & -1 & 1 & 0 & 0 & 1 \end{array}\right) \xrightarrow[L_3 \leftarrow L_3 - L_1]{L_2 \leftarrow L_2 + L_1} \left(\begin{array}{ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 \\ \boxed{0} & 2 & 0 & 1 & 1 & 0 \\ \boxed{0} & -2 & 2 & -1 & 0 & 1 \end{array}\right)$$

$$\xrightarrow{L_2 \leftarrow L_2/2} \left(\begin{array}{ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 \\ 0 & \boxed{1} & 0 & 1/2 & 1/2 & 0 \\ 0 & -2 & 2 & -1 & 0 & 1 \end{array}\right) \xrightarrow[L_3 \leftarrow L_3 + 2L_2]{L_1 \leftarrow L_1 - L_2} \left(\begin{array}{ccc|ccc} 1 & \boxed{0} & -1 & 1/2 & -1/2 & 0 \\ 0 & 1 & 0 & 1/2 & 1/2 & 0 \\ 0 & \boxed{0} & 2 & 0 & 1 & 1 \end{array}\right)$$

$$\xrightarrow{L_3 \leftarrow L_3/2} \left(\begin{array}{ccc|ccc} 1 & 0 & -1 & 1/2 & -1/2 & 0 \\ 0 & 1 & 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & \boxed{1} & 0 & 1/2 & 1/2 \end{array}\right) \xrightarrow{L_1 \leftarrow L_1 + L_3} \left(\begin{array}{ccc|ccc} 1 & 0 & \boxed{0} & 1/2 & 0 & 1/2 \\ 0 & 1 & \boxed{0} & 1/2 & 1/2 & 0 \\ 0 & 0 & 1 & 0 & 1/2 & 1/2 \end{array}\right) = [\mathbb{I}_3|\mathbb{A}^{-1}].$$

1.4.6 Système sur-déterminé

Si le système (S) a n équations et m inconnues avec $n > m$, on dit que le système est sur-déterminé. On considère alors (S') un sous-système carré d'ordre m qu'on peut résoudre par exemple par la méthode du pivot de Gauss. Parmi les solutions de ce système carré, on cherchera celles qui vérifient les équations de (S) qui n'apparaissent pas dans (S').

EXEMPLE
Soit les systèmes linéaires de $n = 3$ équations et $m = 2$ inconnues

$$(S_1) \quad \begin{cases} x + y = 2 \\ x + 2y = 3 \\ x + 3y = 4 \end{cases} \qquad (S_2) \quad \begin{cases} x + y = 2 \\ x + 2y = 3 \\ x + 3y = 0 \end{cases}$$

Prenons comme sous-système carré d'ordre $m = 2$ celui constitué des deux premières équations et résolvons-le :

$$(S') \quad \begin{cases} x + y = 2 \\ x + 2y = 3 \end{cases} \xrightarrow{L_2 \leftarrow L_2 - L_1} \begin{cases} x + y = 2 \\ y = 1 \end{cases}$$

Ce système admet une seule solution : $x = y = 1$.
On vérifie si cette solution satisfait l'équation de (S_1) qui n'apparaît pas dans (S') :

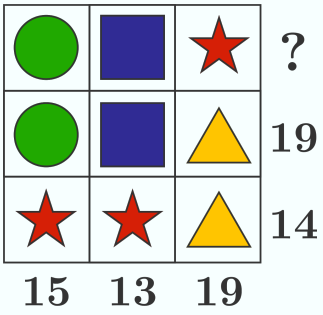
$$x + 3y = 1 + 3 = 4$$

donc $x = y = 1$ est l'unique solution de (S_1).
On vérifie si cette solution satisfait l'équation de (S_2) qui n'apparaît pas dans (S') :

$$x + 3y = 1 + 3 = 4 \neq 0$$

donc (S_2) n'admet pas de solution.

EXEMPLE
Dans ce puzzle, chaque forme correspond à une valeur.



Le numéro à coté de chaque ligne ou colonne représente la somme des valeurs dans cette ligne ou colonne.

Que vaut le point d'interrogation?

On doit calculer $\bullet + \blacksquare + \bigstar$ sachant que

$$\begin{cases} \bullet + \blacksquare + \blacktriangle = 19 \\ 2\bigstar + \blacktriangle = 14 \\ 2\bullet + \bigstar = 15 \\ 2\blacksquare + \bigstar = 13 \\ \bigstar + 2\blacktriangle = 19 \end{cases}$$

Comme au lycée :

- La deuxième et la dernière équations $\begin{cases} 2\bigstar + \blacktriangle = 14 \\ \bigstar + 2\blacktriangle = 19 \end{cases}$ donnent $\begin{cases} \bigstar = 3 \\ \blacktriangle = 8 \end{cases}$
- En injectant ces résultats dans la troisième équation on trouve $\bullet = 6$ et dans la quatrième $\blacksquare = 5$.
- La première équation est "en trop" : le système est sur-déterminé! Puisque $\bullet + \blacksquare + \blacktriangle = 6 + 5 + 8 = 19$, elle est vérifiée et on a bien trouvé l'unique solution de ce système linéaire.
- *Conclusion* : $\bullet + \blacksquare + \bigstar = 14$.

Avec Gauss : Si on écrit le système dans l'ordre suivant,

$$\begin{cases} \blacksquare + \bullet + \blacktriangle = 19 \\ 2\blacksquare + \bigstar = 13 \\ 2\bullet + \bigstar = 15 \\ 2\blacktriangle + \bigstar = 19 \\ \blacktriangle + 2\bigstar = 14 \end{cases}$$

et si l'on considère le sous-système carré obtenu en ne gardant pas la première équation, il suffit d'un seul pas de la méthode de Gauss pour échelonner le système :

$$\begin{cases} 2\blacksquare + \bigstar = 13 \\ 2\bullet + \bigstar = 15 \\ 2\blacktriangle + \bigstar = 19 \\ \blacktriangle + 2\bigstar = 14 \end{cases} \xrightarrow{L_4 \leftarrow L_4 - 2L_3} \begin{cases} 2\blacksquare + \bigstar = 13 \\ 2\bullet + \bigstar = 15 \\ 2\blacktriangle + \bigstar = 19 \\ 3\bigstar = 9 \end{cases}$$

ensuite on résout le système triangulaire obtenu :

$$\bigstar = \frac{9}{3} = 3, \qquad \blacktriangle = \frac{19 - \bigstar}{2} = 8, \qquad \bullet = \frac{15 - \bigstar}{2} = 6, \qquad \blacksquare = \frac{13 - \bigstar}{2} = 5.$$

Il ne reste plus qu'à vérifier que la solution obtenue satisfait l'équation ■ + • + ▲ = 19.

Astucieusement : la somme de toutes les colonne vaut 15 + 13 + 19 = 47 ; la somme de toutes les lignes vaut ? + 19 + 14. Ces deux quantités devant être égale, on obtient ? = 47 − 33 = 14.

Comment envoyer un message secret avec plusieurs espions. Imaginons que l'on désire envoyer un message secret. Par codage, on peut remplacer ce message par un nombre, appelons-le n.

Considérons un polynôme $P(X) = a_k X^k + \cdots + a_1 X + n$ de degré k dont le terme indépendant vaut exactement n. En particulier, on a $P(0) = n$. Un corollaire du théorème fondamental de l'algèbre stipule que le polynôme P est complètement caractérisé par les valeurs qu'il prend en $k+1$ points, par exemple en $X = 1, 2, \ldots, k+1$.

On engage alors au moins $k+1$ espions (mieux en engager un peu plus au cas où certains seraient capturés par les «ennemis»). On donne au i-ème espion le nombre $P(i)$. Les espions se dispersent (par exemple, pour passer les lignes ennemies). Une fois qu'au moins $k+1$ espions sont arrivés à destination, il est aisé de reconstituer le polynôme (on a *un système d'au moins $k+1$ équations linéaires* pour retrouver les $k+1$ coefficients de P) et ainsi retrouver la valeur secrète n.
Si un espion est capturé et qu'il parle, les ennemis auront à leur disposition un des $P(i)$, cela ne leur permet nullement de retrouver n.
De même, si un espion étaient en fait un agent double, connaître $P(i)$ seul ne sert à rien.

Source : http://michelrigo.wordpress.com/2010/01/30/partage-de-secrets-et-tfa/

EXEMPLE

On se propose de calculer n si $k = 2$ et on envoie des espions avec les messages suivants :

- espion 1, message 45
- espion 2, message 50
- espion 3, message 57
- espion 4, message 66

$k = 2$ donc on cherche un polynôme de la forme $p(x) = a + bx + cx^2$ qui satisfait les conditions $p(1) = 45$, $p(2) = 50$, $p(3) = 57$ et $p(4) = 66$. Cela donne le système linéaire

$$\begin{cases} a + b + c = 45 \\ a + 2b + 2^2 c = 50 \\ a + 3b + 3^2 c = 57 \\ a + 4b + 4^2 c = 66 \end{cases}$$

On a 4 équations et 3 inconnues : le système est sur-déterminé.
Négligeons pour le moment la dernière équation et résolvons avec Gauss

$$\begin{cases} a + b + c = 45 \\ a + 2b + 4c = 50 \\ a + 3b + 9c = 57 \end{cases} \xrightarrow[\substack{L_3 \leftarrow L_3 - L_1 \\ \text{Étape } j=1}]{L_2 \leftarrow L_2 - L_1} \begin{cases} a + b + c = 45 \\ b + 3c = 5 \\ 2b + 8c = 12 \end{cases} \xrightarrow[\text{Étape } j=2]{L_3 \leftarrow L_3 - 2L_2} \begin{cases} a + b + c = 45 \\ b + 3c = 5 \\ 2c = 2 \end{cases} \Longrightarrow \begin{array}{c} c = 1 \\ b = 5 - 3c = 2 \\ a = 45 - b - c = 42 \end{array}$$

Vérifions si la dernière équation est bien satisfaite :

$$42 + 4 \times 2 + 4^2 \times 1 = 66.$$

Le message secret est donc $n = a = 42$.

1.4.7 Système sous-déterminé

Un système est sous-déterminé si, après échelonnage, le nombre d'équations significatives est inférieur au nombre d'inconnues.

Équilibrage de réactions chimiques Du point de vue mathématique, équilibrer une réaction chimique signifie trouver des coefficients (dans $\mathbb{N}$ ou $\mathbb{Q}$), appelés coefficients stœchiométriques, qui satisfont certaines contraintes.
Toutes ces contraintes dépendent linéairement des coefficients stœchiométriques, ce qui amène tout naturellement à l'écriture d'un système linéaire.
Typiquement on aura n inconnues mais seulement $n-1$ équations linéairement indépendantes : en effet, les coefficients stœchiométriques ne définissent pas des quantités absolues mais seulement les rapports entre les différents éléments. Par

conséquent, si les coefficients trouvés équilibrent la réaction, alors tous les multiples entiers de ces coefficients équilibrent aussi la réaction.

EXEMPLE
Si on mélange de la soude caustique et de l'acide sulfurique, on obtient du sulfate de sodium et de l'eau :

$$\underbrace{?NaOH}_{\text{soude caustique}} + \underbrace{?H_2SO_4}_{\text{acide sulfurique}} \rightarrow \underbrace{?Na_2SO_4}_{\text{sulfate de sodium}} + ?H_2O$$

Pour que cette réaction ait lieu, il faut que tous les atomes (par exemple de sodium) qui sont à gauche se retrouvent à droite et vice-versa.

$$\underbrace{x\,NaOH}_{\text{soude caustique}} + \underbrace{y\,H_2SO_4}_{\text{acide sulfurique}} \rightarrow \underbrace{z\,Na_2SO_4}_{\text{sulfate de sodium}} + w\,H_2O$$

On voit bien qu'il nous faut au moins 2 molécules de NaOH à gauche pour tomber sur le Na_2 de droite. On pose alors $x = 2$ (mieux, un multiple de 2) et $z = 1$:

$$2NaOH + yH_2SO_4 \rightarrow Na_2SO_4 + wH_2O$$

Le 2OH à gauche venant de la soude et la yH_2 venant de l'acide sulfurique se combinent pour donner wH_2O. On peut alors poser $y = 1$ et $w = 2$:

$$2NaOH + H_2SO_4 \rightarrow Na_2SO_4 + 2H_2O$$

Le SO_4 se trouve bien à gauche et à droite et l'équation est alors équilibrée.
En système cela devient

$$\begin{cases} x = 2z & [Na] \\ x + 2y = 2w & [H] \\ x + 4y = 4z + w & [O] \\ y = z & [S] \end{cases}$$

On trouve $z = y = \kappa$, $x = 2\kappa$ et $w = \kappa$ et l'équation est alors équilibrée. On peut alors poser $\kappa = 1$.

EXEMPLE
Considérons la réaction

$$H_2 + O_2 \rightarrow H_2O.$$

Notons x_1, x_2 et x_3 les coefficients stœchiométriques

$$x_1 H_2 + x_2 O_2 \rightarrow x_3 H_2 O.$$

Les contraintes sont :

- la conservation du nombre d'atomes d'hydrogène : $2x_1 = 2x_3$,
- la conservation du nombre d'atomes d'oxygène : $2x_2 = x_3$.

On note qu'on a 3 inconnues mais seulement 2 équations linéairement indépendantes.
Pour résoudre le problème sans paramètres, fixons arbitrairement un des coefficients, par exemple $x_3 = 1$. On doit alors résoudre le système linéaire

$$\begin{cases} 2x_1 = 2 \\ 2x_2 = 1 \end{cases}$$

On trouve alors $x_1 = 1$ et $x_2 = 1/2$. Si nous voulons des coefficients stœchiométriques entiers, il suffit de multiplier tous les coefficients par 2 et on a ainsi

$$2H_2 + 1O_2 \rightarrow 2H_2O.$$

Astuce
Pour résoudre un système (S) de m équations à n inconnues où $m > n$ on considère un sous-système carré (S') de n équations à n inconnues et on résout ce système :

- si (S') n'admet pas de solution, alors (S) non plus ;
- si (S') admet une unique solution $(c_1, c_2, \ldots, c_n)$, alors on vérifie si cette solution vérifie les autres $m - n$ équations du système (S) :

- ⋆ si oui, alors (S) admet l'unique solution $(c_1, c_2, \ldots, c_n)$,
- ⋆ si non, alors (S) n'admet pas de solution ;

⋆ si (S') admet une infinité de solutions, on cherche parmi ces solutions celles qui vérifient également les autres équations de (S).

EXEMPLE
Considérons le système de 4 équations à 3 inconnues

$$(S) \quad \begin{cases} x+y+z=3, \\ x+2y+3z=6, \\ -x-y+2z=0, \\ 3x+2y-4z=1, \end{cases}$$

Pour résoudre (S), on considère le sous-système carré d'ordre 3

$$(S') \quad \begin{cases} x+y+z=3, \\ x+2y+3z=6, \\ -x-y+2z=0, \end{cases}$$

qu'on peut résoudre par la méthode du pivot de GAUSS

$$\begin{cases} x+y+z=3, \\ x+2y+3z=6, \\ -x-y+2z=0, \end{cases} \xrightarrow[L_3 \leftarrow L_3+L_1]{L_2 \leftarrow L_2-L_1} \begin{cases} x+y+z=3, \\ y+2z=3, \\ 3z=3, \end{cases}$$

Ce sous-système admet l'unique solution $(1,1,1)$. On étudie alors si elle est aussi solution de l'équation de (S) qui n'apparaît pas dans (S') : pour $(x,y,z)=(1,1,1)$ on a $3x+2y-4z=1$ donc le triplet $(1,1,1)$ est solution de (S) et c'est l'unique.

CHAPITRE 2

Introduction à Octave/Matlab

https://nte.mines-albi.fr/MATLAB/fr/co/Matlab_1.html

Nous illustrerons les concepts vu en cours à l'aide de MATLAB (*MATrix LABoratory*), un environnement de programmation et de visualisation. Nous utiliserons aussi GNU Octave (en abrégé Octave) qui est un logiciel libre distribué sous licence GNU GPL. Octave est un interpréteur de haut niveau, compatible la plupart du temps avec MATLAB et possédant la majeure partie de ses fonctionnalités numériques. Dans ce chapitre, nous proposerons une introduction rapide à MATLAB et Octave. Le but de ce chapitre est de fournir suffisamment d'informations pour pouvoir tester les méthodes numériques vues dans ce polycopié. **Il n'est ni un manuel de Octave/Matlab ni une initiation à la programmation.**

Dans ce chapitre

2.1 Les environnements MATLAB et Octave

MATLAB et Octave sont des environnements intégrés pour le Calcul Scientifique et la visualisation. Ils sont écrits principalement en langage C et C++. MATLAB est distribué par la société *The MathWorks* (voir le site `www.mathworks.com`). Son nom vient de *MATrix LABoratory*, car il a été initialement développé pour le calcul matriciel. Octave, aussi connu sous le nom de GNU Octave (voir le site `www.octave.org`), est un logiciel distribué gratuitement. Vous pouvez le redistribuer et/ou le modifier selon les termes de la licence GNU *General Public License* (GPL) publiée par la *Free Software Foundation*.
Il existe des différences entre MATLAB et Octave, au niveau des environnements, des langages de programmation ou des *toolboxes* (collections de fonctions dédiées à un usage spécifique). Cependant, leur niveau de compatibilité est suffisant pour exécuter la plupart des programmes de ce cours indifféremment avec l'un ou l'autre. Quand ce n'est pas le cas – parce que les commandes n'ont pas la même syntaxe, parce qu'elles fonctionnent différemment ou encore parce qu'elles n'existent pas dans l'un des deux programmes – nous l'indiquons et expliquons comment procéder.

Nous utiliserons souvent dans la suite l'expression "commande MATLAB" : dans ce contexte, MATLAB doit être compris comme le langage utilisé par les deux programmes MATLAB et Octave. De même que MATLAB a ses toolboxes, Octave possède un vaste ensemble de fonctions disponibles à travers le projet Octave-forge. Ce dépôt de fonctions ne cesse de s'enrichir dans tous les domaines. Certaines fonctions que nous utilisons dans ce polycopié ne font pas partie du noyau d'Octave, toutefois, elles peuvent être téléchargées sur le site `octave.sourceforge.net`.

2.2 Installation(s) et version(s) en ligne

- La documentation et les sources d'Octave peuvent être téléchargées à l'adresse `https://www.gnu.org/software/octave/`.
  La version en ligne d'Octave est disponible ici `https://octave-online.net/`.
- L'université de Toulon propose aux étudiants la possibilité de le télécharger et de l'installer sur leur poste MATLAB. Toutes les informations sont ici `http://dsiun.univ-tln.fr/MATLAB.html`
  Par ailleurs, la version on line de MATLAB est disponible ici `https://fr.mathworks.com/products/matlab-online.html` Les étudiants et enseignants de l'université de Toulon peuvent s'y connecter avec leurs paramètres universitaires.

Une fois qu'on a installé MATLAB ou Octave, on peut accéder à l'environnement de travail, caractérisé par le symbole d'invite de commande `>>`. Il représente le prompt : cette marque visuelle indique que le logiciel est prêt à lire une commande. Il suffit de saisir à la suite une instruction puis d'appuyer sur la touche «Entrée».

2.3 Premiers pas

Lorsqu'on démarre Octave, une nouvelle fenêtre va s'ouvrir, c'est la fenêtre principale qui contient trois onglets : l'onglet "Fenêtre de commandes", l'onglet "Éditeur" et l'onglet "Documentation".

2.3.1 Fenêtre de commandes : mode interactif

L'onglet "Fenêtre de commandes" permet d'entrer directement des commandes et dès qu'on écrit une commande, Octave l'exécute et renvoie instantanément le résultat. L'invite de commande se compose de deux chevrons (>>) et représente le prompt : cette marque visuelle indique qu'Octave est prêt à lire une commande. Il suffit de saisir à la suite une instruction puis d'appuyer sur la touche «Entrée». La console Octave fonctionne comme une simple calculatrice : on peut saisir une expression dont la valeur est renvoyée dès qu'on presse la touche «Entrée». Voici un exemple de résolution d'un système d'équations linéaires : [1]

```
>> A = [2 1 0; -1 2 2; 0 1 4];
>> b = [1; 2; 3];
>> soln = A\b
soln =
   0.25000
   0.50000
   0.62500
```

Ce mode interactif est très pratique pour rapidement tester des instructions et directement voir leurs résultats. Son utilisation reste néanmoins limitée à des programmes de quelques instructions. En effet, devoir à chaque fois retaper toutes les instructions s'avérera vite pénible.
Si on ferme Octave et qu'on le relance, comment faire en sorte que l'ordinateur se souvienne de ce que nous avons tapé? On ne peut pas sauvegarder directement ce qui se trouve dans la onglet "Fenêtre de commandes", parce que cela comprendrait à la fois les commandes tapées et les réponses du système. Il faut alors avoir préalablement écrit un fichier avec uniquement les commandes qu'on a tapées et l'avoir enregistré sur l'ordinateur avec l'extension `.m`. Une fois cela fait, on demandera à Octave de lire ce fichier et exécuter son contenu, instruction par instruction, comme si on les avait tapées l'une après l'autre dans la Fenêtre de commandes. Ainsi plus tard on pourra ouvrir ce fichier et lancer Octave sans avoir à retaper toutes les commandes. Passons alors à l'onglet "Éditeur".

2.3.2 Éditeur : mode script

On voit qu'il n'y a rien dans cette nouvelle fenêtre (pas d'en-tête comme dans la "Fenêtre de commandes"). Ce qui veut dire que ce fichier est uniquement pour les commandes : Octave n'interviendra pas avec ses réponses lorsque on écrira le

1. Ces instructions calculent la solution du système linéaire $\begin{pmatrix} 2 & 1 & 0 \\ -1 & 2 & 2 \\ 0 & 1 & 4 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$. Noter l'usage des points-virgules à la fin de certaines instructions du fichier : ils permettent d'éviter que les résultats de ces instructions soit affiché à l'écran pendant l'exécution du script.

programme et ce tant que on ne le lui demandera pas. Ayant sauvé le programme dans un fichier avec l'**extension `.m`**, pour le faire tourner et afficher les résultats dans la "Fenêtre de commandes" il suffira d'appuyer sur la touche «F5». Si on a fait une faute de frappe, Octave le remarquera et demandera de corriger.
Maintenant qu'on a sauvé le programme, on est capable de le recharger.
Un fichier de script contient des instructions qui sont **lues et exécutées séquentiellement** par l'interpréteur d'Octave. Ce sont obligatoirement des fichiers au format texte. Copier par exemple les lignes suivantes dans un fichier appelé `first.m`

```
A = [2 1 0; -1 2 2; 0 1 4];
b = [1; 2; 3];
soln = A\b
```

Appuyer sur la touche «F5», cliquer sur "Changer de répertoire" et regarder le résultat dans l'onglet "Fenêtre de commandes". [2]

Remarque
Pensez à placer la commande `clear all` au début de vos scripts, de manière à nettoyer l'environnement de travail (cela effacera toutes les variables en mémoire). Vous pouvez aussi utiliser la commande `clc` pour nettoyer la fenêtre de commandes.
Pour chaque script, on écrira les instructions dans un fichier `mon_script.m` (**sans espaces, ni accents ni points sauf pour l'extension** `.m`). On pourra bien-sûr utiliser le mode interactif pour simplement vérifier une commande mais chaque exercice devra in fine être résolu dans un fichier script. Tous ces scripts devront se trouver dans **un dossier dont le nom ne contient ni espaces, ni accents, ni points**.

2.4 Notions de base

2.4.1 Variables et affectation

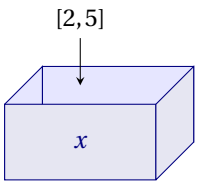
Une variable peut être vue comme une boîte représentant un emplacement en mémoire qui permet de stocker une valeur et à qui on a donné un nom afin de facilement l'identifier (boîte ← valeur) :

```
>> x=1
x =  1
```

```
>> x=[2 5]
x =
   2   5
```

```
>> x='c'
x = c
```

L'affectation `x=[2 5]` crée une association entre le nom `x` et le vecteur $[2,5]$: la boîte de nom `x` contient le vecteur $[2,5]$.



Il faut bien prendre garde au fait que **l'instruction d'affectation** (=) **n'a pas la même signification que le symbole d'égalité** (=) **en mathématiques** (ceci explique pourquoi l'affectation de `1` à `x`, qu'en Octave s'écrit `x = 1`, en algorithmique se note souvent $x \leftarrow 1$).
Une fois une variable initialisée, on peut modifier sa valeur en utilisant de nouveau l'opérateur d'affectation (=). La valeur actuelle de la variable est remplacée par la nouvelle valeur qu'on lui affecte. Dans l'exemple précédent, on initialise une variable à la valeur 1 et on remplace ensuite sa valeur par le vecteur $[1,2]$.
Il est très important de donner un nom clair et précis aux variables. Par exemple, avec des noms bien choisis, on comprend tout de suite ce que calcule le code suivant :

```
base = 8
hauteur = 3
aire = base * hauteur / 2
```

Octave distingue les majuscules des minuscules. Ainsi `mavariable`, `Mavariable` et `MAVARIABLE` sont des variables différentes.
Les noms de variables peuvent être non seulement des lettres, mais aussi des mots; ils peuvent contenir des chiffres (à condition toutefois de ne pas commencer par un chiffre), ainsi que certains caractères spéciaux comme le tiret bas «_» (appelé *underscore* en anglais). Cependant, certains mots sont réservés :

2. Sinon, si ce fichier se trouve dans le répertoire courant d'Octave, pour l'exécuter on peut juste taper son nom **(sans l'extension)** sur la ligne de commande d'Octave : `>> first`
On peut aussi l'exécuter au moyen de la commande `source` qui prend en argument le nom du fichier ou son chemin d'accès (complet ou relatif au répertoire courant). Par exemple : `>> source("Bureau/TP1/first.m")`

`ans`	Nom pour les résultats
`eps`	Le plus petit nombre tel que 1+`eps`>1
`inf`	∞
`NaN`	Not a number
`i` ou `j`	i
`pi`	π

```
>> 5/0
warning: division by zero
ans = Inf
>> 0/0
warning: division by zero
ans = NaN
>> 5*NaN   % Most operations with NaN result in NaN
ans = NaN
>> NaN==NaN   % Different NaN's are not equal!
ans = 0
>> eps
ans =    2.2204e-16
```

Si on écrit une instruction sans affectation, le résultat sera affecté à la variable **`ans`**.

```
>> [4,3]
ans =

   4   3
>> 'Ciao'
ans = Ciao
```

Pour effacer la mémoire et désaffecter toutes les variables, utiliser la fonction **`clear all`**.

2.5 Commentaires

Le symbole `%` indique le début d'un **commentaire** : tous les caractères entre `%` et la fin de la ligne sont ignorés par l'interpréteur.

- ⋆ Dans l'éditeur d'Octave, pour commenter plusieurs lignes en même temps, les sélectionner et appuyer sur les touches «Ctrl+R». Pour dé-commenter plusieurs lignes en même temps, les sélectionner et appuyer sur les touches «Ctrl+Maj+R».
- ⋆ Dans l'éditeur de Matlab, pour commenter plusieurs lignes en même temps, les sélectionner et appuyer sur les touches «Ctrl+R». Pour dé-commenter plusieurs lignes en même temps, les sélectionner et appuyer sur les touches «Ctrl+T».

2.6 Affichage

Lors de l'affectation d'une variable, le résultat de l'affectation sera affiché ; le symbole `;` supprime cet affichage.

```
>> a=[1,2]
a =

   1   2

>> a=[4,3];
>> 'Ciao'
ans = Ciao
```

Pour afficher seulement le **contenu** d'une variable utiliser la fonction **`disp`** (en effet, si on écrit juste le nom de la variable, on affichera aussi le nom de la variable)

```
>> a=[4,3];
>> disp(a)
   4   3
>> a
a =
```

```
   4   3
>> disp('Ciao')
Ciao
```

Pour nettoyer la fenêtre de commandes, utiliser la fonction `clc`.

2.7 Opérations arithmétiques

Dans Octave on a les opérations arithmétiques usuelles :

`+`	Addition
`-`	Soustraction
`*`	Multiplication
`/`	Division
`^`	Exponentiation

Quelques exemples :

```
>> a = 100
a =  100
>> b = 17
b =  17
```

```
>> c = a-b
c =  83
>> a/b
ans =  5.8824
```

```
>> a^b
ans =    1.0000e+34
```

Les opérateurs arithmétiques possèdent chacun une priorité qui définit dans quel ordre les opérations sont effectuées. Par exemple, lorsqu'on écrit `1 + 2 * 3`, la multiplication va se faire avant l'addition. Le calcul qui sera effectué est donc `1 + (2 * 3)`. Dans l'ordre, l'opérateur d'exponentiation est le premier exécuté, viennent ensuite les opérateurs *, /, // et %, et enfin les opérateurs + et -.
Lorsqu'une expression contient plusieurs opérations de même priorité, ils sont évalués de gauche à droite. Ainsi, lorsqu'on écrit `1 - 2 - 3`, le calcul qui sera effectué est `(1 - 2) - 3`. En cas de doutes, vous pouvez toujours utiliser des parenthèses pour rendre explicite l'ordre d'évaluation de vos expressions arithmétiques.
Il existe aussi les opérateurs augmentés (seulement dans Octave) :

`a += b`	équivaut à a = a+b
`a -= b`	équivaut à a = a-b
`a *= b`	équivaut à a = a*b
`a /= b`	équivaut à a = a/b
`a ^ = b`	équivaut à a = a^ b

2.8 Division euclidienne

Lorsqu'on divise un nombre entier D (appelé dividende) par un autre nombre entier d (appelé diviseur), on obtient deux résultats : un quotient q et un reste r, tels que $D = qd + r$ (avec $r < d$). La valeur q est le résultat de la division entière et la valeur r celui du reste de cette division. Par exemple, si on divise 17 par 5, on obtient un quotient de 3 et un reste de 2 puisque $17 = 3 \times 5 + 2$. Ces deux opérateurs sont très utilisés dans plusieurs situations précises. Par exemple, pour déterminer si un nombre entier est pair ou impair, il suffit de regarder le reste de la division entière par deux. Le nombre est pair s'il est nul et est impair s'il vaut 1. Une autre situation où ces opérateurs sont utiles concerne les calculs de temps. Si on a un nombre de secondes et qu'on souhaite le décomposer en minutes et secondes, il suffit de faire la division par 60. Le quotient sera le nombre de minutes et le reste le nombre de secondes restant. Par exemple, 175 secondes correspond à `175//60=2` minutes et `175%60=55` secondes.

```
>> q=fix(9/4)
q =  2
>> % Reste de la division euclidienne de 9 par 4
>> r=rem(9,4)
r =  1
>>
>> r=mod(9,4) % 9 modulo 4
r =  1
>> q=fix(175/60)
q =  2
>> r=rem(175,60)
r =  55
```

2.9 Matrices

Pour définir une matrice on doit écrire ses éléments de la première à la dernière ligne, en utilisant le caractère `;` pour séparer les lignes (ou aller à la ligne). Notons que le symbole `;` a deux fonctions : il supprime l'affichage d'un résultat intermédiaire et il sépare les lignes d'une matrice. Par exemple, la commande

```
>> A = [ 1 2 3; 4 5 6]
```

ou la commande

```
>> A = [ 1 2 3
         4 5 6]
```

donnent

```
A =
   1   2   3
   4   5   6
```

c'est-à-dire, une matrice 2×3 dont les éléments sont indiqués ci-dessus.
Un vecteur colonne est une matrice $1 \times n$, un vecteur ligne est une matrice $n \times 1$:

```
>> b = [1 2 3]
b =
   1   2   3

>> b = [1; 2; 3]
b =
   1
   2
   3
```

L'opérateur **transposition** s'obtient par la commande `'` :

```
>> b = [1 2 3]'
b =
   1
   2
   3
```

- ⋆ Les éléments d'une matrice sont *indexés à partir de* 1.
- ⋆ Pour extraire les éléments d'une matrice on utilise la commande `A(i,j)` où `i` et `j` sont la ligne et la colonne respectivement.
- ⋆ On peut extraire un sous-vecteur en déclarant l'indice i de **début (inclus)** et l'indice j de **fin (inclus)**, séparés par deux-points `v(i:j)`,
- ⋆ ou encore un sous-vecteur en déclarant l'indice i de début (inclus), le pas k et l'indice j de fin (inclus), séparés par des deux-points `v(i:k:j)`.
- ⋆ On peut même utiliser un pas négatif.

Cette opération est connue sous le nom de *slicing* (en anglais).
On peut combiner ces opérations pour extraire des sous-matrices :

```
A(2,3)      % element  A_{23}
A(:,3)      % vecteur colonne [A_{13};...;A_{n3}]
A(1:4,3)    % [A_{13};...A_{43}] premieres 4 lignes du vecteur colonne [A_{13};...A_{n3}]
A(1,:)      % vecteur ligne [A_{11},...,A_{1n}]
A(2,3:end)  % [A_{23},...,A_{2n}] vecteur ligne

diag(A)  % vecteur colonne [A_{11};...;A_{nn}] contenant la diagonale de A
```

Voici des exemples :

```
>> A = [8 1 6; 3 5 7; 4 9 2]
A =
   8   1   6
   3   5   7
   4   9   2

>> A(2,3)   % Element a la ligne 2 colonne 3
ans =  7
```

```
>> A(:,2)   % Toutes les lignes, deuxieme colonne
ans =
   1
   5
   9

>> A(2:3,2:3)   % Sous-matrice 2 x 2
ans =
   5   7
   9   2

>> A(3:-1:1,:)  % les lignes de la derniere a la premiere, toutes les colonnes
ans =
   4   9   2
   3   5   7
   8   1   6
```

ATTENTION
Les indices commencent à 1, ainsi `A(1,:)` indique la première ligne, `A(2,:)` la deuxième etc.

2.9.1 Concaténation de matrices

Nous pouvons générer une matrice par concaténation de deux ou plusieurs autres matrices :
- ⋆ Concaténation horizontale : `A = [B C]` ou `horzcat(A,B)`
- ⋆ Concaténation verticale : `A = [B ; C]` ou `vertcat(A,B)`

Dans le premier cas, on concatène côte à côte (horizontalement) les matrices $\mathbb{B}$ et $\mathbb{C}$. Dans le second, on concatène verticalement les matrices $\mathbb{B}$ et $\mathbb{C}$. Attention aux dimensions qui doivent être cohérentes : dans le premier cas toutes les matrices doivent avoir le même nombre de lignes, et dans le second cas le même nombre de colonnes.
Voici des exemples :

```
>> B = [1 2 3 ; 4 5 6]
B =

   1   2   3
   4   5   6

>> C = [7 8 9; 10 11 12]
C =

    7    8    9
   10   11   12
```

```
>> A = [B C]
A =

    1    2    3    7    8    9
    4    5    6   10   11   12

>> A = [B;C]
A =

    1    2    3
    4    5    6
    7    8    9
   10   11   12
```

2.9.2 Matrices et vecteurs particuliers

Construction de matrices particulières :

① La commande **`zeros`**`(m,n)` construit la matrice rectangulaire nulle $\mathbb{O}$, *i.e.* celle dont tous les éléments a_{ij} sont nuls pour $i = 1,\ldots,m$ et $j = 1,\ldots,n$.
La commande **`zeros`**`(n)` est un raccourci pour **`zeros`**`(n,n)`.

② La commande **`ones`**`(m,n)` construit une matrice rectangulaire dont les éléments a_{ij} sont égaux à 1 pour $i = 1,\ldots,m$ et $j = 1,\ldots,n$.
La commande **`ones`**`(n)` est un raccourci pour **`ones`**`(n,n)`.

③ La commande **`eye`**`(m,n)` renvoie une matrice rectangulaire dont les éléments valent 0 exceptés ceux de la diagonale principale qui valent 1.
La commande **`eye`**`(n)` (qui est un raccourci pour **`eye`**`(n,n)`) renvoie une matrice carrée de dimension n appelée matrice identité et notée $\mathbb{I}$.

④ La commande `A = []` définit une matrice vide.

⑤ La commande `diag(v)` où **v** est un vecteur de n éléments renvoie une matrice carré de taille n dont les éléments valent 0 exceptés ceux de la diagonale principale qui valent **v**.

⑥ Soit **v** un vecteur de n composantes. La commande `diag(v)` renvoie une matrice diagonale carrée de dimension n qui contient **v** sur la diagonale principale ; la commande `diag(v,1)` renvoie une matrice carrée de dimension $n+1$ qui contient **v** sur la sur-diagonale principale etc.

Notons que `diag(ones(1,4))` équivaut à `eye(4)`.

```
>> Z = zeros(2,3)
Z =
   0   0   0
   0   0   0
```

```
>> E = eye(2,5)
E =
Diagonal Matrix
   1   0   0   0   0
   0   1   0   0   0
```

```
>> F = diag(v)
F =
Diagonal Matrix
   1   0   0
   0   2   0
   0   0   3
```

```
>> O = ones(3,2)
O =
   1   1
   1   1
   1   1
```

```
>> A = []
A = [](0x0)
```

```
>> v = [1 2 3]
v =
   1   2   3
```

```
>> G = diag(v,1)
G =
   0   1   0   0
   0   0   2   0
   0   0   0   3
   0   0   0   0
```

Construction de vecteurs particuliers :

① `x = [debut:pas:fin]`

② `x = linspace(debut,fin,N)` (**x** a N points donc le pas h est $\frac{x_{\text{fin}}-x_{\text{debut}}}{N-1} = \frac{x_N-x_1}{N-1}$)

```
x = [-5 : 0.25 : 1]      % x(k)= -5 + 0.25*(k-1), tant que k<=(fin-debut)/N
y = linspace(-5, 1, 25)  % y(k)= -5 + h*(k-1), k=1,2,...,N avec h=(fin-debut)/(N-1)
```

Notons que la première instruction ne garantit pas que le dernier point soit pris, cela dépend du pas choisi (et des erreurs d'arrondis) :

```
x = [0 : 0.4 : 1]     % output : x = 0.00000   0.40000   0.80000
```

Dans la première instructions on peut utiliser un pas négatif :

```
x = [1 : -0.4 : 0]     % output : x = 1.00000   0.60000   0.20000
```

Dimensions de matrices et vecteurs :

```
A = eye(3,4);
[r,c] = size(A) % r = nb de lignes et c=nb de colonnes de A

x = [0:10];
n = length(x) % n = nb d'elements de x
n = numel(x)  % n = nb d'elements de x
```

2.9.3 Opérations entre matrices

Opérations sur les matrices (lorsque les dimensions sont compatibles) :

- ⋆ Somme $\mathbb{C} = \mathbb{A} + \mathbb{B}$, *i.e.* $\mathbb{C}_{ij} = \mathbb{A}_{ij} + \mathbb{B}_{ij}$: `C=A+B`
- ⋆ Produit $\mathbb{C} = \mathbb{A}\mathbb{B}$, *i.e.* $\mathbb{C}_{ij} = \sum_{k=1} \mathbb{A}_{ik} + \mathbb{B}_{kj}$: `C=A*B` NB il s'agit du **produit matriciel**!
- ⋆ Division à droite $\mathbb{C} = \mathbb{A}\mathbb{B}^{-1}$: `C=A/B`
- ⋆ Division à gauche $\mathbb{C} = \mathbb{A}^{-1}\mathbb{B}$: `C=A\B` (si $\mathbb{B}$ est un vecteur colonne alors $\mathbb{C}$ est un vecteur colonne **solution du système linéaire** $\mathbb{A}\mathbb{C} = \mathbb{B}$)
- ⋆ Élévation à la puissance $\mathbb{C} = \mathbb{A}\mathbb{A}\mathbb{A}$: `C=A^3`
- ⋆ Calcul du déterminant (si la matrice est carrée) : `det(A)`
- ⋆ Calcul de la matrice inverse (si la matrice est inversible) : `inv(A)`

```
>> A=[1 2 3; 4 5 6]
A =
   1   2   3
   4   5   6
```

```
>> B=ones(2,3)
B =
   1   1   1
   1   1   1
```

```
>> C=[1 2; 3 4; 5
    6]
C =
   1   2
   3   4
   5   6
```

```
>> D=eye(3,2)
D =
Diagonal Matrix
   1   0
   0   1
   0   0
```

```
>> E=A(1:2,1:2)
E =
   1   2
   4   5
```

```
>> A+B
ans =
   2   3   4
   5   6   7

>> A*C
ans =
   22   28
   49   64

>> A/B
ans =
   1.00000   1.00000
   2.50000   2.50000

>> b=[28;64]
b =
   28
   64

>> A\b
ans =
   2.0000
   4.0000
   6.0000

>> A\B
ans =
  -5.0000e-01  -5.0000e-01  -5.0000e-01
   8.3267e-17   8.3267e-17   8.3267e-17
   5.0000e-01   5.0000e-01   5.0000e-01

>> E^2
ans =
    9   12
   24   33
```

Quand on tente d'effectuer des opérations entre matrices de dimensions incompatibles on obtient un message d'erreur.

```
>> A+C
error: operator +: nonconformant arguments (op1 is 2x3, op2 is 3x2)
```

2.9.4 ♡ Opérations pointées ♡

Quand il s'agit des opérations impliquant des multiplication (donc le produit mais aussi la division et l'élévation à la puissance), la multiplication de deux matrices, avec les notations habituelles, ne signifie pas la multiplication élément par élément mais la multiplication au sens mathématique du produit matriciel. C'est pour cela qu'Octave utilise deux opérateurs distincts pour représenter la multiplication matricielle `*` et la multiplication élément par élément `.*` **Le point placé avant l'opérateur indique que l'opération est effectuée élément par élément.** Les autres opérations de ce type sont la division à droite et l'élévation à la puissance :

- ⋆ Produit $\mathbb{C}_{ij} = \mathbb{A}_{ij}\mathbb{B}_{ij}$: `C=A.*B` NB il s'agit du produit d'Hadamard et non pas du produit matriciel
- ⋆ Division $\mathbb{C}_{ij} = \mathbb{A}_{ij}/\mathbb{B}_{ij}$: `C=A./B`
- ⋆ Élévation à la puissance $\mathbb{C}_{ij} = \mathbb{A}_{ij}^3$: `C=A.^3`

Ces opérations sont à la base de la "**vectorisation**" car elles permettent le **remplacement d'une boucle par une opération matricielle pointée** qui est généralement beaucoup plus performante.

```
>> A=[1 2 3; 4 5 6]
A =
   1   2   3
   4   5   6
>> B=[0 2 0; 0 0 1]
B =
   0   2   0
   0   0   1
>> A.*B
ans =
   0   4   0
   0   0   6
>> A*B
error: operator *: nonconformant arguments (op1 is 2x3, op2 is 2x3)
```

2.10 Fonctions

2.10.1 Fonctions prédéfinies

De très nombreuses fonctions sont déjà disponibles dans Octave/Matlab. Voici quelques exemples de fonction mathématique :

```
abs(-5)

sin(pi)
cos(pi)
tan(pi)

factorial(5) % output : 120

rem(11,3) % output : 2

fix(3.7) % output : 3
fix(3.3) % output : 3
fix(-3.7) % output : -3
fix(-3.3) % output : -3
```

```
round(3.7) % output : 4
round(3.3) % output : 3
round(-3.7) % output : -4
round(-3.3) % output : -3

floor(3.7) % output : 3
floor(3.3) % output : 3
floor(-3.7) % output : -4
floor(-3.3) % output : -4

ceil(3.7) % output : 4
ceil(3.3) % output : 4
ceil(-3.7) % output : -3
ceil(-3.3) % output : -3
```

Remarque (Fonction vectorisée)
Une fonction vectorisée (“universelle” ou «ufunc», abréviation de *universal function*, est le terme exacte) est une fonction qui peut s’appliquer terme à terme aux éléments d’un vecteur ou d’une matrice. Si f est une ufunc et si $a = [a_1, a_2, \ldots, a_n]$ est un tableau, alors $f(a)$ renvoie le tableau $[f(a_1), f(a_2), \ldots, f(a_n)]$. En générale les fonctions prédéfinies sont vectorisées, autrement dit si on applique la fonction à une matrice, elle renvoie une matrice de la même taille en ayant appliqué la fonction à chaque élément.

```
x=[1:5]

sqrt(x)  % output 1.0000    1.4142    1.7321    2.0000    2.2361
exp(x)   % output 2.7183    7.3891    20.0855   54.5982   148.4132
log(x)   % output 0.00000   0.69315   1.09861   1.38629   1.60944
log10(x) % output 0.00000   0.30103   0.47712   0.60206   0.69897
log2(x)  % output 0.00000   1.00000   1.58496   2.00000   2.32193

ismember(2,x)  % output 1
ismember(11,x) % output 0
```

Certaines fonctions sont spécifiques aux matrices :

```
A=[1 2; 3 4];
det(A)
inv(A)
trace(A)
size(A) % dimensions de A
```

```
x=[1 2 3];
length(x) % longueur d'un vecteur
numel(x) % nombre d'elements d'un vecteur
```

2.10.2 Définition d'une fonction

On peut définir nos propres fonctions au moyen de la commande `function` et les utiliser dans plusieurs scripts.

```
function [y1,...,yN] = myfunc(x1,...,xM)
   instruction_1
   instruction_2
   ...
   [y1,...,yN] = ...
end
```

La structure type d'une fonction est la suivante :

- on utilise la commande `function` dans laquelle on indique les arguments `(x1,...,xM)` et la valeur de retour `[y1,...,yN]`
- cette déclaration est suivie du corps de la définition qui est un bloc d'instructions à exécuter et se termine par le mot-clé `end` ou `endfunction`

Quelques conseils :

- Lorsque l'on définit une fonction dans un fichier, il est préférable de mettre un `;` à la fin de chaque commande constituant la fonction. Ainsi, on évitera l'affichage de résultats intermédiaires. Attention cependant à ne pas en mettre sur la première ligne.
- Lorsque l'on définit une fonction, il est préférable d'utiliser systématiquement les opérateurs terme à terme `.*` `./` et `.^` au lieu de `*` `/` et `^` **si l'on veut que cette fonction puisse s'appliquer à des scalaires, mais aussi à des tableaux**.

ATTENTION
Pour éviter de surcharger une fonction déjà définie dans Matlab/Octave, prendre l'habitude d'appeler ses fonctions par `my...`

Voir aussi https://fr.mathworks.com/help/matlab/matlab_prog/create-functions-in-files.html

Fichiers fonctions et fichiers script séparés

Par convention, **chaque définition de fonction est stockée dans un fichier séparé qui porte le nom de la fonction** suivi de l'**extension `.m`** Ces fichiers s'appellent des fichiers de fonction. Notez que c'est la même extension que les fichiers de scripts mais, de plus, **il faut absolument que le fichier s'appelle comme la fonction qu'il contient**.
La structure type d'un fichier de fonction est la suivante :

- toute ligne commençant par un # ou un % est considérée comme un commentaire
- les premières lignes du fichier sont des commentaires qui décrivent la syntaxe de la fonction. Ces lignes seront affichées si on utilise la commande `help myfunc`
- la fonction elle-même.

ATTENTION
Pour éviter toute confusion, utilisez le même nom pour le fichier de fonction et la première fonction du fichier. Matlab/Octave associe votre programme au nom du fichier, pas au nom de la fonction. Les fichiers de script ne peuvent pas avoir le même nom qu'une fonction du fichier.

Voici un exemple qui prend en entrée un vecteur de valeurs et renvoie la moyenne et la déviation standard :

Fichier `stat.m`

```
% Cette fonction calcule la moyenne et la
    deviation
% standard d'un vecteur x
function [m,s] = stat(x)
   n = numel(x);
   m = sum(x)/n;
   s = sqrt(sum((x-m).^2/n)); % NB vectorisation !
end
```

Fichier `test_stat.m`

```
values = [12.7, 45.4, 98.9, 26.6, 53.1];
[average,stdeviation] = stat(values)
```

Remarque (Sous-fonctions)
Un fichier de fonction peut en réalité contenir plusieurs fonctions déclarées au moyen de la commande `function` mais seule la première définition est accessible depuis un script. Les autres définitions concernent des fonctions annexes (on dit parfois des sous-fonctions) qui ne peuvent être utilisées que dans la définition de la fonction principale.

Fonctions locales dans un script - Matlab VS Octave

Les versions récentes de MATLAB permettent la création de fonctions directement dans un script. C'est très pratique pour de petites fonctions "outils", qui ne nécessitent pas d'être externalisées. En ajoutant des fonctions locales vous pouvez **éviter de créer et de gérer des fichiers de fonctions séparés**. Cependant, la syntaxe est différente entre Matlab et Octave.

Matlab, depuis la version R2016b. Les fonctions locales peuvent apparaître dans n'importe quel ordre mais doivent être placées **à la fin du fichier, après le code du script**. Voici un exemple :

```
values = [12.7, 45.4, 98.9, 26.6, 53.1];
[average,stdeviation] = stat(values)

function [m,s] = stat(x)
   n = numel(x);
   m = sum(x)/n;
   s = sqrt(sum((x-m).^2/n));
end
```

Octave. Les fonctions locales peuvent apparaître dans n'importe quel ordre mais doivent être placées **avant le code du script et après l'instruction** `1;` Voici un exemple :

```
1; % Prevent Octave from thinking that this is a function file

function [m,s] = stat(x)
   n = numel(x);
   m = sum(x)/n;
   s = sqrt(sum((x-m).^2/n));
end

values = [12.7, 45.4, 98.9, 26.6, 53.1];
[average,stdeviation] = stat(values)
```

2.10.3 Fonctions anonymes (*lambda functions*)

Les fonctions anonymes permettent de définir une fonction directement dans le script, à condition que la fonction se compose d'une seule instruction. La syntaxe usuelle d'une fonction anonyme est

```
fun = @(arg1, arg2,...,argn) [expr] ; % crochets facultatifs si une seule expression
```

Nous utiliserons les fonctions anonymes surtout pour **écrire directement la fonction dans le fichier de script sans créer un fichier séparé en étant compatibles à la fois avec Matlab et avec Octave.**
Une application courante des fonctions anonymes consiste à définir une expression mathématique.

```
f = @(x) 2*x ; % equivaut a definir f(x)=2x
f(2)           % on evalue f(2) et on obtient 4
```

Cela permet entre autre de calculer rapidement une solution approchée d'une équation :

```
% on veut resoudre x=cos(x)
f = @(x) x.^2-2 ; % on pose f(x)=x^2-2
% fsolve( fct dont on cherche un zero ,  un point pas trop eloigne de la solution )
fsolve( f, 1 ) % ou fzero(f, 1)
fsolve( f,-1 ) % ou fzero(f,-1)
```

La contrainte d'utiliser une seule instruction n'empêche pas de calculer plusieurs résultats (car on peut renvoyer un vecteur) ni d'écrire des boucles (grâce à l'utilisation des instructions pointées) ni des conditions (grâce à l'utilisation de masques, par exemple pour la définition d'une fonction par morceaux, comme on verra à la page 46).

2.10.4 Arrayfun (listes de compréhension)

Les listes de compréhension sont une fonctionnalité puissante de Python qui permet de créer de nouvelles listes en appliquant une certaine expression à chaque élément d'une séquence (comme une liste, un tuple ou une chaîne de caractères). La syntaxe générale est

```
[ f(x) for x in xx ]
```

où $f(x)$ est l'évaluation d'une fonction f en chaque élément x de la séquence `xx`.
De manière analogue, en Matlab/Octave on peut utiliser `arrayfun` qui permet d'appliquer une fonction à chaque élément d'un tableau. La syntaxe générale est

```
arrayfun( f, xx );
```

Exemples :

Python

```
f = lambda x : x**2
xx = [1,3,10]
[ f(x) for x in xx ]
```

Octave/Matlab

```
f = @(x) x^2;
xx = [1,3,10];
arrayfun( f, xx )
```

Bien-sur dans cet exemple simplifié on peut vectoriser directement f :

```
f = @(x) x.^2;
xx = [1,3,10];
f(xx)
```

2.11 Graphes de fonctions $\mathbb{R} \to \mathbb{R}$

Pour tracer le graphe d'une fonction $f\colon [a,b] \to \mathbb{R}$, il faut tout d'abord générer une liste de points x_i où évaluer la fonction f, puis la liste des valeurs $f(x_i)$ et enfin, avec la fonction `plot`, Octave reliera entre eux les points $(x_i, f(x_i))$ par des segments. Plus les points sont nombreux, plus le graphe est proche du graphe de la fonction f.
Pour générer les points x_i on peut utiliser

- ⋆ soit l'instruction `linspace(a,b,n)` qui construit la liste de n éléments

$$[a, a+h, a+2h, \ldots, b = a+nh] \quad \text{avec } h = \frac{b-a}{n-1}$$

```
x = linspace(1,5,5)
x =
   1   2   3   4   5
```

- ⋆ soit l'instruction `[a:h:b]` qui construit la liste

$$[a, a+h, a+2h, \ldots, a+nh]$$

Dans ce cas, attention au dernier terme : b peut ne pas être pris en compte.

Voici un exemple avec une sinusoïde (en utilisant la fonction prédéfinie sin) :

```
x = linspace(-5,5,101); # x = [-5:0.1:5] with 101 elements
y = sin(x); # operation is broadcasted to all elements of the array
plot(x,y)
```

On obtient une courbe sur laquelle on peut zoomer, modifier les marges et sauvegarder dans différents formats.

Si la fonction n'est pas prédéfinie, il est bonne pratique de la définir pour qu'elle opère composante par composante lorsqu'on lui passe un vecteur ou une matrice. Les opérations `/`, $*$ et `\^` agissant sur elle doivent être remplacées par les opérations point correspondantes `./`, `.*` et `\.^` qui opèrent composante par composante.
Par exemple, on se propose de tracer la fonction

$$\begin{aligned} f\colon [-2;2] &\to \mathbb{R} \\ x &\mapsto \frac{1}{1+x^2} \end{aligned}$$

Suivant la façon de définir la fonction, on pourra utiliser l'une des trois méthodes suivantes.

Méthode 1. En utilisant une **fonction anonyme**.
Dans un script ou dans la `prompt` on écrit les instructions suivantes :

```
f = @(x)[1./(1+x.^2)] ; % declaration de la fonction
x = [-2:0.5:2];
y = f(x); % evaluation en plusieurs points
plot(x,y) % affichage des points (x_i,y_i)
```

Méthode 2. En utilisant une **fonction locale**.
Dans un script on écrit les instructions suivantes (attention : syntaxe différente selon qu'on utilise Matlab ou Octave) :

Matlab

```
x = [-2:0.5:2];
y = f(x); % evaluation en plusieurs points
plot(x,y) % affichage des points (x_i,y_i)

function y=f(x)
   y = 1./(1+x.^2);
end
```

Octave

```
1;

function y=f(x)
   y = 1./(1+x.^2);
end

x = [-2:0.5:2];
y = f(x); % evaluation en plusieurs points
plot(x,y) % affichage des points (x_i,y_i)
```

Méthode 3. En utilisant un **fichier fonction**.
On utilise deux fichiers :

3.1. Dans le fichier `f.m` on écrit la fonction informatique suivante

```
function y=f(x)
   y = 1./(1+x.^2);
end
```

3.2. Dans un script ou dans la `prompt` on écrit

```
x = [-2:0.5:2];
y = f(x); % evaluation en plusieurs points
plot(x,y) % affichage des points (x_i,y_i)
```

Canevas

Affichage du graphe de la fonction $f : [a;b] \to \mathbb{R}$ définie par $y = f(x)$ avec $n+1$ points :

```
f = @(x) ... ; % a completer
a = ...;       % a completer
b = ...;       % a completer
n = ...;       % a completer

xx = linspace(a,b,n+1); % n+1 points, n sous-intervalles de largeur (b-a)/n
yy = f(xx); % si f non vectorisee on pourra ecrire yy = arrayfun(f,xx);
plot(xx,yy);
grid();
title("Ma jolie figure");
```

2.11.1 Plusieurs courbes sur le même repère

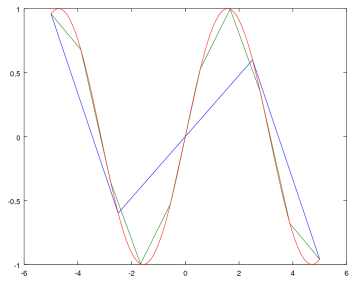
On peut **tracer plusieurs courbes sur le même repère**. Par défaut, MATLAB/Octave efface la figure avant chaque commande de traçage `plot`. Vous pouvez tracer plusieurs lignes soit en écrivant plusieurs courbes dans une même instruction `plot` ou à l'aide de la commande de mise en attente `hold on`. Tant que vous n'utilisez pas de suspension (`hold off`) ou que vous ne fermez pas la fenêtre, tous les tracés apparaissent dans la fenêtre de la figure actuelle.

Par exemple, dans la figure suivante, on a tracé la même fonction : la courbe bleu correspond à la grille la plus grossière, la courbe rouge correspond à la grille la plus fine :

	linestyle=		color=		marker=
-	solid line	r	red	.	points
-	dashed line	g	green	,	pixel
:	dotted line	b	blue	o	filled circles
-.	dash-dot line	c	cyan	v	triangle down
		m	magenta	^	triangle up
		y	yellow	>	triangle right
		w	white	<	triangle left symbols
		k	black	*	star
				+	plus
				s	square
				p	pentagon
				x	x
				X	x filled
				d	thin diamond
				D	diamond

TABLE 2.1 – Quelques options de `plot`

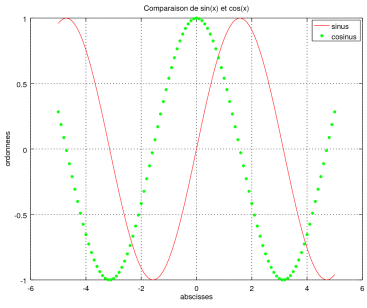
```
a = linspace(-5,5,5); % a = [-5,-3,-1,1,3,5]
fa = sin(a);
b = linspace(-5,5,10); % b = [-5,-4,-3,...,5]
fb = sin(b);
c = linspace(-5,5,101); % c = [-5,-4.9,-4.8,...,5]
fc = sin(c);
plot(a,fa,b,fb,c,fc)
% la derniere ligne peut etre remplacee par
% hold on
% plot(a,fa)
% plot(b,fb)
% plot(c,fc)
% hold off
```



On peut spécifier la couleur et le type de trait, changer les étiquettes des axes, donner un titre, ajouter une grille, une légende etc.

Par exemple, dans le code ci-dessous `"r-"` indique que la première courbe est à tracer en rouge (red) avec un trait continu, et `"g."` que la deuxième est à tracer en vert (green) avec des points.

```
x = linspace(-5,5,101); # x = [-5,-4.9,-4.8,...,5] with
    101 elements
y1 = sin(x); # operation is broadcasted to all elements
    of the array
y2 = cos(x);
plot(x,y1,"r-",x,y2,"g.")
legend(['sinus';'cosinus'])
xlabel('abscisses')
ylabel('ordonnees')
title('Comparaison de sin(x) et cos(x)')
grid
```

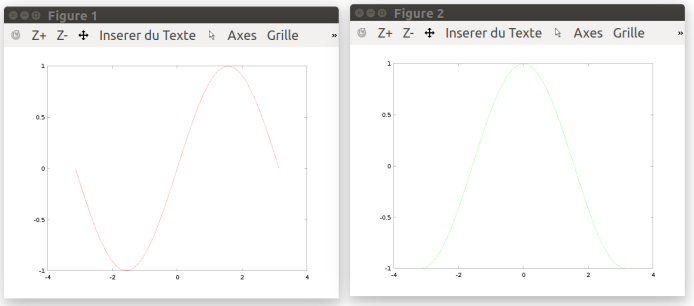


Voir la table 2.1 et la documentation de `Matlab` pour connaître les autres options.

2.11.2 Plusieurs “fenêtres” graphiques

Avec `figure()` on génère une nouvelle fenêtres graphique :

```
x = [-pi:0.05*pi:pi];
figure(1)
plot(x, sin(x), 'r')
figure(2)
plot(x, cos(x), 'g')
```



2.11.3 Plusieurs repères dans la même fenêtre

La fonction `subplot(r,c,z)` subdivise la fenêtre sous forme d'une grille de r lignes et c colonnes. Chaque case est numérotée et `z` est le numéro de la case où afficher le graphe. Les cases sont numérotées de gauche à droite, du haut vers le bas, à partir de l'indice 1. Par exemple, si on écrit `subplot(4,3,3)`, cela signifie que la fenêtre principale a été subdivisée en $4 \times 3 = 12$ cases ; la case qui a pour indice 3 est celle en haut à droite.

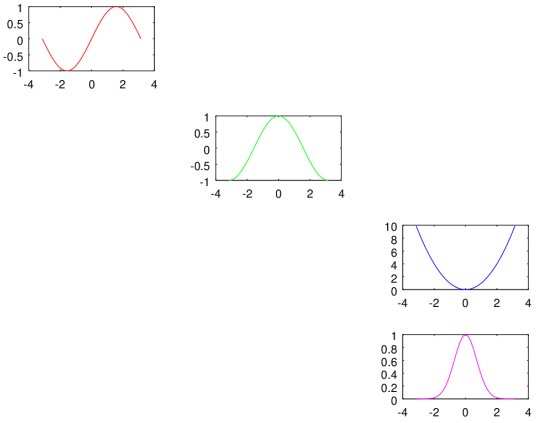
```
x = [-pi:0.05*pi:pi];

subplot(4,3,1)
plot(x, sin(x), 'r')

subplot(4,3,5)
plot(x, cos(x), 'g')

subplot(4,3,9)
plot(x, x.*x, 'b')

subplot(4,3,12)
plot(x, exp(-x.*x), 'm')
```



Exemple

Dans le code suivant on voit comment tracer plusieurs courbes dans un même graphique (avec légende), plusieurs graphe sur une même figure et plusieurs figures.

```
figure(1)
x = [-2:0.5:2];

subplot(2,3,2)
  plot(x,x,'r-',x,exp(x),'b*-')
  legend(['y=x';'y=e^x'])
subplot(2,3,4)
  plot(x,x.^2)
  title('y=x^2')
subplot(2,3,5)
  plot(x,x.^3)
  xlabel('Axe x')
subplot(2,3,6)
  plot(x,sqrt(x))

figure(2)
x = linspace(0.1,exp(2),100);
plot(x,log(x));
```

2.12 Polynômes

Soit $\mathbb{R}_n[x]$ l'ensemble des polynômes de degré inférieur ou égale à n, $n \in \mathbb{N}$. Tout polynôme de cet espace vectoriel s'écrit de manière unique comme

$$p_n(x) = \sum_{i=0}^{n} a_i x^i = a_0 + a_1 x + \cdots + a_n x^n, \qquad \text{où } a_i \in \mathbb{R} \text{ pour } i = 0, \ldots n.$$

Les $n+1$ valeurs réels $a_0, a_1, \ldots, a_n$ sont appelés les **coordonnées de p_n dans la base canonique**[3] de $\mathbb{R}_n[x]$ et on peut les stocker dans un vecteur $\mathbf{p}$:

$$\mathbf{p} = \text{coord}(p_n, \mathscr{C}_n) = (a_n, a_{n-1}, \ldots, a_2, a_1, a_0) \in \mathbb{R}^{n+1}$$

Sous Octave le polynôme $p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 \in \mathbb{R}_n[x]$ est défini par un vecteur $\mathbf{p}$ de dimension $n+1$ contenant les coefficients $\{a_i\}_{i=0,\ldots,n}$ rangés dans l'ordre décroissant des indices, c'est-à-dire que l'on a $p(1) = a_n$, …, $p(n+1) = a_0$. Par exemple, pour construire le polynôme $p(x) = 2 - x + x^2$ nous écrirons

```
p = [1 -1 2]
```

1. La commande `polyval` permet d'évaluer le polynôme p (la fonction polynomiale) en des points donnés. La syntaxe est `polyval(p,x)` où x est une valeur numérique ou un vecteur. Dans le second cas on obtient un vecteur contenant les valeurs de la fonction polynomiale aux différents points spécifiés dans le vecteur $\mathbf{x}$. Par exemple, pour évaluer le polynôme $p(x) = 1 + 2x + 3x^2$ en $\mathbf{x} = (-1, 0, 1, 2)$ nous écrirons

   ```
   p = [3 2 1] % p(x)=1+2x+3x^2
   y = polyval(p,[-1,0,1,2])
   ```

2. Utilisée avec la commande `fplot`, la commande `polyval` permet de tracer le graphe de la fonction polynomiale sur un intervalle `[xmin, xmax]` donné. La syntaxe de l'instruction est `('polyval([a_n,...,a_0],x)', [x_min, x_max])`. Par exemple, pour tracer le graphe du polynôme $p(x) = 1 + 2x + 3x^2$ sur l'intervalle $[-2; 2]$ nous écrirons

   ```
   fplot('polyval([3 2 1],x)',[-2,2])
   ```

3. La commande `roots` calcule les racines du polynôme dans $\mathbb{C}$. La syntaxe est `roots(p)`. Par exemple, pour calculer les racines du polynôme $p(x) = 1 - x^2$ nous écrirons

   ```
   p = [-1 0 1] % p(x) = -x^2+1
   racines = roots(p)
   ```

4. La commande `poly` définit un polynôme à partir de ses racines $r_0, r_1, \ldots r_n$ comme suit : $p(x) = \prod_{i=0}^{n}(x - r_i)$. La syntaxe est `poly(r)` où $\mathbf{r}$ est un vecteur contenant ses racines. Par exemple, pour définir le polynôme $p(x) = (x-1)(x+1)$ nous écrirons

   ```
   poly([1 -1])
   ```

5. Somme de deux polynômes : si les deux polynômes n'ont pas même degré, il faut ajouter des zéros en début du polynôme de plus petit degré afin de pouvoir calculer l'addition des deux vecteurs représentatifs. Par exemple,

   ```
   p = [1 2 3 4] % p(x) =  4 + 3x + 2x^2 +x^3
   q = [0 4 5 6] % q(x) =  6 + 5x + 4x^2 (+0x^3)
   s = p+q       % s(x) = 10 + 8x + 6x^2 +x^3
   ```

6. La commande `conv` permet de calculer le polynôme u produit de deux polynômes p et q. La syntaxe est `conv(p,q)`. Par exemple, pour calculer $u(x) = p(x)q(x)$ avec $p(x) = 4 + 3x + 2x^2 + x^3$ et $q(x) = 6 + 5x + 4x^2$ nous écrirons

   ```
   p = [1 2 3 4] % p(x) = 4+3x+3x^2+x^3
   q = [4 5 6]   % q(x) = 6+5x+4x^2
   u = conv(p,q) % u(x) = 24+38x+43x^2+28x^3+13x^4+4x^5
   ```

7. La commande `deconv` permet de calculer les polynômes q et r quotient et reste de la division du polynôme u par le polynôme p. La syntaxe est `conv(u,p)`. Par exemple, pour calculer q et r tel que $u(x) = q(x)p(x) + r(x)$ avec $u(x) = 24 + 38x + 43x^2 + 28x^3 + 13x^4 + 4x^5$ et $p(x) = 4 + 3x + 2x^2 + x^3$ nous écrirons

   ```
   u = [4 13 28 43 38 25] % u(x)=25+38x+43x^2+28x^3+13x^4+4x^5
   p = [1 2 3 4]          % p(x)=4+3x+3x^2+x^3
   [q,r] = deconv(u,p)
   ```

3. La base canonique de l'espace vectoriel $\mathbb{R}_n[x]$ est l'ensemble $\mathscr{C}_n = \{1, x, x^2, \ldots, x^n\}$

8. La commande `polyder` permet de calculer le polynôme d dérivée d'un polynôme p. La syntaxe est `polyder(p)`. Par exemple, pour calculer $p'(x)$ avec $p(x) = 1+2x+3x^2$ nous écrirons

```
p = [3 2 1]  % p(x) =1+2x+3x^2
polyder(p) % p'(x)=2+6x
```

9. La commande `polyint` permet de calculer le polynôme $\int_0^x p(t)\,\mathrm{d}t$ qui s'annule en 0 et qui est une primitive d'un polynôme p. La syntaxe est `polyint(p)`. Par exemple, pour calculer $\int_0^x p(t)\,\mathrm{d}t$ avec $p(x) = 1+x^2$ nous écrirons

```
p = [1 0 1]           % p(x)=1+x^2
integral = polyint(p) % int(p,0..x)=x+x^3/3 donc integral = [1/3 0 1 0]
```

10. Utilisée avec la commande `polyval`, la commande `polyint` permet de calculer l'intégrale d'un polynôme sur un intervalle `[a,b]` donné. Par exemple, pour calculer $\int_0^3 p(t)\,\mathrm{d}t$ avec $p(x) = 1+x^2$ nous écrirons

```
area = polyval(integral,3)-polyval(integral,0)  % area=3+27/3-0=12
```

11. La commande `polyfit` permet de calculer le polynôme de $\mathbb{R}_m[x]$ de meilleure approximation au sens des moindres carrés d'un ensemble de points. La syntaxe est `polyfit(xx,yy,m)` où `xx` et `yy` sont deux vecteurs de n composantes et m le degré du polynôme cherché. Si $m = n$ on obtient le polynôme d'interpolation. Par exemple, pour calculer l'équation de la droite de meilleur approximation de l'ensemble $\{(0,0.1),(1,0.9),(2,2)\}$ nous écrirons :

```
xx = [0 1 2]
yy = [0.1 0.9 2]
polyfit(xx,yy,1)
```

Pour calculer le polynôme d'interpolation du même ensemble on pose $m = 2$ et on écrit :

```
polyfit(xx,yy,2)
```

12. Pour afficher le polynôme de façon naturelle il faut utiliser la fonction `polyout`. Par exemple,

```
p = [3 2 1]
polyout(p,'x')
```

2.13 Opérateurs de comparaison et connecteurs logiques

Les opérateurs de comparaison renvoient `1` si la condition est vérifiée, `0` sinon. Ces opérateurs sont

On écrit	<	>	<=	>=	==	~=
Ça signifie	$<$	$>$	$\leq$	$\geq$	$=$	$\neq$

Bien distinguer l'instruction d'affectation = du symbole de comparaison ==.

ATTENTION

Les opérateurs de comparaison agissent élément par élément, ainsi lorsqu'on les applique à une matrice le résultat est une matrice qui contient que des `0` ou `1` (parfois appelée "**masque**"). Par exemple

```
>> A = [1 2 3; 4 -5 6]; B = [7 8 9; 0 1 2];
>> A>B
ans =
   0   0   0
   1   0   1
```

On peut utiliser un masque pour remplacer seulement les éléments qui satisfont une conditions :

```
>> A = [1 -2 3; 4 -5 6];
>> A(A<0) = -100
A =
     1  -100     3
     4  -100     6
```

Les masques sont à la base de la "vectorisation" car permettent le remplacement d'un boucle avec des conditions par une opération matricielle qui est généralement beaucoup plus performante.

EXEMPLE (MASQUE)
Étant donné trois vecteurs :
- ⋆ `hotels` : une liste de noms d'hôtels
- ⋆ `ratings` : leurs notes dans une ville
- ⋆ `cutoff` : la note minimale

on souhaite afficher les noms des hôtels ayant une note supérieure ou égale au seuil.

```
>> hotels = ["CityLights"; "SeaView"; "MarketPlace"; "ResortSpa"; "Nightingale"; "Clubadub"; "
    SkylineView"; "MarinaBay"; "ComfortFirst"; "VillageValley"]; % vecteur colonne
>> ratings = [7.2; 8.7; 6.5; 9.3; 4.3; 6.9; 8.8; 5.9; 7.4; 9.1]; % vecteur colonne
>> cutoff = 8;
>> good = hotels(ratings>=cutoff,:) % NB ":" pour selectionner toute la chaine de caracteres
good =

SeaView
ResortSpa
SkylineView
VillageValley
```

Pour combiner des conditions complexes (par exemple $x > -2$ et $x^2 < 5$), on peut combiner les opérateurs de comparaison avec les connecteurs logiques :

On écrit	&	|	~
Ça signifie	et	ou	non

Par exemple

```
>> A = [1 2 3; 4 -5 6]; B = [7 8 9; 0 1 2];
>> (A > B) | (B > 5)
ans =
   1   1   1
   1   0   1
```

Le & peut être remplacé par la multiplication pointée :

```
>> A = [1 2 3; 4 -5 6]; B = [7 8 9; 0 1 2];
>> (A > B) & (B <5)
ans =
  0  0  0
  1  0  1
>>  (A > B) .* (B <5)
ans =
   0   0   0
   1   0   1
```

2.14 Structures itératives

Les structures de répétition se classent en deux catégories : les *répétitions inconditionnelles* pour lesquelles le bloc d'instructions est à répéter un nombre donné de fois et les *répétitions conditionnelles* pour lesquelles le bloc d'instructions est à répéter autant de fois qu'une condition est vérifiée.

2.14.1 Répétition `for`

Lorsque l'on souhaite répéter un bloc d'instructions un nombre déterminé de fois, on peut utiliser un *compteur actif*, c'est-à-dire une variable qui compte le nombre de répétitions et conditionne la sortie de la boucle.
La syntaxe de la commande `for` est schématiquement

```
for var = expression
   instruction_1
   instruction_2
end
```

`expression` peut être un vecteur ou une matrice. Par exemple, le code suivant calcule les 12 premières valeurs de la suite de Fibonacci définie par la relation de récurrence $u_n = u_{n-1} + u_{n-2}$ avec pour valeurs initiales $u_1 = u_2 = 1$:

```
n = 12;
u = ones(1, n); # allocation
for i = 3:n
   u(i) = u(i-1)+u(i-2);
end
disp(u)
```

```
n = 12;
u = [1,1];
for i = 3:n
   u = [ u , u(end)+u(end-1) ]; # concatenation
end
disp(u)
```

Dans les deux cas le résultat affiché est

```
1     1     2     3     5     8    13    21    34    55    89   144
```

Dans l'exemple suivant on calcul la somme des n premiers entiers (et on vérifie qu'on a bien $n(n+1)/2$) :

```
n = 100;
s = 0;
for i=1:n
   s += i;
end
s
n*(n+1)/2
```

Bien sur, dans ce cas il est préférable d'écrire

```
sum(1:100)
```

Il est possible d'imbriquer des boucles, c'est-à-dire que dans le bloc d'une boucle, on utilise une nouvelle boucle.

```
for x = [10,20,30,40,50] % for x = 10:10:50
   for y=[3,7]
      disp(x+y)
   end
end
```

Dans ce petit programme x vaut d'abord 10, y prend la valeur 3 puis la valeur 7 (le programme affiche donc d'abord 13, puis 17). Ensuite $x = 20$ et y vaut de nouveau 3 puis 7 (le programme affiche donc ensuite 23, puis 27).
Voir aussi https://fr.mathworks.com/help/matlab/ref/for.html

2.14.2 Boucle `while` : répétition conditionnelle

`While` est la traduction de "tant que…". Concrètement, la boucle s'exécutera tant qu'une condition est remplie (donc tant qu'elle renverra la valeur 1). Le constructeur `while` a la forme générale suivante :

```
i = ...
while condition(i)
   instruction_1
   instruction_2
   i = ...
end
```

où `condition` représente des ensembles d'instructions dont la valeur est `1` ou `0`. Tant que la condition `condition` a la valeur `1`, on exécute les instructions `instruction_i`.

ATTENTION
Si la condition ne devient jamais fausse, le bloc d'instructions est répété indéfiniment et le programme ne se termine pas.

Voici un exemple pour créer la liste $\left[1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}\right]$:

```
nMax = 4;
n = 1;
a = [];
while n<=nMax
   a=[a,1/n]; # Append element to list
   n += 1; % si Matlab: n=n+1
end
disp(a)
```

Dans l'exemple suivant on calcul la somme des n premiers entiers tant que la somme ne dépasse pas 100 :

```
s = 0;
n = 0;
while s<100
   n += 1; % si Matlab: n=n+1
   s += n; % si Matlab: s=s+n
end
disp(n-1)
disp(s-n)
% En effet avec le dernier n on depasse 100
```

2.15 Vectorisation, *i.e.* optimisation des performances

La plupart du temps on manipule des vecteurs et des matrices. Les opérateurs et les fonctions élémentaires sont conçus pour favoriser ce type de manipulation et, de manière plus générale, pour permettre la vectorisation des programmes. Certes, le langage Octave contient des instructions conditionnelles, des boucles et la programmation récursive, mais la vectorisation permet de limiter le recours à ces fonctionnalités qui ne sont jamais très efficaces dans le cas d'un langage interprété. Les surcoûts d'interprétation peuvent être très pénalisants par rapport à ce que ferait un programme C ou FORTRAN compilé lorsque l'on effectue des calculs numériques. Il faut donc veiller à réduire autant que possible le travail d'interprétation en vectorisant les programmes.

Dans les exemples suivant, la fonction `tic` s'utilise avec la fonction `toc` pour mesurer le temps écoulé. La fonction `tic` enregistre l'heure actuelle et la fonction `toc` utilise la valeur enregistrée pour calculer le temps écoulé.

EXEMPLE
Quasiment toutes les fonctions prédéfinies sont vectorisées. Voici un exemple avec la fonction sin.

Le code

```
n = 10000;
xx = linspace(0,2*pi,n);
yy = zeros(numel(xx));
tic
for i = 1:n
   yy(i) = sin(xx(i));
end;
toc
```

est significativement plus lent que

```
n = 10000;
xx = linspace(0,2*pi,n);
tic
yy = sin(xx);
toc
```

EXEMPLE
Pour calculer $\sum_{n=1}^{10000} \frac{1}{n^2}$, on peut utiliser les trois codes suivants, le deuxième étant significativement plus rapide :

```
n = 1:10^7;

tic
s = 0;
for i = n,
   s += 1/i^2; % si Matlab: s =
       s+1/i^2;
end;
s
toc
```

```
n = 1:10^7;

tic
s = sum(1./n.^2)
toc
```

```
n = 1:10^7;

tic
s = (1./n)*(1./n)'
toc
```

Un exemple de sorties obtenues :
- ⋆ avec le premier script : `Elapsed time is 12.7138 seconds.`
- ⋆ avec le deuxième script : `Elapsed time is 0.116321 seconds.`
- ⋆ avec le troisième script : `Elapsed time is 0.098047 seconds.`

2.16 Structure conditionnelle

Supposons vouloir calculer la valeur $y = f(x)$ d'un nombre x selon la règle suivante :

$$f(x) = \begin{cases} x & \text{si } x \leq -5, \\ 100 & \text{si } -5 < x \leq 0, \\ x^2 & \text{si } 0 < x < 10, \\ x-2 & \text{sinon.} \end{cases}$$

On a besoin d'une instruction qui opère une disjonction de cas. En Octave il s'agit de l'instruction de choix introduite par le mot-clé `if`. La syntaxe complète est la suivante :

```
if condition_1
   instruction_1.1
   instruction_1.2
   ...
elseif condition_2
   instruction_2.1
   instruction_2.2
   ...
...
else
   instruction_n.1
   instruction_n.2
   ...
end
```

où `condition_1`, `condition_2`... représentent des ensembles d'instructions dont la valeur est `1` ou `0` (on les obtient en général en utilisant les opérateurs de comparaison). La première condition `condition_i` ayant la valeur `1` entraîne l'exécution des instructions `instruction_i.1`, `instruction_i.2`... Si toutes les conditions sont `0`, les instructions `instruction_n.1`, `instruction_n.2`... sont exécutées. Les blocs `elseif` et `else` sont optionnels.

Pour l'exemple donné, la fonction (non vectorisée) peut s'écrire comme suit :

```
function y = f_scal(x)
   if x<=-5
      y = x;
   elseif x<=0 % inutile d'ecrire (x>-5)&(x<=0) car le cas x>-5 a deja ete considere
      y = 100;
   elseif x<10
      y = x^2;
   else
      y = x-2;
   end
end
```

On peut ensuite définir la fonction vectorisée avec une boucle explicite ou avec `arrayfun` :

```
function yy = f1(xx)
   yy = [];
   for x = xx
      yy = [yy, f_scal(x)];
   end
end
```

```
function yy = f1(xx)
   yy = arrayfun(f_scal,xx)
end
```

La même fonction peut aussi s'écrire comme suit grâce aux masques :

```
f2 = @(x) [  x.*(x<=-5) + 100.*(x>-5).*(x<=0) + (x.^2).*(x>0).*(x<10) + (x-2).*(x>=10) ];
```

Pour vérifier qu'on a bien la même fonction on compare le graphe des deux fonctions :

```
xx = [-7:0.1:12];
yy1 = f1(xx);
yy2 = f2(xx);
plot(xx,yy1,'r',xx,yy2,'b.')
```

Voici un exemple pour établir si un nombre est compris entre deux valeurs :

```
x = 10;
minVal = 2;
maxVal = 6;

if (x >= minVal) && (x <= maxVal) % equivaut a (x >= minVal) .* (x <= maxVal)
    disp('Value within specified range.')
elseif (x > maxVal)
    disp('Value exceeds maximum value.')
else
    disp('Value is below minimum value.')
end
```

Voir aussi https://fr.mathworks.com/help/matlab/ref/if.html

EXEMPLE

Étant donné deux matrices d'entrée $\mathbb{A}$ et $\mathbb{B}$, vérifier si on peut calculer le produit $\mathbb{AB}$. Si c'est le cas, créez une matrice $\mathbb{C}$ qui contient le produit $\mathbb{AB}$, sinon, $\mathbb{C}$ doit contenir une chaîne de caractère contenant un message d'erreur.

```
function C = in_prod(A,B)
  [rA,cA]=size(A);
  [rB,cB]=size(B);
  if cA==rB
      C = A*B;
  else
      C = "Have you checked the inner dimensions?"
  end
end

# TESTS
C = in_prod([1 2],[2;3])
C = in_prod(-5,100)
C = in_prod([1 2;3 4],[5;6])
C = in_prod([1 2 3; 4 5 6],[2 5;3 6])
```

MathWorks®

MATLAB® Basic Functions Reference

MATLAB Environment

`clc`	Clear command window
`help fun`	Display in-line help for `fun`
`doc fun`	Open documentation for `fun`
`load("filename","vars")`	Load variables from `.mat` file
`uiimport("filename")`	Open interactive import tool
`save("filename","vars")`	Save variables to file
`clear item`	Remove items from workspace
`examplescript`	Run the script file named `examplescript`
`format style`	Set output display format
`ver`	Get list of installed toolboxes
`tic, toc`	Start and stop timer
`Ctrl+C`	Abort the current calculation

Operators and Special Characters

`+, -, *, /`	Matrix math operations
`.*, ./`	Array multiplication and division (element-wise operations)
`^, .^`	Matrix and array power
`\`	Left division or linear optimization
`.', '`	Normal and complex conjugate transpose
`==, ~=, <, >, <=, >=`	Relational operators
`&&, ||, ~, xor`	Logical operations (AND, NOT, OR, XOR)
`;`	Suppress output display
`...`	Connect lines (with break)
`% Description`	Comment
`'Hello'`	Definition of a character vector
`"This is a string"`	Definition of a string
`str1 + str2`	Append strings

Special Variables and Constants

`ans`	Most recent answer
`pi`	π=3.141592654...
`i, j, 1i, 1j`	Imaginary unit
`NaN, nan`	Not a number (i.e., division by zero)
`Inf, inf`	Infinity
`eps`	Floating-point relative accuracy

Defining and Changing Array Variables

`a = 5`	Define variable a with value 5

`A = [1 2 3; 4 5 6]`
`A = [1 2 3`
`4 5 6]`
	Define A as a 2x3 matrix
	"space" separates columns
	";" or new line separates rows

`[A,B]`	Concatenate arrays horizontally
`[A;B]`	Concatenate arrays vertically
`x(4) = 7`	Change 4th element of x to 7
`A(1,3) = 5`	Change A(1,3) to 5
`x(5:10)`	Get 5th to 10th elements of x
`x(1:2:end)`	Get every 2nd element of x (1st to last)
`x(x>6)`	List elements greater than 6
`x(x==10)=1`	Change elements using condition
`A(4,:)`	Get 4th row of A
`A(:,3)`	Get 3rd column of A
`A(6, 2:5)`	Get 2nd to 5th element in 6th row of A
`A(:,[1 7])=A(:,[7 1])`	Swap the 1st and 7th column
`a:b`	[a, a+1, a+2, ..., a+n] with a+n≤b
`a:ds:b`	Create regularly spaced vector with spacing ds
`linspace(a,b,n)`	Create vector of n equally spaced values
`logspace(a,b,n)`	Create vector of n logarithmically spaced values
`zeros(m,n)`	Create m x n matrix of zeros
`ones(m,n)`	Create m x n matrix of ones
`eye(n)`	Create a n x n identity matrix
`A=diag(x)`	Create diagonal matrix from vector
`x=diag(A)`	Get diagonal elements of matrix
`meshgrid(x,y)`	Create 2D and 3D grids
`rand(m,n), randi`	Create uniformly distributed random numbers or integers
`randn(m,n)`	Create normally distributed random numbers

Complex Numbers

`i, j, 1i, 1j`	Imaginary unit
`real(z)`	Real part of complex number
`imag(z)`	Imaginary part of complex number
`angle(z)`	Phase angle in radians
`conj(z)`	Element-wise complex conjugate
`isreal(z)`	Determine whether array is real

MathWorks®

Elementary Functions

`sin(x), asin`	Sine and inverse (argument in radians)
`sind(x), asind`	Sine and inverse (argument in degrees)
`sinh(x), asinh`	Hyperbolic sine and inverse (arg. in radians)
Analogous for the other trigonometric functions: `cos`, `tan`, `csc`, `sec`, and `cot`	
`abs(x)`	Absolute value of x, complex magnitude
`exp(x)`	Exponential of x
`sqrt(x), nthroot(x,n)`	Square root, real nth root of real numbers
`log(x)`	Natural logarithm of x
`log2(x), log10`	Logarithm with base 2 and 10, respectively
`factorial(n)`	Factorial of n
`sign(x)`	Sign of x
`mod(x,d)`	Remainder after division (modulo)
`ceil(x), fix, floor`	Round toward +inf, 0, -inf
`round(x)`	Round to nearest decimal or integer

Tables

`table(var1,...,varN)`	Create table from data in variables var1, ..., varN
`readtable("file")`	Create table from file
`array2table(A)`	Convert numeric array to table
`T.var`	Extract data from variable var
`T(rows,columns)`, `T(rows,["col1","coln"])`	Create a new table with specified rows and columns from T
`T.varname=data`	Assign data to (new) column in T
`T.Properties`	Access properties of T
`categorical(A)`	Create a categorical array
`summary(T), groupsummary`	Print summary of table
`join(T1, T2)`	Join tables with common variables

Tasks (Live Editor)

Live Editor tasks are apps that can be added to a live script to interactively perform a specific set of operations. Tasks represent a series of MATLAB commands. To see the commands that the task runs, show the generated code.

Common tasks available from the Live Editor tab on the desktop toolstrip:

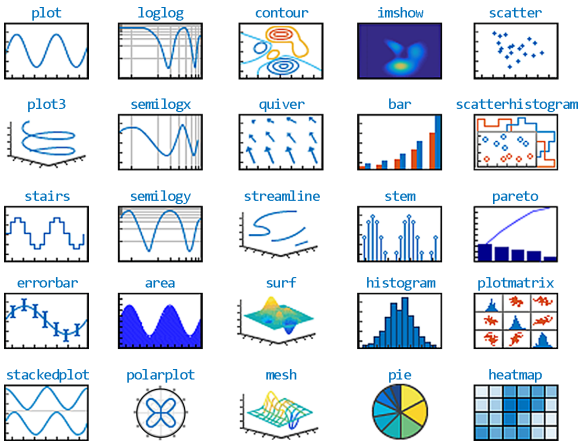
- Clean Missing Data
- Clean Outlier
- Find Change Points
- Find Local Extrema
- Remove Trends
- Smooth Data

Plotting

`plot(x,y,LineSpec)`
Line styles:
`-, --, :, -.`
Markers:
`+, o, *, ., x, s, d`
Colors:
`r, g, b, c, m, y, k, w`
	Plot y vs. x
	(`LineSpec` is optional)
	`LineSpec` is a combination of `linestyle`, `marker`, and `color` as a string.
	Example: `"-r"`
	= red solid line without markers

`title("Title")`	Add plot title
`legend("1st", "2nd")`	Add legend to axes
`x/y/zlabel("label")`	Add x/y/z axis label
`x/y/zticks(ticksvec)`	Get or set x/y/z axis ticks
`x/y/zticklabels(labels)`	Get or set x/y/z axis tick labels
`x/y/ztickangle(angle)`	Rotate x/y/z axis tick labels
`x/y/zlim`	Get or set x/y/z axis range
`axis(lim), axis style`	Set axis limits and style
`text(x,y,"txt")`	Add text
`grid on/off`	Show axis grid
`hold on/off`	Retain the current plot when adding new plots
`subplot(m,n,p)`, `tiledlayout(m,n)`	Create axes in tiled positions
`yyaxis left/right`	Create second y-axis
`figure`	Create figure window
`gcf, gca`	Get current figure, get current axis
`clf`	Clear current figure
`close all`	Close open figures

Common Plot Types



Plot Gallery: *mathworks.com/products/matlab/plot-gallery*

MathWorks®

Programming Methods

Functions

```
% Save your function in a function file or at the end
% of a script file. Function files must have the
% same name as the 1st function
function cavg = cumavg(x) %multiple args. possible
    cavg=cumsum(vec)./(1:length(vec));
end
```

Anonymous Functions

```
% defined via function handles
fun = @(x) cos(x.^2)./abs(3*x);
```

Control Structures

if, elseif Conditions

```
if n<10
    disp("n smaller 10")
elseif n<=20
    disp("n between 10 and 20")
else
    disp("n larger than 20")
```

Switch Case

```
n = input("Enter an integer: ");
switch n
    case -1
        disp("negative one")
    case {0,1,2,3} % check four cases together
        disp("integer between 0 and 3")
    otherwise
        disp("integer value outside interval [-1,3]")
end % control structures terminate with end
```

For-Loop

```
% loop a specific number of times, and keep
% track of each iteration with an incrementing
% index variable
for i = 1:3
    disp("cool");
end % control structures terminate with end
```

While-Loop

```
% loops as long as a condition remains true
n = 1;
nFactorial = 1;
while nFactorial < 1e100
    n = n + 1;
    nFactorial = nFactorial * n;
end % control structures terminate with end
```

Further programming/control commands

`break`	Terminate execution of for- or while-loop
`continue`	Pass control to the next iteration of a loop
`try, catch`	Execute statements and catch errors

Numerical Methods

`fzero(fun,x0)`	Root of nonlinear function
`fminsearch(fun,x0)`	Find minimum of function
`fminbnd(fun,x1,x2)`	Find minimum of fun in [x1, x2]
`fft(x), ifft(x)`	Fast Fourier transform and its inverse

Integration and Differentiation

`integral(f,a,b)`	Numerical integration (analogous functions for 2D and 3D)
`trapz(x,y)`	Trapezoidal numerical integration
`diff(X)`	Differences and approximate derivatives
`gradient(X)`	Numerical gradient
`curl(X,Y,Z,U,V,W)`	Curl and angular velocity
`divergence(X,..,W)`	Compute divergence of vector field
`ode45(ode,tspan,y0)`	Solve system of nonstiff ODEs
`ode15s(ode,tspan,y0)`	Solve system of stiff ODEs
`deval(sol,x)`	Evaluate solution of differential equation
`pdepe(m,pde,ic,... bc,xm,ts)`	Solve 1D partial differential equation
`pdeval(m,xmesh,... usol,xq)`	Interpolate numeric PDE solution

Interpolation and Polynomials

`interp1(x,v,xq)`	1D interpolation (analogous for 2D and 3D)
`pchip(x,v,xq)`	Piecewise cubic Hermite polynomial interpolation
`spline(x,v,xq)`	Cubic spline data interpolation
`ppval(pp,xq)`	Evaluate piecewise polynomial
`mkpp(breaks,coeffs)`	Make piecewise polynomial
`unmkpp(pp)`	Extract piecewise polynomial details
`poly(x)`	Polynomial with specified roots x
`polyeig(A0,A1,...,Ap)`	Eigenvalues for polynomial eigenvalue problem
`polyfit(x,y,d)`	Polynomial curve fitting
`residue(b,a)`	Partial fraction expansion/decomposition
`roots(p)`	Polynomial roots
`polyval(p,x)`	Evaluate poly p at points x
`polyint(p,k)`	Polynomial integration
`polyder(p)`	Polynomial differentiation

MathWorks®

Accelerating the pace of engineering and science

Matrices and Arrays

`length(A)`	Length of largest array dimension
`size(A)`	Array dimensions
`numel(A)`	Number of elements in array
`sort(A)`	Sort array elements
`sortrows(A)`	Sort rows of array or table
`flip(A)`	Flip order of elements in array
`squeeze(A)`	Remove dimensions of length 1
`reshape(A,sz)`	Reshape array
`repmat(A,n)`	Repeat copies of array
`any(A), all`	Check if any/all elements are nonzero
`nnz(A)`	Number of nonzero array elements
`find(A)`	Indices and values of nonzero elements

Linear Algebra

`rank(A)`	Rank of matrix
`trace(A)`	Sum of diagonal elements of matrix
`det(A)`	Determinant of matrix
`poly(A)`	Characteristic polynomial of matrix
`eig(A), eigs`	Eigenvalues and vectors of matrix (subset)
`inv(A), pinv`	Inverse and pseudo inverse of matrix
`norm(x)`	Norm of vector or matrix
`expm(A), logm`	Matrix exponential and logarithm
`cross(A,B)`	Cross product
`dot(A,B)`	Dot product
`kron(A,B)`	Kronecker tensor product
`null(A)`	Null space of matrix
`orth(A)`	Orthonormal basis for matrix range
`tril(A), triu`	Lower and upper triangular part of matrix
`linsolve(A,B)`	Solve linear system of the form AX=B
`lsqminnorm(A,B)`	Least-squares solution to linear equation
`qr(A), lu, chol`	Matrix decompositions
`svd(A)`	Singular value decomposition
`gsvd(A,B)`	Generalized SVD
`rref(A)`	Reduced row echelon form of matrix

Descriptive Statistics

`sum(A), prod`	Sum or product (along columns)
`max(A), min, bounds`	Largest and smallest element
`mean(A), median, mode`	Statistical operations
`std(A), var`	Standard deviation and variance
`movsum(A,n), movprod, movmax, movmin, movmean, movmedian, movstd, movvar`	Moving statistical functions n = length of moving window
`cumsum(A), cumprod, cummax, cummin`	Cumulative statistical functions
`smoothdata(A)`	Smooth noisy data
`histcounts(X)`	Calculate histogram bin counts
`corrcoef(A), cov`	Correlation coefficients, covariance
`xcorr(x,y), xcov`	Cross-correlation, cross-covariance
`normalize(A)`	Normalize data
`detrend(x)`	Remove polynomial trend
`isoutlier(A)`	Find outliers in data

Symbolic Math*

`sym x, syms x y z`	Declare symbolic variable
`eqn = y == 2*a + b`	Define a symbolic equation
`solve(eqns,vars)`	Solve symbolic expression for variable
`subs(expr,var,val)`	Substitute variable in expression
`expand(expr)`	Expand symbolic expression
`factor(expr)`	Factorize symbolic expression
`simplify(expr)`	Simplify symbolic expression
`assume(var,assumption)`	Make assumption for variable
`assumptions(z)`	Show assumptions for symbolic object
`fplot(expr), fcontour, fsurf, fmesh, fimplicit`	Plotting functions for symbolic expressions
`diff(expr,var,n)`	Differentiate symbolic expression
`dsolve(deqn,cond)`	Solve differential equation symbolically
`int(expr,var,[a, b])`	Integrate symbolic expression
`taylor(fun,var,z0)`	Taylor expansion of function

*requires Symbolic Math Toolbox

mathworks.com/help/matlab

CHAPITRE 3

Exercices

Dans ce chapitre

3.1 Calcul matriciel

Exercice 3.1 (Écriture matricielle)
On considère les matrices $\mathbb{A} = (a_{ij})$, $\mathbb{B} = (b_{ij})$ et $\mathbb{C} = (c_{ij})$ carrées d'ordre 4 définies par $a_{ij} = i^2$, $b_{ij} = i + j$, $c_{ij} = \min\{i; j\}$. Écrire ces matrices sous la forme de tableaux de nombres.

Correction

$$\mathbb{A} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 4 & 4 & 4 & 4 \\ 9 & 9 & 9 & 9 \\ 16 & 16 & 16 & 16 \end{pmatrix}$$

```
A = zeros(4);
for i=1:4
   A(i,:)=i^2;
end
A
```

$$\mathbb{B} = \begin{pmatrix} 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \\ 5 & 6 & 7 & 8 \end{pmatrix}$$

```
B = zeros(4);
for i=1:4
   for j=1:4
      B(i,j)=i+j;
   end
end
B
```

$$\mathbb{C} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 2 & 3 & 3 \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

```
C = zeros(4);
for i=1:4
   for j=1:4
      C(i,j)=min(i,j);
   end
end
C
```

Exercice 3.2
Soient les matrices

$$\mathbb{A} = \begin{pmatrix} -3 & 2 \\ 0 & 4 \\ 1 & -1 \end{pmatrix} \quad \text{et} \quad \mathbb{B} = \begin{pmatrix} 1 & 2 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}.$$

1. Trouver une matrice $\mathbb{C}$ telle que $\mathbb{A} - 2\mathbb{B} - \mathbb{C} = \mathbb{O}$.
2. Trouver une matrice $\mathbb{D}$ telle que $\mathbb{A} + \mathbb{B} + \mathbb{C} - 4\mathbb{D} = \mathbb{O}$.

Correction

1. On cherche $\mathbb{C}$ telle que $\mathbb{C} = \mathbb{A} - 2\mathbb{B}$, *i.e.*

$$\begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \end{pmatrix} = \begin{pmatrix} -3 & 2 \\ 0 & 4 \\ 1 & -1 \end{pmatrix} - 2\begin{pmatrix} 1 & 2 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} -3-2\times 1 & 2-2\times 2 \\ 0-2\times 0 & 4-2\times 1 \\ 1-2\times 1 & -1-2\times 1 \end{pmatrix} = \begin{pmatrix} -5 & -2 \\ 0 & 2 \\ -1 & -3 \end{pmatrix}.$$

   $\mathbb{A} - 2\mathbb{B} - \mathbb{C} = \mathbb{O}$.

2. On cherche $\mathbb{D}$ telle que $\mathbb{D} = \frac{1}{4}(\mathbb{A} + \mathbb{B} + \mathbb{C}) = \frac{1}{4}(\mathbb{A} + \mathbb{B} + \mathbb{A} - 2\mathbb{B}) = \frac{1}{2}\mathbb{A} - \frac{1}{4}\mathbb{B}$, *i.e.*

$$\begin{pmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \\ d_{31} & d_{32} \end{pmatrix} = \frac{1}{2}\begin{pmatrix} -3 & 2 \\ 0 & 4 \\ 1 & -1 \end{pmatrix} - \frac{1}{4}\begin{pmatrix} 1 & 2 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\times(-3) - \frac{1}{4}\times 1 & \frac{1}{2}\times 2 - \frac{1}{4}\times 2 \\ \frac{1}{2}\times 0 - \frac{1}{4}\times 0 & \frac{1}{2}\times 4 - \frac{1}{4}\times 1 \\ \frac{1}{2}\times 1 - \frac{1}{4}\times 1 & \frac{1}{2}\times(-1) - \frac{1}{4}\times 1 \end{pmatrix} = \begin{pmatrix} -7/4 & 1/2 \\ 0 & 7/4 \\ 1/4 & -3/4 \end{pmatrix}.$$

```
A = [-3 2; 0 4; 1 -1]
B = [1 2; 0 1; 1 1]
C = A-2*B
D = 1/4*(A+B+C)
```

Exercice 3.3

Effectuer les multiplications suivantes

$$\begin{pmatrix} 3 & 1 & 5 \\ 2 & 7 & 0 \end{pmatrix}\begin{pmatrix} 2 & 1 & -1 & 0 \\ 3 & 0 & 1 & 8 \\ 0 & -5 & 3 & 4 \end{pmatrix}, \qquad \begin{pmatrix} -3 & 0 & 5 \end{pmatrix}\begin{pmatrix} 2 \\ -4 \\ -3 \end{pmatrix}, \qquad \begin{pmatrix} -3 \\ 0 \\ 5 \end{pmatrix}\begin{pmatrix} 2 & -4 & -3 \end{pmatrix}.$$

Correction

$$\overbrace{\begin{pmatrix} 3 & 1 & 5 \\ 2 & 7 & 0 \end{pmatrix}}^{2\times 3}\overbrace{\begin{pmatrix} 2 & 1 & -1 & 0 \\ 3 & 0 & 1 & 8 \\ 0 & -5 & 3 & 4 \end{pmatrix}}^{3\times 4} = \overbrace{\begin{pmatrix} 3\times 2+1\times 3+5\times 0 & 3\times 1+1\times 0+5\times(-5) & 3\times(-1)+1\times 1+5\times 3 & 3\times 0+1\times 8+5\times 4 \\ 2\times 2+7\times 3+0\times 0 & 2\times 1+7\times 0+0\times(-5) & 2\times(-1)+7\times 1+0\times 3 & 2\times 0+7\times 8+0\times 4 \end{pmatrix}}^{2\times 4}$$

$$= \begin{pmatrix} 9 & -22 & 13 & 28 \\ 25 & 2 & 5 & 56 \end{pmatrix}$$

$$\overbrace{\begin{pmatrix} -3 & 0 & 5 \end{pmatrix}}^{1\times 3}\overbrace{\begin{pmatrix} 2 \\ -4 \\ -3 \end{pmatrix}}^{3\times 1} = \overbrace{\begin{pmatrix} -3\times 2+0\times(-4)+5\times(-3) \end{pmatrix}}^{1\times 1} = -21$$

$$\overbrace{\begin{pmatrix} -3 \\ 0 \\ 5 \end{pmatrix}}^{3\times 1}\overbrace{\begin{pmatrix} 2 & -4 & -3 \end{pmatrix}}^{1\times 3} = \overbrace{\begin{pmatrix} -3\times 2 & -3\times(-4) & -3\times(-3) \\ 0\times 2 & 0\times(-4) & 0\times(-3) \\ 5\times 2 & 5\times(-4) & 5\times(-3) \end{pmatrix}}^{3\times 3} = \begin{pmatrix} -6 & 12 & 9 \\ 0 & 0 & 0 \\ 10 & -20 & -15 \end{pmatrix}$$

```
[3 1 5; 2 7 0]*[2 1 -1 0; 3 0 1 8; 0 -5 3 4]
[-3 0 5]*[2 -4 -3]' % ce qui equivaut a [-3 0 5]*[2; -4; -3]
[-3 0 5]'*[2 -4 -3] % ce qui equivaut a [-3; 0; 5]*[2 -4 -3]
```

Exercice 3.4

Soit les matrices

$$\mathbb{A} = \begin{pmatrix} 1 & 2 & 3 \\ -1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \qquad \mathbb{B} = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \qquad \mathbf{v} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

1. Calculer tous les produits possibles à partir de $\mathbb{A}$, $\mathbb{B}$ et $\mathbf{v}$.

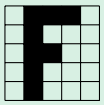
2. Calculer $(\mathbb{A}-\mathbb{I})^7$ et en extraire le coefficient en position $(2,3)$.
3. Calculer $\mathbb{A}^{-1}$ et la trace de $\mathbb{A}^{-1}$ (*i.e.* la somme des coefficients sur la diagonale).

Correction

```
A = [1 2 3; -1 0 1; 0 1 0] % 3x3
B = [2 -1 0; -1 0 1]       % 2x3
v = [1;0;1]                % 3x1
% Les produits possibles sont
B*A % (2x3)*(3x3) -> 2x3
A*v % (3x3)*(3x1) -> 3x1
B*v % (2x3)*(3x1) -> 2x1
%
Id = eye(3)
D = (A-Id)^7 % c'est bien ^7 (produit matriciel) et non .^7
D(2,3)     % -> 153
%
invA = A^(-1) % ou inv(A)
sum(diag(invA))
```

Exercice 3.5 (Multiplication matricielle appliquée)

On modélise une image en noir et blanc formée de 25 pixels par une matrice de 5 lignes et 5 colonnes, dans laquelle 0 correspond à un pixel blanc et 1 à un pixel noir. L'image à modéliser est la suivante :

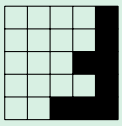


1. Donner la matrice $\mathbb{M}$ associée à l'image.
2. Soient

$$\mathbb{A}=\begin{pmatrix}0&0&0&0&1\\0&0&0&1&0\\0&0&1&0&0\\0&1&0&0&0\\1&0&0&0&0\end{pmatrix} \qquad \mathbb{B}=\begin{pmatrix}0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\\0&0&0&0&1\\1&0&0&0&0\end{pmatrix}$$

   Calculer le produit $\mathbb{AM}$. Quel est l'effet de la matrice $\mathbb{A}$ sur l'image ? Quel est l'effet si on fait le produit $\mathbb{MA}$ sur l'image ?

   Calculer le produit $\mathbb{BM}$. Quel est l'effet de la matrice $\mathbb{B}$ sur l'image ? Quel est l'effet si on fait le produit $\mathbb{MB}$ sur l'image ?
3. Quelle image obtient-on en faisant le produit $\mathbb{AMB}$?
4. Quelle image obtient-on en faisant le produit $\mathbb{AMA}$?
5. Quel produit matriciel peut-on faire pour obtenir la figure suivante ?



6. Calculer le produit $\mathbb{AM}^T$.

Correction

1.

$$\mathbb{M}=\begin{pmatrix}0&1&1&1&0\\0&1&0&0&0\\0&1&1&0&0\\0&1&0&0&0\\0&1&0&0&0\end{pmatrix}$$

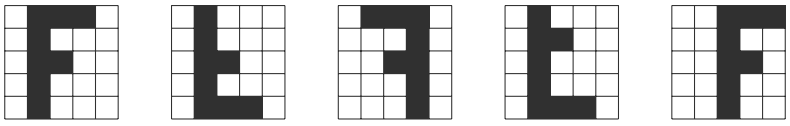
2. Le produit $\mathbb{AM}$ correspond à inverser l'ordre des lignes de la matrice $\mathbb{M}$: l'image est alors symétrique par rapport à la troisième ligne.

Le produit $\mathbb{MA}$ correspond à inverser l'ordre des colonnes de la matrice $\mathbb{M}$: l'image est alors symétrique par rapport à la troisième colonne.

Le produit $\mathbb{BM}$ correspond à translater les lignes de la matrice $\mathbb{M}$ d'un rang vers le haut (la première ligne passant en cinquième ligne) : l'image est alors translatée d'un rang vers le haut (la première ligne passant en cinquième ligne).

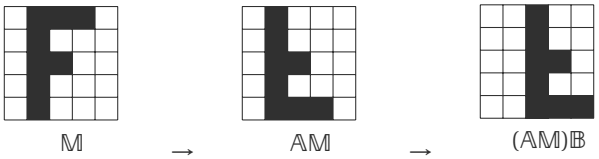
Le produit $\mathbb{MB}$ correspond à translater les colonnes de la matrice $\mathbb{M}$ d'un rang vers la droite (la cinquième colonne passant en première colonne) : l'image est alors translatée d'un rang vers la droite (la cinquième colonne passant en première colonne).

$$\mathbb{AM} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix} \quad \mathbb{MA} = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \mathbb{BM} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix} \quad \mathbb{MB} = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

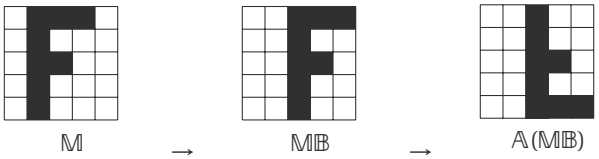


$\mathbb{M}$ $\mathbb{AM}$ $\mathbb{MA}$ $\mathbb{BM}$ $\mathbb{MB}$

3. Le produit $\mathbb{AMB} = (\mathbb{AM})\mathbb{B}$ correspond par exemple à une symétrie horizontale par rapport à la troisième ligne suivie d'une translation d'un rang vers la droite. On peut aussi l'écrire comme $\mathbb{AMB} = \mathbb{A}(\mathbb{MB})$ qui correspond à une translation d'un rang vers la droite suivie d'une symétrie horizontale par rapport à la troisième ligne. Dans tous les cas on obtient l'image suivante :



$\mathbb{M} \rightarrow \mathbb{AM} \rightarrow (\mathbb{AM})\mathbb{B}$

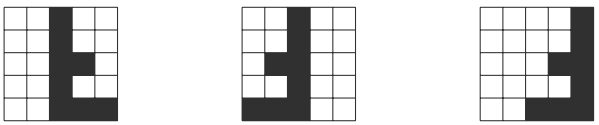


$\mathbb{M} \rightarrow \mathbb{MB} \rightarrow \mathbb{A}(\mathbb{MB})$

4. On peut par exemple calculer $\mathbb{AMAB}$.

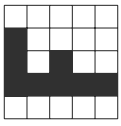
$\mathbb{M} \rightarrow \mathbb{AM} \rightarrow (\mathbb{AM})\mathbb{A} \rightarrow (\mathbb{AMA})\mathbb{B}$

Ou encore calculer $\mathbb{AMBABB}$.



$\mathbb{AMB} \rightarrow \mathbb{AMBA} \rightarrow \mathbb{AMBABB}$

5. Le produit $\mathbb{AM}^T$ correspond à une rotation antihoraire, elle équivaut à l'instruction `rot90(A)`.



$\mathbb{AM}^T$

```
M = [0 1 1 1 0
     0 1 0 0 0
     0 1 1 0 0
     0 1 0 0 0
     0 1 0 0 0]
```

```
A = eye(5);
A = A(:,5:-1:1)
%
B = diag(ones(4,1),1);
B(5,1) = 1;
B
```

```
A*M
M*A
B*M
M*B
A*M*B
A*M*A*B
A*M*B*A*B*B
```

Exercice 3.6
Calculer a, b, c et d tels que

$$\text{①} \begin{pmatrix}1 & 3\\ 2 & 8\end{pmatrix}\begin{pmatrix}a & b\\ c & d\end{pmatrix} = \mathbb{I}_2, \qquad \text{②} \begin{pmatrix}a & b\\ c & d\end{pmatrix}\begin{pmatrix}1 & 3\\ 2 & 8\end{pmatrix} = \mathbb{I}_2.$$

Que peut-on conclure?

Correction
Comme

$$\begin{pmatrix}1 & 3\\ 2 & 8\end{pmatrix} \times \begin{pmatrix}a & b\\ c & d\end{pmatrix} = \begin{pmatrix}a+3c & b+3d\\ 2a+8c & 2b+8d\end{pmatrix}$$

il faut que

$$\begin{cases} a+3c=1,\\ b+3d=0,\\ 2a+8c=0,\\ 2b+8d=1, \end{cases} \iff \begin{pmatrix}a & b\\ c & d\end{pmatrix} = \begin{pmatrix}4 & -3/2\\ -1 & 1/2\end{pmatrix}.$$

De la même manière, pour avoir

$$\begin{pmatrix}a & b\\ c & d\end{pmatrix}\begin{pmatrix}1 & 3\\ 2 & 8\end{pmatrix} = \begin{pmatrix}a+2b & 3a+8b\\ c+2d & 3c+8d\end{pmatrix}$$

il faut que

$$\begin{cases} a+2b=1,\\ 3a+8b=0,\\ c+2d=0,\\ 3c+8d=1, \end{cases} \iff \begin{pmatrix}a & b\\ c & d\end{pmatrix} = \begin{pmatrix}4 & -3/2\\ -1 & 1/2\end{pmatrix}.$$

```
A = [1 3; 2 8]
I2 = eye(2)
I2/A
A\I2
```

On conclut que

$$\begin{pmatrix}a & b\\ c & d\end{pmatrix} = \begin{pmatrix}1 & 3\\ 2 & 8\end{pmatrix}^{-1} = \begin{pmatrix}4 & -3/2\\ -1 & 1/2\end{pmatrix}.$$

```
inv([1 3; 2 8])
```

Exercice 3.7 (`warning: matrix singular to machine precision`)
Calculer a, b, c et d tels que

$$\text{①} \begin{pmatrix}1 & 2\\ 2 & 4\end{pmatrix}\begin{pmatrix}a & b\\ c & d\end{pmatrix} = \mathbb{I}_2, \qquad \text{②} \begin{pmatrix}a & b\\ c & d\end{pmatrix}\begin{pmatrix}1 & 2\\ 2 & 4\end{pmatrix} = \mathbb{I}_2.$$

Que peut-on conclure?

Correction
Comme

$$\begin{pmatrix}1 & 2\\ 2 & 4\end{pmatrix} \times \begin{pmatrix}a & b\\ c & d\end{pmatrix} = \begin{pmatrix}a+2c & b+2d\\ 2a+4c & 2b+4d\end{pmatrix}$$

il faut que

$$\begin{cases} a+2c=1, \\ b+2d=0, \\ 2a+4c=0, \\ 2b+4d=1, \end{cases} \quad \Longrightarrow \quad \begin{cases} a+2c=1, \\ a+2c=\frac{1}{2}, \end{cases}$$

ce qui est impossible : on conclut que la matrice $\begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$ n'est pas inversible. En effet, $\det \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix} = 0$ car la deuxième ligne est le double de la première.

```
inv([1 2; 2 4])
warning: matrix singular to machine precision
```

Attention à la précision machine! Soit $\mathbb{A} = \begin{pmatrix} 1 & 2 \\ 2 & 4+\varepsilon \end{pmatrix}$. Si $\varepsilon \neq 0$ alors $\det(\mathbb{A}) \neq 0$ et donc $\mathbb{A}$ est inversible. Cependant, si $\varepsilon \leq 1 \times 10^{-15}$ (précision machine), Matlab/Octave considérera $\varepsilon = 0$ et donc répondra que la matrice est singulière. Il faut toujours avoir du recul pour interpreter les réponses obtenues avec un ordinateur (peu importe le langage de programmation choisi).

Exercice 3.8

On dit que deux matrices $\mathbb{A}$ et $\mathbb{B}$ commutent si $\mathbb{AB} = \mathbb{BA}$. Trouver toutes les matrices qui commutent avec

$$\mathbb{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 5 \end{pmatrix}.$$

En déduire $\mathbb{A}^{-1}$.

Correction

On cherche $\mathbb{B}$ telle que

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 5 \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 5 \end{pmatrix}$$

Comme

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 5 \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ 3b_{21} & 3b_{22} & 3b_{23} \\ 5b_{31} & 5b_{32} & 5b_{33} \end{pmatrix}$$

et

$$\begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 5 \end{pmatrix} = \begin{pmatrix} b_{11} & 3b_{12} & 5b_{13} \\ b_{21} & 3b_{22} & 5b_{23} \\ b_{31} & 3b_{32} & 5b_{33} \end{pmatrix}$$

il faut que

$$\begin{cases} b_{11}=b_{11}, \\ b_{12}=3b_{12}, \\ b_{13}=5b_{13}, \\ 3b_{21}=b_{21}, \\ 3b_{22}=3b_{22}, \\ 3b_{23}=5b_{23}, \\ 5b_{31}=b_{31}, \\ 5b_{32}=3b_{32}, \\ 5b_{33}=5b_{33}, \end{cases} \quad \Longleftrightarrow \quad \mathbb{B} = \begin{pmatrix} \kappa_1 & 0 & 0 \\ 0 & \kappa_2 & 0 \\ 0 & 0 & \kappa_3 \end{pmatrix} \quad \text{avec } \kappa_1, \kappa_2, \kappa_3 \in \mathbb{R}.$$

Si de plus on veut que $\mathbb{AB} = \mathbb{BA} = \mathbb{I}_3$, *i.e.* $\mathbb{B} = \mathbb{A}^{-1}$, il faut $\kappa_1 = 1$, $\kappa_2 = 1/3$ et $\kappa_3 = 1/5$.

```
inv(diag([1 3 5]))
```

Exercice 3.9 (`fsolve, @, det`)
Trouver pour quelles valeurs de $t \in \mathbb{R}$ les matrices suivantes sont inversibles :

$$\mathbb{A} = \begin{pmatrix} t+3 & t^2-9 \\ t^2+9 & t-3 \end{pmatrix}, \qquad \mathbb{B} = \begin{pmatrix} t^2-9 & t+3 \\ t-3 & t^2+9 \end{pmatrix}.$$

Correction

$$\det(\mathbb{A}) = \det\begin{pmatrix} t+3 & t^2-9 \\ t^2+9 & t-3 \end{pmatrix} = (t+3)\times(t-3) - (t^2-9)\times(t^2+9) = -(t-3)(t+3)(t^2+8).$$

La matrice est inversible pour tout $t \in \mathbb{R}\setminus\{-3,3\}$.

```
determinant = @(t)[det([t+3, t^2-9; t^2+9, t-3])];
fsolve(determinant,1)
fsolve(determinant,-1)
```

$$\det(\mathbb{B}) = \det\begin{pmatrix} t^2-9 & t+3 \\ t-3 & t^2+9 \end{pmatrix} = (t^2-9)\times(t^2+9) - (t+3)\times(t-3) = (t-3)(t+3)(t^2+8).$$

La matrice est inversible pour tout $t \in \mathbb{R}\setminus\{-3,3\}$.

```
determinant=@(t)[det([t^2-9, t+3; t-3, t^2+9])];
fsolve(determinant,1)
fsolve(determinant,-1)
```

Exercice 3.10
Trouver pour quelles valeurs de t la matrice suivante est inversible

$$\begin{pmatrix} t+3 & -1 & 1 \\ 5 & t-3 & 1 \\ 6 & -6 & t+4 \end{pmatrix}.$$

Correction
On commence par calculer le déterminant de la matrice. Étant une matrice d'ordre 3, on peut par exemple utiliser la méthode de SARRUS :

$$\begin{aligned} &\det\begin{pmatrix} t+3 & -1 & 1 \\ 5 & t-3 & 1 \\ 6 & -6 & t+4 \end{pmatrix} \\ &\quad = \Big((t+3)\times(t-3)\times(t+4) + 5\times(-6)\times 1 + 6\times(-1)\times 1\Big) - \Big(1\times(t-3)\times 6 + 1\times(-6)\times(t+3) + (t+4)\times(-1)\times 5\Big) \\ &\quad = t^3 - 4t + 4t^2 - 16 = t(t^2-4) + 4(t^2-4) = (t^2-4)(t+4) = (t-2)(t+2)(t+4). \end{aligned}$$

La matrice est inversible pour tout $t \in \mathbb{R}\setminus\{-4,-2,2\}$.

```
determinant = @(t)[det([t+3, -1, 1; 5, t-3, 1; 6, -6, t+4])];
fsolve(determinant,1)
fsolve(determinant,-1)
fsolve(determinant,-10)
```

Exercice 3.11
Soit a, b et c trois réels quelconques, calculer les déterminants suivants :

$$D_1 = \det\begin{pmatrix} 1 & 1 & 1 \\ a & b & c \\ a^2 & b^2 & c^2 \end{pmatrix} \qquad D_2 = \det\begin{pmatrix} 1+a & 1 & 1 \\ 1 & 1+a & 1 \\ 1 & 1 & 1+a \end{pmatrix}$$

Correction
Pour calculer un déterminant comportant des paramètres, il est souvent intéressant de faire apparaître des zéros dans une ligne ou une colonne :

$$D_1 = \det\begin{pmatrix} 1 & 1 & 1 \\ a & b & c \\ a^2 & b^2 & c^2 \end{pmatrix} \xrightarrow[=]{\substack{C_2\leftarrow C_2-C_1 \\ C_3\leftarrow C_3-C_1}} \det\begin{pmatrix} 1 & 0 & 0 \\ a & b-a & c-a \\ a^2 & b^2-a^2 & c^2-a^2 \end{pmatrix} = \det\begin{pmatrix} b-a & c-a \\ b^2-a^2 & c^2-a^2 \end{pmatrix}$$
$$= (b-a)(c^2-a^2) - (c-a)(b^2-a^2) = (b-a)(c-a)\Big((c+a)-(b+a)\Big) = (b-a)(c-a)(c-b);$$
$$D_2 = \det\begin{pmatrix} 1+a & 1 & 1 \\ 1 & 1+a & 1 \\ 1 & 1 & 1+a \end{pmatrix} \xrightarrow[=]{\substack{C_2\leftarrow C_2-C_1 \\ C_3\leftarrow C_3-C_1}} \det\begin{pmatrix} 1+a & -a & -a \\ 1 & a & 0 \\ 1 & 0 & a \end{pmatrix} \xrightarrow[=]{L_1\leftarrow L_1+L_2+L_3} \det\begin{pmatrix} 3+a & 0 & 0 \\ 1 & a & 0 \\ 1 & 0 & a \end{pmatrix} = a^2(3+a).$$

Exercice 3.12 (`fsolve, @, det, rank`)

1. Pour quelles valeurs de $\kappa \in \mathbb{R}$ la matrice $\mathbb{A} = \begin{pmatrix} 1 & \kappa \\ 2 & 3 \end{pmatrix}$ est inversible ?
2. Calculer le rang des matrices $\mathbb{B} = \begin{pmatrix} 1 & 2 & 8 \\ 2 & 1 & 4 \\ 0 & 3 & 12 \end{pmatrix}$ et $\mathbb{C} = \begin{pmatrix} 2 & 1 & 3 \\ 8 & 4 & 12 \\ 1 & 2 & 0 \end{pmatrix}$.
3. Calculer le déterminant des matrices $\mathbb{D} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 2 & 3 & 7 & 4 \\ 3 & 1 & 12 & 0 \\ 4 & 0 & -5 & 0 \end{pmatrix}$ et $\mathbb{E} = \begin{pmatrix} 0 & 2 & 3 & 4 \\ 1 & 7 & 12 & -5 \\ 0 & 3 & 1 & 0 \\ 0 & 4 & 0 & 0 \end{pmatrix}$.

Correction

1. La matrice $\mathbb{A}$ est inversible pour $\kappa \neq \frac{3}{2}$ car $\det(\mathbb{A}) = 3 - 2\kappa$.

```
determinant = @(k)[det([1 k; 2 3])];
fsolve(determinant,0)
```

2. Sans faire de calcul on peut déjà affirmer que $1 \leq \mathrm{rg}(\mathbb{B}) \leq 3$. Comme $\det(\mathbb{B}) = 0$ (sans faire de calcul, il suffit de remarquer que $C_3 = 4C_2$), alors $1 \leq \mathrm{rg}(\mathbb{B}) \leq 2$. Comme $\det\begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} = -3 \neq 0$, on conclut que $\mathrm{rg}(\mathbb{B}) = 2$. De la même manière, $1 \leq \mathrm{rg}(\mathbb{C}) \leq 3$ et puisque $\det(\mathbb{C}) = 0$ (sans faire de calcul, il suffit de remarquer que $L_2 = 4L_1$), alors $1 \leq \mathrm{rg}(\mathbb{C}) \leq 2$. Comme $\det\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} = 3 \neq 0$, on conclut que $\mathrm{rg}(\mathbb{C}) = 2$.

```
rank([1 2 8; 2 1 4; 0 3 12])
rank([2 1 3; 8 4 12; 1 2 0])
```

3. $\det(\mathbb{D}) = \det\begin{pmatrix} 0 & 0 & \boxed{1} & 0 \\ 2 & 3 & 7 & 4 \\ 3 & 1 & 12 & 0 \\ 4 & 0 & -5 & 0 \end{pmatrix} = \det\begin{pmatrix} 2 & 3 & 4 \\ 3 & 1 & 0 \\ \boxed{4} & 0 & 0 \end{pmatrix} = 4\det\begin{pmatrix} 3 & 4 \\ 1 & 0 \end{pmatrix} = -16.$

$\det(\mathbb{E}) = \det\begin{pmatrix} 0 & 2 & 3 & 4 \\ \boxed{1} & 7 & 12 & -5 \\ 0 & 3 & 1 & 0 \\ 0 & 4 & 0 & 0 \end{pmatrix} = -\det\begin{pmatrix} 2 & 3 & 4 \\ 3 & 1 & 0 \\ \boxed{4} & 0 & 0 \end{pmatrix} = -4\det\begin{pmatrix} 3 & 4 \\ 1 & 0 \end{pmatrix} = 16.$

```
det([0 0 1 0; 2 3 7 4; 3 1 12 0; 4 0 -5 0])
det([0 2 3 4; 1 7 12 -5; 0 3 1 0; 0 4 0 0])
```

Exercice 3.13 (F. Le Roux)

En admettant le fait que les nombres 2001, 1073, 5800 et 8903 sont tous divisibles par 29, montrer que le déterminant de la matrice

$$\mathbb{A} = \begin{pmatrix} 2 & 0 & 0 & 1 \\ 1 & 0 & 7 & 3 \\ 5 & 8 & 0 & 0 \\ 8 & 9 & 0 & 3 \end{pmatrix}$$

est aussi divisible par 29 (*sans* calculer ce déterminant!).

Correction
Le déterminant ne change pas lorsque on ajoute à une colonne une combinaison linéaire des autres colonnes. Si on ajoute à la quatrième colonne la combinaison linéaire $\sum_{i=1}^{3} 10^{4-i} C_i$ on obtient

$$\sum_{i=1}^{3} 10^{4-i} C_i + C_4 = 10^3 \begin{pmatrix} 2 \\ 1 \\ 5 \\ 8 \end{pmatrix} + 10^2 \begin{pmatrix} 0 \\ 0 \\ 8 \\ 9 \end{pmatrix} + 10 \begin{pmatrix} 0 \\ 7 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 3 \\ 0 \\ 3 \end{pmatrix} = \begin{pmatrix} 2001 \\ 1073 \\ 5800 \\ 8903 \end{pmatrix} = 29 \begin{pmatrix} 69 \\ 37 \\ 200 \\ 307 \end{pmatrix}$$

donc

$$\begin{aligned} \det(\mathbb{A}) &= \det \begin{pmatrix} 2 & 0 & 0 & 29\times 69 \\ 1 & 0 & 7 & 29\times 37 \\ 5 & 8 & 0 & 29\times 200 \\ 8 & 9 & 0 & 29\times 307 \end{pmatrix} \\ &= -29\times 69\times \det \begin{pmatrix} 1 & 0 & 7 \\ 5 & 8 & 0 \\ 8 & 9 & 0 \end{pmatrix} + 29\times 37\times \det \begin{pmatrix} 2 & 0 & 0 \\ 5 & 8 & 0 \\ 8 & 9 & 0 \end{pmatrix} - 29\times 200\times \det \begin{pmatrix} 2 & 0 & 0 \\ 1 & 0 & 7 \\ 8 & 9 & 0 \end{pmatrix} + 29\times 307\times \det \begin{pmatrix} 2 & 0 & 0 \\ 1 & 0 & 7 \\ 5 & 8 & 0 \end{pmatrix} \\ &= 29\times \left[-69\times \det \begin{pmatrix} 1 & 0 & 7 \\ 5 & 8 & 0 \\ 8 & 9 & 0 \end{pmatrix} + 37\times \det \begin{pmatrix} 2 & 0 & 0 \\ 5 & 8 & 0 \\ 8 & 9 & 0 \end{pmatrix} - 200\times \det \begin{pmatrix} 2 & 0 & 0 \\ 1 & 0 & 7 \\ 8 & 9 & 0 \end{pmatrix} + 307\times \det \begin{pmatrix} 2 & 0 & 0 \\ 1 & 0 & 7 \\ 5 & 8 & 0 \end{pmatrix} \right] \end{aligned}$$

Exercice 3.14
Soit $n \geq 2$ un entier naturel pair. Une matrice de taille $n \times n$ est à remplir par deux joueurs, A (qui veut un déterminant non nul, commence) et B (qui veut un déterminant nul). Ils ont le droit de mettre n'importe quel réel dans une case vide de la matrice, chacun leur tour. Trouver une stratégie gagnante pour B.

Correction
Il suffit de faire en sorte qu'une colonne soit identique (ou un multiple) d'une autre.

3.2 Matlab/Octave

Attention

Pensez à inclure la commande `clear all` au début de vos scripts afin de nettoyer l'environnement de travail (ceci supprimera toutes les variables en mémoire). Vous pouvez également utiliser la commande `clc` pour vider la fenêtre de commandes. On peut écrire le tout en une seule ligne : `clear all, clc`

Pour chaque exemple ou exercice, écrivez les instructions dans un fichier script nommé par exemple `exo_1_3.m` (sans espaces, ni accents ni points sauf pour l'extension `.m`). Bien entendu, vous pouvez utiliser le mode interactif pour vérifier simplement une commande, mais chaque exercice devra finalement être résolu dans un fichier script. Tous ces scripts devront être situés dans un dossier dont le nom ne contient ni espaces, ni accents, ni points.

Exercice 3.15

Copier les instructions suivantes dans des *script files*. Exécuter les *script* et commenter les résultats. Si une instruction génère une erreur, expliquer pourquoi.

① Somme et produit de matrices, calcul de l'inverse d'une matrice :

```
A = [1 2 3; 4 5 6]
B = [7 8 9; 10 11 12]
C = [13 14; 15 16; 17 18]
A+B
A*C
A+C
inv(A)
A = [1 2; 0 0]
inv(A)
```

② La commande `diag`

```
v = [2 5 10]
A = diag(v,-1)
v = [2]
A = diag(v,-1)
```

③ Matrices triangulaires

```
A = [3 1 2; -1 3 4; -2 -1 3]
L1 = tril(A)
L2 = tril(A,-1)
```

Exercice 3.16 (Opérations élément par élément, produit scalaire, produit vectoriel)

① Copier les instructions suivantes dans un *script file*. Exécuter le *script* et commenter les résultats.

```
clear all;
clc;
x = [1; 2; 3]
v = x.^2
b = sum(x)

p = x.^x

y = [4; 5; 6]
u = x.*y
w = sum(x.*y)
d = dot(x,y)
xTy = x'*y
yTx = y'*x

c = cross(x,y)
```

② Définir le vecteur `x=[pi/6 pi/4 pi/3]` et calculer `s=sin(x)` et `c=cos(x)`. En déduire $\tan(x)$ à l'aide des vecteurs `s` et `c`.

③ Calculer la somme des nombres entiers de 1 à 500. Calculer la somme des carrés des nombres entiers de 1 à 500. Calculer la somme des nombres impaires inférieurs ou égaux à 500. Calculer la somme des nombres paires inférieurs ou égaux à 500.

Correction

① **x**, **y** et **v** sont des vecteurs-colonne 3×1

v est le vecteur tel que $v_i = x_i^2$ pour $i = 1,2,3$: $\mathbf{v} = (1,4,9)^T$

$b = x_1 + x_2 + x_3 = 1 + 2 + 3 = 6$

p est le vecteur tel que $p_i = (x_i)^{x_i}$ pour $i = 1,2,3$: $\mathbf{p} = (1,4,27)^T$
u est le vecteur tel que $u_i = x_i y_i$ pour $i = 1,2,3$: $\mathbf{u} = (4,10,18)^T$
w, d, $\mathbf{x}^T\mathbf{y}$ et $\mathbf{y}^T\mathbf{x}$ sont quatre méthodes pour calculer le produit scalaire de **x** et **y** qui est égale à $\sum_i y_i x_i = y_1x_1 + y_2x_2 + y_3x_3 = 4\times1+5\times2+6\times3 = 32$
c est le vecteur obtenu par le produit vectoriel de **x** et **y** : $\mathbf{c} = (-3,6,-3)^T$

②
```
x = [pi/6 pi/4 pi/3]
s = sin(x)
c = cos(x)
t = s./c
tan(x) % on verifie qu'on a le bon resultat
```

③ $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$

```
n = 500
sum([1:n])
n*(n+1)/2
```

$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$

```
n = 500
sum([1:n].^2)
n*(n+1)*(2*n+1)/6
```

$\sum_{i=1}^{N} (2i-1) = N^2$

```
n = 500
imp = sum(1:2:n)
N = length(1:2:n);
N^2
```

$\sum_{i=1}^{N} (2i) = N(N+1)$

```
n = 500
pair = sum(2:2:n)
N = length(2:2:n);
N*(N+1)
```

Exercice 3.17 (Sum of series : opérations pointées)

Pour $n, p > 0$ donnés, calculer les sommes suivantes en évitant les boucles lorsque possible.

$$\sum_{k=1}^{n} (2k-1)^p, \qquad \sum_{k=1}^{n} k(k+1), \qquad \sum_{k=1}^{n} (-1)^{k+1}(2k-1)^2.$$

Correction

```
% sum (2k-1)^2 for k=1...n
s1 = @(n,p) sum((2*[1:n]-1).^p)
% TEST
s1(3,1) % (2*1-1)^1 + (2*2-1)^1 + (2*3-1)^1 = 1+3+5 = 9

%  sum k(k+1) for k=1...n
s2 = @(n) sum( [1:n].*[2:n+1] )
% TEST
s2(3) % 1*(1+1) + 2*(2+1) + 3*(3+1) = 2+6+12 = 20

%  sum (-1)^(k+1) (2k-1)^2 for k=1...n
s3 = @(n) sum( (-1).^[2:n+1].*(2*[1:n]-1).^2 )
```

Exercice 3.18 (Vectorisation : boucles → opérations pointées, conditions → masques)

Réécrire les codes suivants sans utiliser de boucles :

①
```
[n,m]=size(a);
for i = 1:n
   for j = 1:m
      c(i,j) = a(i,j) + b(
          i,j);
   end
end
```

②
```
n=length(b);
for i = 1:n-1
   a(i) = b(i+1) - b(i);
end
```

③
```
n=length(a);
for i = 1:n
   if (a(i) > 5)
      a(i) -= 20
   end
end
```

Correction

①
```
c=a+b
```

②
```
a=b(2:n)-b(1:n-1)
```

③
```
a(a> 5)-= 20
```

Exercice 3.19 (Sommes, produits et algèbre linéaire pour éliminer les boucles)
Soient $\mathbf{u} =, \mathbf{v}, \mathbf{w}, \mathbf{x}, \mathbf{y}$ des vecteurs ligne de $\mathbb{R}^n$. On se propose de calculer

$$\sum_{i=1}^{n} u_i v_i, \qquad \sum_{i=1}^{n} w_i x_i^2, \qquad \sum_{i=1}^{n} w_i x_i y_i.$$

Correction

1. Notons que $\sum_{i=1}^{n} u_i v_i = \mathbf{u} \cdot \mathbf{v}^T$ donc

```
u = [1 2 3 4 5];
v = [3 6 8 9 10];

% Avec une boucle
y = 0;
for i=1:length(u)
   y = y + u(i)*v(i);
end
y
```

```
% dot notation
y = sum(u.*v)

% produit scalaire
y = u*v'
y = v*u'
y=dot(u,v)
y=dot(v,u)
```

2. $\sum_{i=1}^{n} w_i x_i^2 = \mathbf{x}\mathbb{D}_{\mathbf{w}}\mathbf{x}^T$ avec

$$\mathbb{D}_{\mathbf{w}} = \begin{pmatrix} w_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & & w_n \end{pmatrix}$$

donc

```
w = [0.1 0.25 0.12 0.45 0.98];
x = [9 7 11 12 8];

% Avec une boucle
y = 0;
for i = 1:length(w)
    y = y + w(i)*x(i)^2;
end
```

```
y

% dot notation
y = sum(w.*(x.^2))

% algebre lieaire
y = x*diag(w)*x'
```

3. $\sum_{i=1}^{n} w_i x_i y_i = \mathbf{x}\mathbb{D}_{\mathbf{w}}\mathbf{y}^T$ donc

```
w = [0.1 0.25 0.12 0.45 0.98];
x = [9 7 11 12 8];
y = [2 5 3 8 0];

% Avec une boucle
z = 0;
for i=1:length(w)
    z = z + w(i)*x(i)*y(i);
end
```

```
z

% dot notation
z = sum(w.*x.*y)

% algebre lieaire
z = x*diag(w)*y'
z = y*diag(x)*w'
z = w*diag(y)*x'
```

Exercice 3.20
① Copier les instructions suivantes dans un *script file*. Exécuter le *script* et commenter les résultats.

```
A=[1 2; 4 5];
B=[1 0; 1 1];
```

```
A*B
A.*B
```

```
A^2
A.^2
```

```
A/B
A.\B
```

② Afficher la table de multiplication par $1,\dots,10$, *i.e.* la matrice

	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

Correction

① `A*B` calcule le produit $\mathbb{A}\mathbb{B} = (\sum_{k=1}^{3} a_{ik}b_{kj})_{1\le i,j\le 3}$, `A.*B` calcule la matrice $\mathbb{C} = (a_{ij}b_{ij})_{1\le i,j\le 3}$.

```
>> A*B
ans =
   3   2
   9   5
```

```
>> A.*B
ans =
   1   0
   4   5
```

`A^2` calcule le produit $\mathbb{A}\mathbb{A} = (\sum_{k=1}^{3} a_{ik}a_{kj})_{1\le i,j\le 3}$, `A.^2` calcule la matrice $\mathbb{C} = (a_{ij}^2)_{1\le i,j\le 3}$.

```
>> A^2
ans =
    9   12
   24   33
```

```
>> A.^2
ans =
    1    4
   16   25
```

`A/B` calcule le produit $\mathbb{A}\mathbb{B}^{-1}$ si $\mathbb{B}$ est inversible, `A.\B` calcule la matrice $\mathbb{C} = (a_{ij}/b_{ij})_{1\le i,j\le 3}$.

```
>> A/B
ans =
  -1   2
  -1   5
```

```
>> A.\B
ans =
   1.00000   0.00000
   0.25000   0.20000
```

② Avec les instructions suivantes, on construit la matrice $\mathbb{A}$ qui contient juste les produits, et la matrice $\mathbb{B}$ qui contient aussi l'entête des lignes et colonnes :

```
A(:,:)=[1:10]'.*[1:10]
B(:,:)=[1,1:10]'.*[1,1:10]
```

Exercice 3.21 (Construction de matrices)

① Écrire les instructions pour construire une matrice triangulaire supérieure de dimension 10 ayant des 2 sur la diagonale principale et des -3 sur la seconde sur-diagonale.

$$\begin{bmatrix} 2 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & -3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & -3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & -3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & -3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & -3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & -3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

② Écrire les instructions permettant d'interchanger la troisième et la septième ligne de la matrice construite au

point précédent, puis les instructions permettant d'échanger la quatrième et la huitième colonne.

③ Écrire la matrice carrée de taille n comprenant des n sur la diagonale principale, des $n-1$ sur les deux lignes qui l'encadrent, etc.

④ Écrire la matrice à n lignes et m colonnes dont la première colonne ne contient que des 1, la deuxième colonne ne contient que des 2, etc. Écrire ensuite la matrice à m lignes et n colonnes dont la première ligne ne contient que des 1, la deuxième ligne ne contient que des 2, etc.

⑤ Écrire la matrice carrée $\mathbb{A}$ de taille $2n+1$ comportant des 1 sur la $(n+1)^{\text{ième}}$ ligne et la $(n+1)^{\text{ième}}$ colonne et des 0 ailleurs.

⑥ Écrire la matrice carrée $\mathbb{Z}$ de taille n comportant des 1 sur la première et dernière ligne et sur la deuxième diagonale et des 0 ailleurs.

⑦ Étant donné une liste de nombres retourner la liste obtenue en écrivant d'abord les termes de rang pair suivis des termes de rang impair.

⑧ Écrire la matrice (n,n) dont les éléments sont $1,2,\ldots,n^2$ écrits dans l'ordre habituel (sur chaque ligne, de la gauche vers la droite, de la première à la dernière ligne).

Correction

①
```
U = 2*eye(10)+diag(-3*ones(8,1),2)
```

② On peut échanger les troisième et septième lignes de la matrice (sans modifier la matrice initiale) avec les instructions :

```
r = [1:10]
r(3) = 7
r(7) = 3
Ur = U(r,:)
```

Remarquer que le caractère `:` dans `U(r,:)` fait que toutes les colonnes de `U` sont parcourues dans l'ordre croissant habituel (du premier au dernier terme).

Sinon, si on veut modifier la matrice initiale, on peut utiliser l'instruction :

```
U([3 7],:) = U([7 3],:)
```

Pour échanger les quatrième et huitième colonnes on peut écrire

```
c = [1:10]
c(8) = 4
r(4) = 8
Uc = U(:,c)
```

③ En utilisant deux boucles

```
for i = 1:n
   for j = 1:n
      M(i,j) = n-abs(i-j);
   end
end
```

qu'on peut écrire en version compacte

```
for i = 1:n
   M(i,1:n) = n-abs(i-[1:n]);
end
```

En utilisant l'instruction **`diag`**

```
M = diag(n*ones(1,n));
for i = 2:n
   M += diag((n-i+1)*ones(1,n-i+1),i-1)+diag((n-i+1)*ones(1,n-i+1),1-i);
end
```

④ Soit n et m fixés.

```
A(1:n,:) = ones(n,1)*[1:m]
A'
```

⑤
```
n = 5;
A = zeros(2*n+1);
A(:,n+1) = 1;
A(n+1,:) = 1
```

⑥
```
n = 5;
A = eye(n);
A(1,:) = 1;
A(n,:) = 1;
A = A(:,n:-1:1)
```

⑦
```
liste = rand(1,10)
liste2 = [liste([2:2:end]) liste([1:2:end])]
```

⑧
```
n = 5;
M = [1:n^2];
M = reshape(M,n,n)'
```

Exercice 3.22 (Construction de matrices, vectorisation, `script` et `function`)

1. Dans un **fichier `zorro.m`** écrire une **`function`** appelée `zorro` qui prend en entrée un entier $n \in \mathbb{N}^*$ et renvoi la matrice carrée $\mathbb{Z}$ de taille n comportant des 1 sur la première et dernière ligne et sur la deuxième diagonale et des 0 ailleurs (sans utiliser de boucles).
   Par exemple, pour $n = 5$, la commande `Z=zorro(5)` devra donner
$$\mathbb{Z} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$
2. Dans un fichier **`exercice1.m`** écrire un **script** pour tester cette fonction pour $n = 1, \ldots, 5$.

Dans le fichier `zorro.m` on écrit la fonction

```
function A=zorro(n)
  A=eye(n);
  A(1,:)=1;
  A(n,:)=1;
  A=A(:,n:-1:1);
end
```

Dans le fichier `exercice1.m` on écrit le **script**

```
for n=1:5
   Z=zorro(n)
end
```

Exercice 3.23 (Coût (en temps) d'un produit matrice-vecteur)

Exécuter les instructions suivantes et commenter :

```
n = 10000;
step = 100;
A = rand(n,n);
v = rand(n,1);
T = [];
sizeA = [];
for k = 500:step:n
   AA = A(1:k,1:k);
   vv = v(1:k);
   t = cputime;
   b = AA*vv;
   tt = cputime - t;
   T = [T, tt];
   sizeA = [sizeA,k];
end
```

```
plot(sizeA,T,'o')
```

Correction
L'instruction `a:step:b` intervenant dans la boucle **`for`** génère tous les nombres de la forme `a+step*k` où `k` est un entier variant de `0` à `kmax`, où `kmax` est le plus grand entier tel que `a+step*kmax` est plus petit que `b` (dans le cas considéré, `a=500`, `b=10000` et `step=100`). La commande **`rand`**`(n,m)` définit une matrice $n \times m$ dont les éléments sont aléatoires. Enfin, `T` est le vecteur contenant les temps CPU nécessaires à chaque produit matrice-vecteur, et **`cputime`** renvoie le temps CPU (en secondes) consommé par Octave depuis son lancement. Le temps nécessaire à l'exécution d'un programme est donc la différence entre le temps CPU effectif et celui calculé juste avant l'exécution du programme courant, stocké dans la variable `t`. La commande **`plot`**`(sizeA,T,'o')`, montre que le temps CPU augmente comme le carré de n l'ordre de la matrice.

Exercice 3.24 (Boucle `for`)
Il est possible de calculer les premières décimales de π avec l'aide du hasard. On considère un carré de coté 1 et un cercle de rayon 1 centré à l'origine :

Si on divise l'aire de la portion de disque par celle du carré on trouve $\frac{\pi}{4}$. Si on tiré au hasard dans le carré, on a une probabilité de $\frac{\pi}{4}$ que le point soit dans la portion de disque. On considère l'algorithme suivant pour approcher π : on génère N couples $\{(x_k, y_k)\}_{k=1}^{N}$ de nombres aléatoires dans l'intervalle $[0,1]$, puis on calcule le nombre $m \leq N$ de ceux qui se trouvent dans le premier quart du cercle unité. π est la limite de la suite $4m/N$ lorsque $N \to +\infty$. Écrire un programme pour calculer cette suite et observer comment évolue l'erreur quand N augmente.

Correction
La méthode proposée est une méthode de Monte Carlo. Elle est implémentée dans le programme suivant :

```
format long

N = 10^4
# On tire au hasard N points [x,y] dans [0,1[ x [0,1[
[xx,yy] = rand(N,2);
# Nombre de tirs dans le disque
m = sum( xx.^2+yy.^2<=1 );
myPi = 4*m/N
err = abs(pi-myPi)
```

La commande **`rand`** génère une suite de nombres pseudo-aléatoires. L'instruction `z <= 1` se lit de la manière suivante : on teste si `z(k)<= 1` pour chaque composante du vecteur `z` ; si l'inégalité est satisfaite pour la k-ème composante de `z` (c'est-à-dire, si le point (x_k, y_k) appartient à l'intérieur du disque unité) on donne la valeur 1, sinon on lui donne la valeur 0. La commande **`sum`**`(z <= 1)` calcule la somme de toutes les composantes de ce vecteur, c'est-à-dire le nombre de points se trouvant à l'intérieur du disque unité.
On peut réécrire le tout comme une fonction anonyme :

```
format long
myPi = @(N) [ 4*sum( rand(N,1).^2+rand(N,1).^2<=1 )/N ];
err = abs(pi-myPi(10^4)) % test
```

On exécute maintenant le programme pour différentes valeurs de N. Plus N est grand, meilleure est l'approximation de π. Par exemple, pour $N = 1000$ on obtient 3.1120, tandis qu'avec $N = 300000$ on a 3.1406 (naturellement, comme les nombres sont générés aléatoirement, les résultats obtenus pour une même valeur de N peuvent changer à chaque exécution).

```
format long
myPi = @(N) [ 4*sum( rand(N,1).^2+rand(N,1).^2<=1 )/N ];
NN = 100:100:10000;
for i = 1:length(NN)
    err(i) = abs(pi-myPi(NN(i)));
end
plot(NN,err)
```

Cette méthode n'est pas très efficace, il faut beaucoup de tirs pour obtenir le deux premières décimales de π.

Exercice 3.25 (`function`)

Comme π vérifie

$$\pi = \lim_{N\to+\infty} \sum_{n=0}^{N} 16^{-n}\left(\frac{4}{8n+1} - \frac{2}{8n+4} - \frac{1}{8n+5} - \frac{1}{8n+6}\right)$$

on peut calculer une approximation de π en sommant les N premiers termes, pour N assez grand.

- ⋆ Écrire une fonction pour calculer les sommes partielles de cette série (*i.e.* pour $n = 0\dots N$ avec N donné en paramètre).
- ⋆ Pour quelles valeurs de N obtient-on une approximation de π aussi précise que celle fournie par la variable π ?

Correction

Pour répondre à la question on peut utiliser le script suivant :

```
format long

piapproche = @(v) sum( ( 4./(8*v+1) - 2./(8*v+4) - 1./(8*v+5) - 1./(8*v+6) ).*(1/16).^v );

N = 0;
while abs(pi-piapproche([0:N]))>0
    N += 1; # seulement Octave, dans Matlab N = N+1;
end
N
piapproche([0:N])
pi
```

Pour $n = 10$ on obtient une approximation de π qui coïncide (à la précision Octave) avec la variable interne `pi` d'Octave. Cet algorithme est en effet extrêmement efficace et permet le calcul rapide de centaines de chiffres significatifs de π.

Exercice 3.26 (`function`)

Fabriquer une fonction-octave qui calcule le volume v d'un cylindre de révolution de hauteur h et dont la base est un disque de rayon r. Cette fonction doit accepte que r et h soient des listes de nombres et renvoyer un tableau. Si r est un vecteur de n nombres et h de m nombres alors v sera une matrice $\mathbb{A}$ de dimension $n \times m$ telle que $a_{ij} = v(r_i, h_j)$.

Correction

On écrit la fonction soit avec `function` soit avec une fonction anonyme, puis on valide la fonction avec un test dont on connaît le résultat :

```
function v=cylindre(r,h) % h et r sont 2 matrices-lignes
   v = (r.^2)'.*h *pi;
end
% cylindre = @(r,h) (r.^2)'.*h *pi ;
```

Quand on exécute cette fonction, on obtient un tableau à n lignes et à p colonnes qui sont les volumes recherchés. Sur les lignes de ce tableau, on lit les volumes pour r fixé, h variant. Sur les colonnes, c'est h qui est fixé.

Exercice 3.27

Un dispositif fournit un signal $s(t) = A\sin(2\pi t + \varphi)$ avec A et φ inconnus. On mesure le signal à deux instants (en ms) : $s(0.5) = -1.76789123$ et $s(0.6) = -2.469394443$. On posera $\alpha = A\cos(\varphi)$ et $\beta = A\sin(\varphi)$.

1. Écrire et résoudre le système d'inconnue α et β. En déduire A et φ.
2. Tracer le signal et montrer qu'il passe par les points mesurés.

Correction

Rappel : $\sin(a+b) = \sin(a)\cos(b) + \cos(a)\sin(b)$ ainsi

$$s(t) = A\sin(2\pi t + \varphi) = A\big(\sin(2\pi t)\cos(\varphi) + \cos(2\pi t)\sin(\varphi)\big) = \alpha\sin(2\pi t) + \beta\cos(2\pi t).$$

On doit donc résoudre le système linéaire :

$$\begin{cases} \alpha\sin(\pi) + \beta\cos(\pi) = -1.76789123 \\ \alpha\sin\left(\frac{6}{5}\pi\right) + \beta\cos\left(\frac{6}{5}\pi\right) = -2.469394443 \end{cases}$$

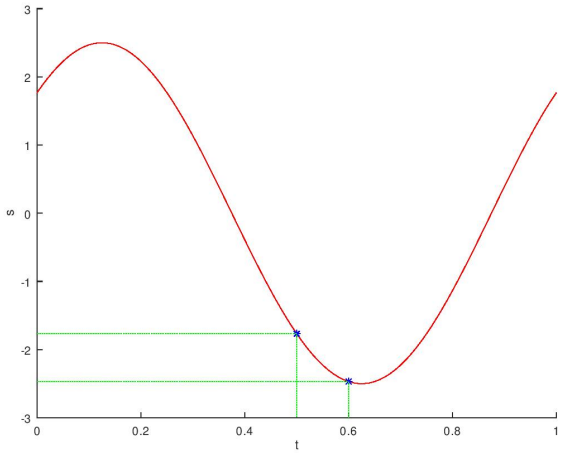
qu'on écriture matricielle s'écrit

$$\begin{pmatrix} \sin(\pi) & \cos(\pi) \\ \sin\left(\frac{6}{5}\pi\right) & \cos\left(\frac{6}{5}\pi\right) \end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}-1.76789123\\ -2.469394443\end{pmatrix} \quad i.e. \quad \begin{pmatrix} 0 & -1 \\ \sin\left(\frac{6}{5}\pi\right) & \cos\left(\frac{6}{5}\pi\right) \end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}-1.76789123\\ -2.469394443\end{pmatrix}$$

```
xx = [sin(2*pi*0.5) , cos(2*pi*0.5) ;  sin(2*pi*0.6) , cos(2*pi*0.6)]\[-1.76789123;-2.469394443]
alpha = xx(1)
beta = xx(2)
```

Puisqu'on trouve $\alpha = \beta$, alors $\cos(\varphi) = \sin(\varphi)$, *i.e.* $\varphi = \frac{\pi}{4}$ et $A = \sqrt{2}\alpha$ avec $\alpha \approx 1.7679$:

```
s = @(t) sqrt(2)*alpha*sin(2*pi*t+pi/4);
tt = [0:0.01:1];
hold on
plot(tt,s(tt),'r-') % la courbe
plot([0.5 0.6],[-1.76789123;-2.469394443],'b*') % les deux points
plot([0 0.5 0.5], [-1.76789123 -1.76789123 -3], 'g:') % trait pointille
plot([0 0.6 0.6], [-2.469394443 -2.469394443 -3], 'g:') % trait pointille
xlabel('t')
ylabel('s')
hold off
print("signal.jpg") % sauvagarde de la figure en .jpg
```



Exercice 3.28 (Vectorisation - `if` → masques)
Considérons la fonction $f : \mathbb{R} \to \mathbb{R}$ définie par

$$f(x) = \begin{cases} 0 & \text{si } x < 0, \\ x & \text{si } 0 \le x < 1, \\ 2 - x & \text{si } 1 \le x \le 2, \\ 0 & \text{si } x > 2. \end{cases}$$

Écrire cette fonction de deux manières différentes et en afficher le graphe :

1. avec une instruction conditionnelle du type `if ... elseif ... else` (vectorisée) dans une fonction définie avec `function`,
2. avec une instruction conditionnelle vectorisée dans une fonction anonyme.

Correction

Avec function La fonction (non vectorisée) peut s'écrire comme suit :

```
function y = f(x)
   if x<0
      y = 0;
   elseif x<1
```

```
      y = x;
   elseif x<=2
      y = 2-x;
   else
      y = 0;
   end
end
```

Pour la tester, il fat d'abord la **sauvegarder dans un fichier qui porte le même nom** (ici le fichier devra s'appeler `f1.m`) puis on écrira **un autre fichier qui contient le script** (qui se trouve dans le même dossier) **pour la tester** :

```
xx = [-1:0.1:3];
yy = arrayfun(@f,xx);
plot(xx,yy,'r')
```

Pour écrire **un script qui contient directement cette fonction** (sans utiliser deux fichiers séparés, l'un contenant la fonction et l'autre le script), **il faut décider s'il sera exécuté avec Matlab ou avec Octave** :

Matlab

```
xx = [-1:0.1:3];
yy = arrayfun(@f,xx);
plot(xx,yy,'r')

function y = f(x)
   if x<0
      y = 0;
   elseif x<1
      y = x;
   elseif x<=2
      y = 2-x;
   else
      y = 0;
   end
end
```

Octave

```
1;

function y = f(x)
   if x<0
      y = 0;
   elseif x<1
      y = x;
   elseif x<=2
      y = 2-x;
   else
      y = 0;
   end
end

xx = [-1:0.1:3];
yy = arrayfun(@f,xx);
plot(xx,yy,'r')
```

Anonyme La fonction anonyme vectorisée peut s'écrire comme suit :

```
f = @(x) [  0*(x<0) + x.*(x>=0).*(x<1) + (2-x).*(x>=1).*(x<2) + 0.*(x>2) ];
```

Cette fonction peut être écrite directement avec le script et on a alors **le même script en Octave ou en Matlab** :

```
f = @(x) [  0*(x<0) + x.*(x>=0).*(x<1) + (2-x).*(x>=1).*(x<2) + 0.*(x>2) ];

xx = [-1:0.1:3];
yy = f(xx);
plot(xx,yy,'b.')
```

Exercice 3.29

Soit **x** un vecteur de n valeurs. On rappelle les définitions suivantes (pour l'estimation sur une population à partir d'un échantillon) :

Moyenne arithmétique $\overline{m} = \frac{1}{n}\sum_{i=1}^{n} x_i$

Variance $V = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{m})^2$

Écart-type $\sigma = \sqrt{V}$

Médiane C'est la valeur qui divise l'échantillon en deux parties d'effectifs égaux. Soit **y** le vecteur qui contint les

composantes de **x** ordonné, alors

$$\text{médiane} = \begin{cases} \dfrac{y_{n/2} + y_{n/2+1}}{2} & \text{si } n \text{ est pair,} \\ y_{n/2+1} & \text{si } n \text{ est impair.} \end{cases}$$

Écrire une fonction qui renvoie la moyenne, l'écart type et la médiane d'un vecteur donné en entré. Vérifier l'implémentation sur un vecteur au choix en comparant avec les valeurs obtenues par les fonctions prédéfinies.

Correction
Dans cette correction on utilise deux fichiers séparés, l'un contenant la fonction et l'autre le script. On peut écrire la fonction et le script dans un même fichier comme indiqué dans l'exercice précédent mais il faudra décider en amont si on exécutera le script avec Matlab ou Octave.

'Fichier stat.m'

```
function [moy,et,med]  =  stat(x)
   n  =  length(x);
   % MOYENNE  =  fonction predefinie mean(x)
   moy  =  sum(x)/n;
   % ECART_TYPE  =  fonction predefinie std(x)
   et  =  sqrt(sum((x-moy).^2)/(n-1));
   % MEDIANE  =  fonction predefinie median(x)
   y = sort(x);
   if (rem(n,2) =  = 0) % si n est un nombre pair
      med  =  (y(n/2)+y((n/2)+1))/2;
   else
      med  =  y((n+1)/2);
   end
end
```

'Script'

```
clear all

x = [0 0 8 1 1 2 2]
[my_moy,my_std,my_mediane]  =  stat(x)
octave_moy = mean(x)
octave_std = std(x)
octave_mediane = median(x)

x = [2 3 3 2]
[my_moy,my_std,my_mediane]  =  stat(x)
octave_moy = mean(x)
octave_std = std(x)
octave_mediane = median(x)
```

Exercice 3.30 (Résolution graphique d'une équation)

Soit la fonction

$$f\colon [-10,10] \to \mathbb{R}$$
$$x \mapsto \frac{x^3\cos(x) + x^2 - x + 1}{x^4 - \sqrt{3}x^2 + 127}$$

1. Tracer le graphe de la fonction f en utilisant seulement les valeurs de $f(x)$ lorsque la variable x prend successivement les valeurs $-10, -9.2, -8.4, \ldots, 8.4, 9.2, 10$ (*i.e.* avec un pas 0.8).
2. Apparemment, l'équation $f(x) = 0$ a une solution α voisine de 2. En utilisant le zoom, proposer une valeur approchée de α.
3. Tracer de nouveau le graphe de f en faisant varier x avec un pas de 0.05. Ce nouveau graphe amène-t-il à corriger la valeur de α proposée?
4. Demander à Octave d'approcher α (fonction `fsolve`).

Correction
En utilisant un pas de 0.8 il semblerai que $\alpha = 1.89$. En utilisant un pas de 0.05 il semblerai que $\alpha = 1.965$. En utilisant la fonction `fsolve` on trouve $\alpha = 1.9629$.

```
clear all; clc;
f = @(x) [(x.^3 .*cos(x)+x.^2-x+1) ./ (x.^4-sqrt(3)*x.^2+127)] ;

subplot(1,2,1)
xx=[-10:0.8:10];
plot(xx,f(xx),'r-')
grid()

subplot(1,2,2)
xx=[-10:0.05:10];
plot(xx,f(xx),'r-')
grid()
```

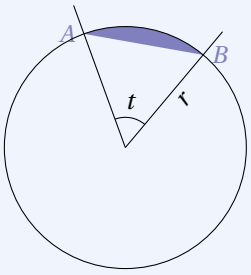
```
fsolve(f,1.9)
```

Exercice 3.31 (`fsolve`, `plot`)

On se propose ici d'utiliser Octave pour résoudre graphiquement des équations.

Considérons un cercle de rayon r. Si nous traçons un angle t (mesuré en radians) à partir du centre du cercle, les deux rayons formant cet angle coupent le cercle en A et B. Nous appelons a l'aire délimitée par la corde et l'arc AB (en bleu sur le dessin). Cette aire est donnée par $a = \frac{r^2}{2}(t - \sin(t))$. Pour un cercle donné (c'est à dire un rayon donné), nous choisissons une aire (partie en bleu) a. Quelle valeur de l'angle t permet d'obtenir l'aire choisie? Autrement dit, connaissant a et r, nous voulons déterminer t solution de l'équation

$$\frac{2a}{r^2} = t - \sin(t).$$



1. Résoudre graphiquement l'équation en traçant les courbes correspondant aux membres gauche et droit de l'équation (pour $a = 4$ et $r = 2$). Quelle valeur de t est solution de l'équation?
2. Comment faire pour obtenir une valeur plus précise du résultat?

Correction

```
t = [0:pi/180:2*pi];
rhs = t-sin(t);
a = 4;
r = 2;
lhs = 2*a/(r^2)*ones(size(t));
plot(t,rhs,t,lhs), grid
```

Le graphe nous dit que la solution est entre 2 et 3. On peut calculer une solution approchée comme suit :

```
fzero( @(x) 2*a/(r^2)-(x-sin(x)) , 2.5)
```

Attention

Jusqu'ici nous avons représenté des courbes engendrées par des équations cartésiennes, c'est-à-dire des fonctions de la forme $y = f(x)$. Pour cela, nous générons d'abord un ensemble de valeurs $\{x_i\}_{i=0\ldots N}$ puis l'ensemble de valeurs $\{y_i\}_{i=0\ldots N}$ avec $y_i = f(x_i)$. Il existe d'autres types de courbes comme par exemple les courbes paramétrées (engendrées par des équations paramétriques). Les équations paramétriques de courbes planes sont de la forme

$$\begin{cases} x = u(t), \\ y = v(t), \end{cases}$$

où u et v sont deux fonctions cartésiennes et le couple $(x; y)$ représente les coordonnées d'un point de la courbe paramétrée. La courbe engendrée par l'équation cartésienne $y = f(x)$ est une courbe paramétrée car il suffit de poser $u(t) = t$ et $v(t) = f(t)$. Un type particulier de courbe paramétrique est constitué par les équations polaires de courbes planes qui sont de la forme [a]

$$\begin{cases} x = r(t)\cos(t), \\ y = r(t)\sin(t). \end{cases}$$

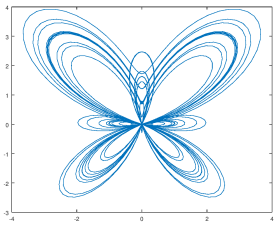
a. r représente la distance de l'origine O du repère $(O; \mathbf{i}, \mathbf{j})$ au point $(x; y)$ de la courbe, et t l'angle avec l'axe des abscisses.

Exercice 3.32 (Courbe parametrée)

Tracer la courbe papillon ($t \in [0;100]$) :

$$\begin{cases} x(t) = \sin(t)\left(e^{\cos(t)} - 2\cos(4t) - \sin^5\left(\frac{t}{12}\right)\right) \\ y(t) = \cos(t)\left(e^{\cos(t)} - 2\cos(4t) - \sin^5\left(\frac{t}{12}\right)\right) \end{cases}$$

```
x = @(t) [ sin(t).*( exp(cos(t))-2*cos(4*t)-(sin(t/12)).^5 ) ];
y = @(t) [ cos(t).*( exp(cos(t))-2*cos(4*t)-(sin(t/12)).^5 ) ];
tt = [0:0.05:100];
plot(x(tt),y(tt))
```

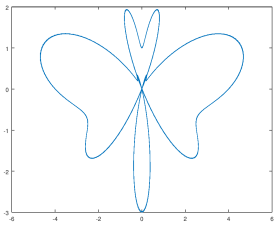


Exercice 3.33 (Équation polaire)

Tracer la courbe papillon ($t \in [0;100]$) :

$$\begin{cases} x(t) = r(t)\cos(t) \\ y(t) = r(t)\sin(t) \end{cases} \quad \text{avec } r(t) = \sin(7t) - 1 - 3\cos(2t).$$

```
r = @(t) [sin(7*t)-1-3*cos(2*t)];
x = @(t) [ r(t).*cos(t) ];
y = @(t) [ r(t).*sin(t) ];
tt = [0:0.05:100];
plot(x(tt),y(tt))
```

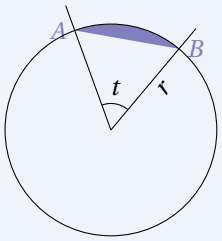


Exercice 3.34

On se propose ici d'utiliser Octave pour résoudre graphiquement des équations.

Considérons un cercle de rayon r. Si nous traçons un angle t (mesuré en radians) à partir du centre du cercle, les deux rayons formant cet angle coupent le cercle en A et B. Nous appelons a l'aire délimitée par la corde et l'arc AB (en bleu sur le dessin). Cette aire est donnée par $a = \dfrac{r^2}{2}(t - \sin(t))$. Pour un cercle donné (c'est à dire un rayon donné), nous choisissons une aire (partie en bleu) a. Quelle valeur de l'angle t permet d'obtenir l'aire choisie? Autrement dit, connaissant a et r, nous voulons déterminer t solution de l'équation

$$\frac{2a}{r^2} = t - \sin(t).$$



1. Résoudre graphiquement l'équation en traçant les courbes correspondant aux membres gauche et droit de l'équation (pour $a = 4$ et $r = 2$). Quelle valeur de t est solution de l'équation?
2. Comment faire pour obtenir une valeur plus précise du résultat?

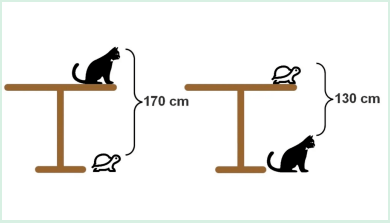
```
t = [0:pi/180:2*pi];
rhs = t-sin(t);
a = 4;
r = 2;
lhs = 2*a/(r^2)*ones(size(t));
plot(t,rhs,t,lhs), grid
```

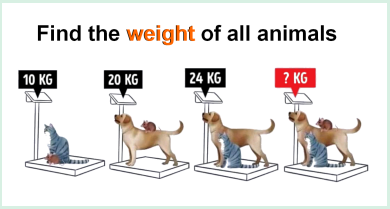
Le graphe nous dit que la solution est entre 2 et 3. On peut calculer une solution approchée comme suit :

```
fsolve( @(x) 2*a/(r^2)-(x-sin(x)) , 2.5)
```

3.3 Systèmes linéaires

Exercice 3.35
Énigme : arriverez-vous à déterminer la hauteur de cette table ? Et le poids de tous les animaux ?





Sources : https://www.science-et-vie.com/questions-reponses/enigme-arriverez-vous-a-determiner-la-hauteur-de-cette-table-66912 et https://mathematicsart.com/solved-exercises/solution-find-the-weight-of-all-animals/

Correction
Pour mettre en équations le schéma, on notera C la hauteur du chat, T la hauteur de la table et t la hauteur de la tortue. On a alors le système

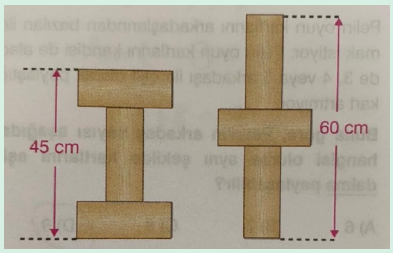
$$\begin{cases} C+T-t=170, \\ t+T-C=130. \end{cases}$$

On a trois inconnues et 2 équations : le système est sous-déterminé. Cependant, il est possible de calculer T. En effet, si on somme les deux équations on obtient $2T=300$ d'où $T=150$.
Pour mettre en équations le deuxième problème, on notera C le poids du chat, M celui de la sourie et D celui du chien. On a alors le système

$$\begin{cases} C+M=10, \\ D+M=20, \\ C+D=24. \end{cases}$$

On a trois inconnues et 3 équations linéaires. On remarque que $(C+M)+(D+M)+(C+D)=2(C+D+M)$ donc $2(C+D+M)=10+20+24=54$ et finalement $C+D+M=27$.

Exercice 3.36
Les rectangles sont identiques. Que vaut le périmètre d'un rectangle ?



Correction

$$\begin{cases} 2\ell+L=45 \\ \ell+2L=60 \end{cases} \rightsquigarrow \quad 2\ell+2L=70$$

3.3.1 Systèmes linéaires : utilisation de fonctions prédéfinies

Exercice 3.37 (Systèmes linéaires)
On a demandé à 215 étudiants de dire quel est leur langage de programmation préféré parmi Python, Matlab, Java et C. Chaque étudiant devait fournir une seule réponse. On sait que 163 étudiants ont déclaré préférer Python ou Matlab ; 65 ont déclaré préférer Matlab ou C ; 158 ont déclaré préférer Python ou C.

Traduire les données par un système linéaire de 4 équations et 4 inconnues et le résoudre par la méthode de Gauss.

Correction
On note p, m, j et c le nombre d'étudiants préférant respectivement Python, Matlab, Java et C. D'après l'énoncé on a

$$\begin{cases} p+m+j+c=215, \\ p+m=163, \\ m+c=65, \\ p+c=158. \end{cases}$$

On passe à la notation matricielle et on résout le système par la méthode de Gauss (4 équations donc 3 étapes) :

$$\left(\begin{array}{cccc|c} 1&1&1&1&215\\ 1&1&0&0&163\\ 0&1&0&1&65\\ 1&0&0&1&158 \end{array}\right) \xrightarrow[\substack{L_3\leftarrow L_3\\ L_4\leftarrow L_4-L_1}]{L_2\leftarrow L_2-L_1} \underset{\text{Étape 1}}{} \left(\begin{array}{cccc|c} 1&1&1&1&215\\ 0&0&-1&-1&-52\\ 0&1&0&1&65\\ 0&-1&-1&0&-57 \end{array}\right) \xrightarrow{L_2\leftrightarrow L_3} \underset{\text{Pivot nul}}{} \left(\begin{array}{cccc|c} 1&1&1&1&215\\ 0&1&0&1&65\\ 0&0&-1&-1&-52\\ 0&-1&-1&0&-57 \end{array}\right)$$

$$\xrightarrow[\substack{L_3\leftarrow L_3\\ L_4\leftarrow L_4+L_2}]{} \underset{\text{Étape 2}}{} \left(\begin{array}{cccc|c} 1&1&1&1&215\\ 0&1&0&1&65\\ 0&0&-1&-1&-52\\ 0&0&-1&1&8 \end{array}\right) \xrightarrow{L_4\leftarrow L_4-L_3} \underset{\text{Étape 3}}{} \left(\begin{array}{cccc|c} 1&1&1&1&215\\ 0&1&0&1&65\\ 0&0&-1&-1&-52\\ 0&0&0&2&60 \end{array}\right).$$

On résout le système triangulaire supérieur ainsi obtenu par remonté :

$$\begin{aligned} 2c=60 &\implies c=30, \\ -j-c=-52 &\implies j=-c+52=22, \\ m+c=65 &\implies m=-c+65=35, \\ p+m+j+c=215 &\implies p=215-m-j-c=128. \end{aligned}$$

Les préférences sont donc : 128 pour Python, 35 pour Matlab, 22 pour Java et 30 pour C.

```
A = [1 1 1 1; 1 1 0 0; 0 1 0 1; 1 0 0 1]
b = [215; 163; 65; 158]
sol = A\b
```

Exercice 3.38 (Système linéaire, existence et unicité)

Considérons le système linéaire de 3 équations en les 3 inconnues x_1, x_2, x_3 suivant :

$$\begin{cases} x_1-x_2+x_3=0 \\ 10x_2+25x_3=90 \\ 20x_1+10x_2=80. \end{cases}$$

Pour résoudre le système linéaire on commence par définir la matrice $\mathbb{A}$ des coefficients du système et le vecteur colonne $\mathbf{b}$ contenant le terme source.

Méthode 1. On calcule la matrice inverse $\mathbb{A}^{-1}$ et on pose $\mathbf{x}=\mathbb{A}^{-1}\mathbf{b}$ (méthode déconseillée).

Méthode 2. On utilise l'opérateur *backslash*.

Méthode 3. On définit la matrice augmentée $[\mathbb{A}|\mathbf{b}]$ et on applique la méthode de GAUSS-JORDAN pour obtenir la forme échelonnée (instruction `rref(Aaug)`.

Dans tous les cas, on teste la solution obtenue en calculant $||\mathbb{A}\mathbf{x}-\mathbf{b}||_2$.

Correction

```
A = [ 1 -1 1; 0 10 25; 20 10 0]
b = [0; 90; 80]

% Methode 1
x = inv(A)*b
norm(A*x-b)
```

```
% while mathematically correct, computing the inverse of a matrix is
% computationally inefficient, and not recommended most of the time.

% Methode 2
x = A\b
norm(A*x-b)

% Methode 3
Aaug=[A b]
RRAaug=rref(Aaug)
x=RRAaug(:,4)
norm(A*x-b)
```

Exercice 3.39 (Système linéaire, non existence)
Considérons le système linéaire de 3 équations en les 3 inconnues x_1, x_2, x_3 suivant :

$$\begin{cases} 3x_1 + 2x_2 + x_3 = 3 \\ 2x_1 + x_2 + x_3 = 0 \\ 6x_1 + 2x_2 + 4x_3 = 6. \end{cases}$$

Pour résoudre le système linéaire on commence par définir la matrice $\mathbb{A}$ des coefficients du système et le vecteur colonne $\mathbf{b}$ contenant le terme source.

1. On définit la matrice augmentée $[\mathbb{A}|\mathbf{b}]$ et on applique la méthode de GAUSS-JORDAN pour obtenir la forme échelonnée (instruction `rref(Aaug)`. Pourquoi peut-on conclure que le système n'a pas de solution ?
2. Octave nous donne malgré tout une solution ! Vérifiez-le avec l'opérateur *backslash*.
3. Que se passe-t-il si on essaye de calculer la matrice inverse $\mathbb{A}^{-1}$ et poser ensuite $\mathbf{x} = \mathbb{A}^{-1}\mathbf{b}$?

Dans tous les cas, on teste la solution obtenue en calculant $||\mathbb{A}\mathbf{x} - \mathbf{b}||_2$.

Correction

```
A = [ 3 2 1; 2 1 1; 6 2 4]
b = [3; 0; 6]

% Point 1
rref([A ,b])
% the last line of this matrix states that 0 = 1. That is not true, which
% means there is no solution.

% Point 2
x = A\b
norm(A*x-b)

% Point 3
invA=inv(A)
x = invA*b
norm(A*x-b)
```

3.3.2 Systèmes linéaires : méthode de Gauss pour des systèmes carrés

Exercice 3.40
Résoudre les systèmes linéaires suivants :

$$\text{①} \begin{cases} x_1 + 2x_2 - x_3 = 2 \\ x_1 - 2x_2 - 3x_3 = -6 \\ x_1 + 4x_2 + 4x_3 = 3 \end{cases} \qquad \text{②} \begin{cases} -x_1 + x_2 + 3x_3 = 12 \\ 2x_1 - x_2 + 2x_3 = -8 \\ 4x_1 + x_2 - 4x_3 = 15 \end{cases} \qquad \text{③} \begin{cases} -2u - 4v + 3w = -1 \\ 2v - w = 1 \\ u + v - 3w = -6 \end{cases}$$

④ $\begin{cases} -2x & -y & & +4t=2 \\ 2x & +3y & +3z & +2t=14 \\ x & +2y & +z & +t=7 \\ -x & & -z & +t=-1 \end{cases}$ ⑤ $\begin{pmatrix} 6 & 1 & 1 \\ 2 & 4 & 0 \\ 1 & 2 & 6 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 12 \\ 0 \\ 6 \end{pmatrix}$ ⑥ $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \\ 3 & 4 & 1 & 2 \\ 4 & 1 & 2 & 3 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 10 \\ 10 \\ 10 \\ 10 \end{pmatrix}$

Correction

On utilise la méthode du pivot de GAUSS :

①

$$\begin{cases} x_1+2x_2 & -x_3 & = 2, \\ x_1-2x_2-3x_3 & = -6, \\ x_1+4x_2+4x_3 & = 3. \end{cases} \xrightarrow[L_3\leftarrow L_3-L_1]{L_2\leftarrow L_2-L_1} \begin{cases} x_1+2x_2 & -x_3 & = 2, \\ -4x_2-2x_3 & = -8, \\ 2x_2+5x_3 & = 1. \end{cases} \xrightarrow{L_3\leftarrow L_3+L_2/2} \begin{cases} x_1+2x_2 & -x_3 & = 2, \\ -4x_2-2x_3 & = -8, \\ 4x_3 & = -3. \end{cases}$$

donc $x_3 = \frac{-3}{4}$, $x_2 = \frac{19}{8}$ et $x_1 = \frac{-7}{2}$.

②

$$\begin{cases} -x_1+x_2+3x_3 = 12 \\ 2x_1-x_2+2x_3 = -8 \\ 4x_1+x_2-4x_3 = 15 \end{cases} \xrightarrow[L_3\leftarrow L_3+4L_1]{L_2\leftarrow L_2+2L_1} \begin{cases} -x_1+x_2+3x_3 = 12 \\ x_2+8x_3 = 16 \\ 5x_2+8x_3 = 63 \end{cases} \xrightarrow{L_3\leftarrow L_3-5L_2} \begin{cases} -x_1+x_2 & +3x_3 & = 12 \\ x_2 & +8x_3 & = 16 \\ & -32x_3 & = -17 \end{cases}$$

donc $x_3 = \frac{17}{32}$, $x_2 = \frac{47}{4}$ et $x_1 = \frac{43}{32}$.

③

$$\begin{cases} -2u-4v+3w=-1 \\ 2v-w=1 \\ u+v-3w=-6 \end{cases} \xrightarrow{L_3\leftarrow L_3+L_1/2} \begin{cases} -2u-4v+3w=-1 \\ 2v-w=1 \\ -v-\frac{3}{2}w=-{}^{13}/_{2} \end{cases} \xrightarrow{L_3\leftarrow L_3+L_2/2} \begin{cases} -2u-4v+3w=-1 \\ 2v-w=1 \\ -2w=-6 \end{cases}$$

donc $w=3$, $v=2$ et $u=1$.

④

$$\begin{cases} -2x & -y & & +4t=2 \\ 2x & +3y & +3z & +2t=14 \\ x & +2y & +z & +t=7 \\ -x & & -z & +t=-1 \end{cases} \xrightarrow[L_4\leftarrow L_4-L_1/2]{\substack{L_2\leftarrow L_2+L_1 \\ L_3\leftarrow L_3+L_1/2}} \begin{cases} -2x-y & & +4t=2 \\ 2y & +3z & +6t=16 \\ \frac{3}{2}y & +z & +3t=8 \\ \frac{1}{2}y & -z & -t=-2 \end{cases}$$

$$\xrightarrow[L_4\leftarrow L_4-L_2/4]{L_3\leftarrow L_3-3L_2/4} \begin{cases} -2x-y & & +4t=2 \\ 2y & +3z & +6t=16 \\ & -\frac{5}{4}z & -\frac{3}{2}t=-4 \\ & -\frac{7}{4}z & -\frac{5}{2}t=-6 \end{cases} \xrightarrow{L_4\leftarrow L_4-7L_3/5} \begin{cases} -2x-y & & +4t=2 \\ 2y & +3z & +6t=16 \\ & -\frac{5}{4}z & -\frac{3}{2}t=-4 \\ & & -\frac{2}{5}t=-\frac{2}{5} \end{cases}$$

donc $t=1$, $z=2$, $y=2$ et $x=0$.

⑤

$$[\mathbb{A}|\mathbf{b}] = \left(\begin{array}{ccc|c} 6 & 1 & 1 & 12 \\ 2 & 4 & 0 & 0 \\ 1 & 2 & 6 & 6 \end{array}\right) \xrightarrow[L_3\leftarrow L_3-\frac{1}{6}L_1]{L_2\leftarrow L_2-\frac{2}{6}L_1} \left(\begin{array}{ccc|c} 6 & 1 & 1 & 12 \\ 0 & \frac{11}{3} & -\frac{1}{3} & -4 \\ 0 & \frac{11}{6} & \frac{35}{6} & 4 \end{array}\right) \xrightarrow{L_3\leftarrow L_3-\frac{\frac{11}{6}}{\frac{11}{3}}L_2} \left(\begin{array}{ccc|c} 6 & 1 & 1 & 12 \\ 0 & \frac{11}{3} & -\frac{1}{3} & -4 \\ 0 & 0 & 6 & 6 \end{array}\right)$$

donc

$$\begin{cases} 6x_1 + x_2 + x_3 = 12, \\ \frac{11}{3}x_2 - \frac{1}{3}x_3 = -4 \\ 6x_3 = 6 \end{cases} \quad \Longrightarrow \quad x_3 = 1, \quad x_2 = -1, \quad x_1 = 2.$$

⑥

$$[\mathbb{A}|\mathbf{b}] = \left(\begin{array}{cccc|c} 1 & 2 & 3 & 4 & 10 \\ 2 & 3 & 4 & 1 & 10 \\ 3 & 4 & 1 & 2 & 10 \\ 4 & 1 & 2 & 3 & 10 \end{array}\right) \xrightarrow[L_4\leftarrow L_4-4L_1]{\substack{L_2\leftarrow L_2-2L_1 \\ L_3\leftarrow L_3-3L_1}} \left(\begin{array}{cccc|c} 1 & 2 & 3 & 4 & 10 \\ 0 & -1 & -2 & -7 & -10 \\ 0 & -2 & -8 & -10 & -20 \\ 0 & -7 & -10 & -13 & -30 \end{array}\right)$$

$$\xrightarrow[L_4\leftarrow L_4-7L_2]{L_3\leftarrow L_3-2L_2} \left(\begin{array}{cccc|c} 1 & 2 & 3 & 4 & 10 \\ 0 & -1 & -2 & -7 & -10 \\ 0 & 0 & -4 & 4 & 0 \\ 0 & 0 & 4 & 36 & 40 \end{array}\right) \xrightarrow{L_4\leftarrow L_4+L_3} \left(\begin{array}{cccc|c} 1 & 2 & 3 & 4 & 10 \\ 0 & -1 & -2 & -7 & -10 \\ 0 & 0 & -4 & 4 & 0 \\ 0 & 0 & 0 & 40 & 40 \end{array}\right)$$

donc

$$\begin{cases} x_1+2x_2+3x_3+4x_4=10 \\ -x_2-2x_3-7x_4=-10 \\ -4x_3+4x_4=0 \\ 40x_4=40 \end{cases} \quad \Longrightarrow \quad x_4=1,\quad x_3=1,\quad x_2=1,\quad x_1=1.$$

```
A = [1 2 -1; 1 -2 -3; 1 4 4]
b = [2; -6; 3]
A\b
A = [-1 1 3; 2 -1 2; 4 1 -4]
b = [12; -8; 15]
A\b
A = [-2 -4 3; 0 2 -1; 1 1 -3]
b = [-1; 1; -6]
A\b
```

```
A = [-2 -1 0 4; 2 3 3 2; 1 2 1 1; -1 0 -1 1]
b = [2; 14; 7; -1]
A\b
A = [6 1 1; 2 4 0; 1 2 6]
b = [12; 0; 6]
A\b
A = [1 2 3 4; 2 3 4 1; 3 4 1 2; 4 1 2 3]
b = [10; 10; 10; 10]
A\b
```

Exercice 3.41

Résoudre le système linéaire $\mathbb{A}\mathbf{x}=\mathbf{b}$ avec

$$\mathbb{A}=\begin{pmatrix} 1 & -1 \\ 1+\varepsilon & -1 \end{pmatrix}, \qquad \mathbf{b}=\begin{pmatrix} 0 \\ \varepsilon \end{pmatrix}.$$

Que se passe-t-il si $\varepsilon=0$?

Correction

Pour $\varepsilon\neq 0$

$$[\mathbb{A}|\mathbf{b}]=\left(\begin{array}{cc|c} 1 & -1 & 0 \\ 1+\varepsilon & -1 & \varepsilon \end{array}\right) \xrightarrow{L_2\leftarrow L_2-(1+\varepsilon)L_1} \left(\begin{array}{cc|c} 1 & -1 & 0 \\ 0 & \varepsilon & \varepsilon \end{array}\right)$$

donc

$$\begin{cases} x-y=0 \\ \varepsilon y=\varepsilon, \end{cases} \quad \Longrightarrow \quad x=y=1.$$

Le couple $x=y=1$ est bien la seule solution du système pour tout $\varepsilon\neq 0$.
Si $\varepsilon=0$, le système admet une infinité de solutions de la forme (κ,κ).
Géométriquement, ce système peut être vu comme la recherche du point d'intersection entre deux droites de pente respectivement 1 et $(1+\varepsilon)$. Cependant, si ε est petit, les deux droites sont "presque" parallèles et il est difficile numériquement de trouver le point d'intersection.

```
for i = [-10:-1:-16]
  a = 10^i
  [1 -1;1+a -1] \ [0;a]
end
```

Exercice 3.42

Soit le système linéaire

$$(S)\quad \begin{cases} 2x_1-x_2-3x_3=0, \\ -x_1+2x_3=0, \\ 2x_1-3x_2-x_3=0. \end{cases}$$

Ce système est-il compatible ? Possède-t-il une solution unique ?

Correction

$$\begin{cases} 2x_1-x_2-3x_3=0, \\ -x_1+2x_3=0, \\ 2x_1-3x_2-x_3=0. \end{cases} \xrightarrow[L_3\leftarrow L_3-L_1]{L_2\leftarrow L_2+L_1/2} \begin{cases} 2x_1-x_2-3x_3=0, \\ -{}^1\!/_2x_2+{}^1\!/_2x_3=0, \\ -2x_2+2x_3=0, \end{cases} \xrightarrow{L_3\leftarrow L_3-4L_2} \begin{cases} 2x_1-x_2-3x_3=0, \\ -{}^1\!/_2x_2+{}^1\!/_2x_3=0, \\ 0=0, \end{cases}$$

Le système est compatible car le rang du système est 2 inférieur au nombre d'inconnues 3 et la solution n'est pas unique car $\mathrm{rg}(S) < 3$. Il admet une infinité de solutions de la forme $(2\kappa, \kappa, \kappa)$, $\kappa \in \mathbb{R}$.
Que dit Octave ?

```
>> [2,-1,-3;-1,0,2;2,-3,-1]\[0;0;0]
warning: matrix singular to machine precision
```

Exercice 3.43
Trouver toutes les solutions du système linéaire homogène

$$(S) \quad \begin{cases} -3x_1 + x_2 + 2x_3 = 0, \\ -2x_1 + 2x_3 = 0, \\ -11x_1 + 6x_2 + 5x_3 = 0. \end{cases}$$

Correction
Le système étant homogène, il est inutile d'écrire le terme source dans la méthode du pivot de GAUSS :

$$\mathbb{A} = \begin{pmatrix} -3 & 1 & 2 \\ -2 & 0 & 2 \\ -11 & 6 & 5 \end{pmatrix} \xrightarrow[L_3 \leftarrow L_3 - 11L_1/3]{L_2 \leftarrow L_2 - 2L_1/3} \begin{pmatrix} -3 & 1 & 2 \\ 0 & -2/3 & 2/3 \\ 0 & 7/3 & -7/3 \end{pmatrix} \xrightarrow{L_3 \leftarrow L_3 + 7L_2/2} \begin{pmatrix} -3 & 1 & 2 \\ 0 & -2/3 & 2/3 \\ 0 & 0 & 0 \end{pmatrix}$$

Le système admet une infinité de solutions de la forme (κ, κ, κ) avec $\kappa \in \mathbb{R}$.

Exercice 3.44
Équilibrer la réaction

$$\mathrm{Zn} + \mathrm{H_2SO_4} + \mathrm{K_2Cr_2O_7} \rightarrow \mathrm{KCr(SO_4)_2} + \mathrm{ZnSO_4} + \mathrm{H_2O}.$$

Correction
Écrivons les coefficients stœchiométriques et les contraintes :

$$x_1\mathrm{Zn} + x_2\mathrm{H_2SO_4} + x_3\mathrm{K_2Cr_2O_7} \rightarrow x_4\mathrm{KCr(SO_4)_2} + x_5\mathrm{ZnSO_4} + x_6\mathrm{H_2O}$$

1. Atomes de Zn : $x_1 = x_5$, *i.e.* $x_1 - x_5 = 0$
2. Atomes de H : $2x_2 = 2x_6$, *i.e.* $x_2 - x_6 = 0$
3. Atomes de S : $x_2 = 2x_4 + x_5$, *i.e.* $x_2 - 2x_4 - x_5 = 0$
4. Atomes de K : $2x_3 = x_4$, *i.e.* $2x_3 - x_4 = 0$
5. Atomes de Cr : $2x_3 = x_4$, *i.e.* $2x_3 - x_4 = 0$
6. Atomes de O : $4x_2 + 7x_3 = 8x_4 + 4x_5 + x_6$, *i.e.* $4x_2 + 7x_3 - 8x_4 - 4x_5 - x_6 = 0$

Notons que la contrainte $2x_3 - x_4 = 0$ est répétée deux fois, donc on ne l'écrira qu'une seule fois dans le système linéaire ; cela donne 5 équations pour 6 inconnues. Fixons arbitrairement un des coefficients, par exemple $x_6 = 1$; on obtient alors le système linéaire

$$\begin{pmatrix} 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 & -1 \\ 0 & 0 & 2 & -1 & 0 \\ 0 & 4 & 7 & -8 & -4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

ce qui donne

$$\left(\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & -2 & -1 & 0 \\ 0 & 0 & 2 & -1 & 0 & 0 \\ 0 & 4 & 7 & -8 & -4 & 1 \end{array}\right) \xrightarrow[L_5 \leftarrow L_5 - 4L_2]{L_3 \leftarrow L_3 - L_2} \left(\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -2 & -1 & -1 \\ 0 & 0 & 2 & -1 & 0 & 0 \\ 0 & 0 & 7 & -8 & -4 & -3 \end{array}\right) \xrightarrow{L_3 \leftrightarrow L_4} \left(\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1 & -1 \\ 0 & 0 & 7 & -8 & -4 & -3 \end{array}\right)$$

$$\xrightarrow{L_5 \leftarrow L_5 - \frac{7}{2}L_3} \left(\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1 & -1 \\ 0 & 0 & 0 & -\frac{9}{2} & -4 & -3 \end{array}\right) \xrightarrow{L_5 \leftarrow L_5 - \frac{9}{4}L_4} \left(\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1 & -1 \\ 0 & 0 & 0 & 0 & -\frac{7}{4} & -\frac{3}{4} \end{array}\right)$$

dont la solution est bien

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 3/7 \\ 1 \\ 1/7 \\ 2/7 \\ 3/7 \end{pmatrix}$$

Si on multiplie tous les coefficients par 7 on obtient

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ 1 \\ 2 \\ 3 \\ 7 \end{pmatrix}$$

et donc la réaction équilibrée

$$3\mathrm{Zn} + 7\mathrm{H_2SO_4} + \mathrm{K_2Cr_2O_7} \to 2\mathrm{KCr(SO_4)_2} + 3\mathrm{ZnSO_4} + 7\mathrm{H_2O}.$$

Exercice 3.45 (V. GUIRARDEL)
Vous projetez de passer un concours de recrutement l'an prochain. Vous avez sous les yeux le tableau de notes suivant :

CANDIDAT	Mathématique	Anglais	Informatique	Moyenne
QUI	7	12	6	8
QUO	11	6	10	9
QUA	11	16	14	14

Retrouver les coefficients de chaque épreuve. La solution est-elle unique ?

Correction
Il s'agit de trouver les trois coefficients $m, a, i \in [0;1]$ tels que

$$\begin{cases} 7m + 12a + 6i = 8, \\ 11m + 6a + 10i = 9, \\ 11m + 16a + 14i = 14. \end{cases}$$

Utilisons la méthode de GAUSS :

$$\begin{cases} 7m+12a+6i=8, \\ 11m+6a+10i=9, \\ 11m+16a+14i=14, \end{cases} \xrightarrow[L_3 \leftarrow L_3 - \frac{11}{7}L_1]{L_2 \leftarrow L_2 - \frac{11}{7}L_1} \begin{cases} 7m+12a+6i=8, \\ -\frac{90}{7}a+\frac{4}{7}i=-\frac{25}{7}, \\ -\frac{20}{7}a+\frac{32}{7}i=\frac{10}{7}, \end{cases} \xrightarrow{L_3 \leftarrow L_3 - \frac{2}{9}L_2} \begin{cases} 7m+12a+6i=8, \\ -\frac{90}{7}a+\frac{4}{7}i=-\frac{25}{7}, \\ \frac{40}{9}i=\frac{20}{9}, \end{cases}$$

qui admet l'unique solution $(0.2, 0.3, 0.5)$.

Une autre interprétation est la suivante : il s'agit de trouver les trois coefficients $m, a, i \in [0;1]$ tels que

$$\begin{cases} 7m + 12a + 6i = 8(m + a + i) \\ 11m + 6a + 10i = 9(m + a + i), \\ 11m + 16a + 14i = 14(m + a + i). \end{cases}$$

Utilisons la méthode de GAUSS :

$$\begin{cases} -m+4a-2i=0, \\ 2m-3a+i=0, \\ -3m+2a=0, \end{cases} \xrightarrow[L_3 \leftarrow L_3 - \frac{11}{7}L_1]{L_2 \leftarrow L_2 - \frac{11}{7}L_1} \begin{cases} -m+4a-2i=0, \\ 5a-3i=0, \\ -10a+6i=0, \end{cases} \xrightarrow{L_3 \leftarrow L_3 - \frac{2}{9}L_2} \begin{cases} -m+4a-2i=0, \\ 5a-3i=0, \\ 0=0, \end{cases}$$

qui admet une infinité de solutions de la forme $(2\kappa, 3\kappa, 5\kappa)$ avec $\kappa \in [0; 1/5]$.

Exercice 3.46 (V. GUIRARDEL)

Une entreprise fabrique des manteaux. Ces manteaux sont composés de tissu rouge, de tissu bleu et d'une doublure noire. Le tableau suivant résume les mètres carrés de chaque tissu nécessaires à la confection du manteau en tailles S, M, L et XL :

	S	M	L	XL
Tissu rouge	0.4	0.5	0.6	0.7
Tissu bleu	1	1.1	1.2	1.3
Doublure	1.5	1.7	1.9	2.1

Chaque tissu est tissé à l'aide de plusieurs types de fil : coton, polyester et polyamide. Le tableau suivant résume les mètres de fil de chaque type nécessaires par mètre carré de tissu :

	Tissu rouge	Tissu bleu	Doublure
Coton	500	400	1000
Polyamide	1000	900	700
Polyester	500	600	0

1. L'entreprise veut produire s manteaux taille S, m manteaux taille M, ℓ manteaux taille L et x manteaux taille XL. Quelle quantité de fil de chaque catégorie doit-elle commander? Répondre à cette question dans le langage des matrices.
2. En fin d'année, l'entreprise veut écouler entièrement ses stocks de fils. Il lui reste 100 000 m de coton et de polyamide, et 20 000 m de Polyester. Peut-elle transformer entièrement ses stocks de fils en manteaux?

Correction

Introduisons les deux matrices $\mathbb{A}$ et $\mathbb{B}$ et les deux vecteurs $\mathbf{u}$ et $\mathbf{v}$ suivants

$$\mathbb{A} = \begin{pmatrix} 0.4 & 0.5 & 0.6 & 0.7 \\ 1 & 1.1 & 1.2 & 1.3 \\ 1.5 & 1.7 & 1.9 & 2.1 \end{pmatrix} \qquad \mathbb{B} = \begin{pmatrix} 500 & 400 & 1000 \\ 1000 & 900 & 700 \\ 500 & 600 & 0 \end{pmatrix} \qquad \mathbf{u} = \begin{pmatrix} s \\ m \\ \ell \\ x \end{pmatrix} \qquad \mathbf{v} = \begin{pmatrix} c \\ a \\ e \end{pmatrix}$$

1. Pour produire s manteaux taille S, m manteaux taille M, ℓ manteaux taille L et x manteaux taille XL, l'entreprise doit commander c mètres de coton, a mètres de polyamide et e mètres de polyester où c, a, e sont les entrées du vecteur $\mathbf{v}$ suivant :
$$\mathbf{v} = \mathbb{B}\mathbb{A}\mathbf{u} = \begin{pmatrix} 2100s + 2390m + 2680\ell + 2970x \\ 2350s + 2680m + 3010\ell + 3340x \\ 800s + 910m + 1020\ell + 1130x \end{pmatrix}.$$
2. On cherche s'il existe un vecteur $\mathbf{u}$ tel que
$$\begin{pmatrix} 100000 \\ 100000 \\ 20000 \end{pmatrix} = \mathbb{B}\mathbb{A}\mathbf{u},$$
i.e. s'il existe une solution du système linéaire
$$\begin{pmatrix} 2100 & 2390 & 2680 & 2970 \\ 2350 & 2680 & 3010 & 3340 \\ 800 & 910 & 1020 & 1130 \end{pmatrix} \begin{pmatrix} s \\ m \\ \ell \\ x \end{pmatrix} = \begin{pmatrix} 100000 \\ 100000 \\ 20000 \end{pmatrix}.$$
En appliquant la méthode de GAUSS on obtient le système
$$\begin{pmatrix} 2100 & 2390 & 2680 & 2970 \\ 0 & \frac{115}{21} & \frac{230}{21} & \frac{115}{7} \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} s \\ m \\ \ell \\ x \end{pmatrix} = \begin{pmatrix} 100000 \\ -\frac{250000}{21} \\ -\frac{440000}{23} \end{pmatrix}$$
qui n'admet pas de solution.

Exercice 3.47
Soit le système linéaire

$$(S) \quad \begin{cases} x - \alpha y = 1, \\ \alpha x - y = 1. \end{cases}$$

Déterminer les valeurs de α de telle sorte que ce système possède :
1. une infinité de solutions ;
2. aucune solution ;
3. une solution unique.

Correction

$$\left(\begin{array}{cc|c} 1 & -\alpha & 1 \\ \alpha & -1 & 1 \end{array}\right) \xrightarrow{L_2 \leftarrow L_2 - \alpha L_1} \left(\begin{array}{cc|c} 1 & -\alpha & 1 \\ 0 & -1+\alpha^2 & 1-\alpha \end{array}\right).$$

Comme $-1+\alpha^2 = (\alpha-1)(\alpha+1)$ on conclut que

1. si $\alpha = 1$ (*i.e.* la dernière équation correspond à $0 = 0$) alors (S) possède une infinité de solutions,
2. si $\alpha = -1$ (*i.e.* la dernière équation correspond à $0 = 2$) alors (S) ne possède aucune solution,
3. si $\alpha \notin \{-1; 1\}$ alors (S) possède une solution unique $x = \frac{1}{\alpha+1}$ et $y = -\frac{1}{\alpha+1}$.

Exercice 3.48
Soit le système linéaire

$$(S) \quad \begin{cases} x + \alpha y = 1, \\ -\alpha x - y = 1. \end{cases}$$

En utilisant le pivot de GAUSS, déterminer les valeurs de $\alpha \in \mathbb{R}$ de telle sorte que ce système possède :
a) une infinité de solutions ;
b) aucune solution ;
c) une solution unique.

Correction

$$\left\{\begin{array}{rr} x+\alpha y=1 \\ -\alpha x \quad -y=1 \end{array}\right. \xrightarrow{L_2 \leftarrow L_2 + \alpha L_1} \left\{\begin{array}{rr} x \quad +\alpha y=1 \\ (-1+\alpha^2)y=\alpha+1 \end{array}\right.$$

Comme $-1+\alpha^2 = (\alpha-1)(\alpha+1)$ on conclut que

a) si $\alpha = -1$ (*i.e.* la dernière équation correspond à $0 = 0$) alors (S) possède une infinité de solutions,

b) si $\alpha = 1$ (*i.e.* la dernière équation correspond à $0 = 2$) alors (S) ne possède aucune solution,

c) si $\alpha \notin \{-1; 1\}$ alors (S) possède une solution unique $x = -\frac{1}{\alpha-1}$ et $y = \frac{1}{\alpha-1}$.

Exercice 3.49
Soit le système linéaire

$$(S) \quad \begin{cases} x + y - z = 1, \\ 2x + 3y + \beta z = 3, \\ x + \beta y + 3z = -3. \end{cases}$$

Déterminer les valeurs de β de telle sorte que ce système possède :
1. une infinité de solutions ;
2. aucune solution ;
3. une solution unique.

Correction

$$\left(\begin{array}{ccc|c} 1 & 1 & -1 & 1 \\ 2 & 3 & \beta & 3 \\ 1 & \beta & 3 & -3 \end{array}\right) \xrightarrow[L_3 \leftarrow L_3 - L_1]{L_2 \leftarrow L_2 - 2L_1} \left(\begin{array}{ccc|c} 1 & 1 & -1 & 1 \\ 0 & 1 & \beta+2 & 1 \\ 0 & \beta-1 & 4 & -4 \end{array}\right) \xrightarrow{L_3 \leftarrow L_3 + (1-\beta)L_2} \left(\begin{array}{ccc|c} 1 & 1 & -1 & 1 \\ 0 & 1 & \beta+2 & 1 \\ 0 & 0 & (6-\beta-\beta^2) & -(3+\beta) \end{array}\right).$$

Comme $6-\beta-\beta^2 = (2-\beta)(3+\beta)$ on conclut que

1. si $\beta = -3$ (*i.e.* la dernière équation correspond à $0z = 0$) alors (S) possède une infinité de solutions,
2. si $\beta = 2$ (*i.e.* la dernière équation correspond à $0z = -5$) alors (S) ne possède aucune solution,
3. si $\beta \notin \{2; -3\}$ alors (S) possède une solution unique.

Exercice 3.50

Trouver les valeurs de $\kappa \in \mathbb{R}$ pour lesquelles le système suivant a un nombre respectivement fini et infini de solutions :

$$\begin{cases} 2x_1 - x_2 = \kappa, \\ x_1 - x_2 - x_3 = 0, \\ x_1 - \kappa x_2 + \kappa x_3 = \kappa. \end{cases}$$

Correction

$$\left(\begin{array}{ccc|c} 2 & -1 & 0 & \kappa \\ 1 & -1 & -1 & 0 \\ 1 & -\kappa & \kappa & \kappa \end{array}\right) \xrightarrow[L_3 \leftarrow L_3 - L_1/2]{L_2 \leftarrow L_2 - L_1/2} \left(\begin{array}{ccc|c} 2 & -1 & 0 & \kappa \\ 0 & -1/2 & -1 & -\kappa/2 \\ 0 & -\kappa+1/2 & \kappa & \kappa/2 \end{array}\right) \xrightarrow{L_3 \leftarrow L_3 + (1-2\kappa)L_2} \left(\begin{array}{ccc|c} 2 & -1 & 0 & \kappa \\ 0 & -1/2 & -1 & -\kappa/2 \\ 0 & 0 & 3\kappa-1 & \kappa^2 \end{array}\right).$$

On conclut que

1. si $\kappa = \frac{1}{3}$ alors (S) ne possède aucune solution,
2. si $\kappa \neq \frac{1}{3}$ alors (S) possède une solution unique donnée par $x_3 = \frac{\kappa^2}{3\kappa-1}$, $x_2 = \frac{-\kappa/2+x_3}{-1/2} = \frac{\kappa(\kappa-1)}{3\kappa-1}$ et $x_1 = \frac{\kappa+x_2}{2} = \frac{\kappa(2\kappa-1)}{3\kappa-1}$,
3. il n'existe aucune valeur de κ pour que (S) possède une infinité de solutions.

Exercice 3.51

Résoudre le système linéaire en discutant suivant la valeur du paramètre $a \in \mathbb{R}$:

$$\begin{cases} x + 2y + 3z = 2, \\ x - y + 2z = 7, \\ 3x + az = 10. \end{cases}$$

Correction
Si on utilise la méthode de GAUSS on trouve

$$[\mathbb{A}|\mathbf{b}] = \left(\begin{array}{ccc|c} 1 & 2 & 3 & 2 \\ 1 & -1 & 2 & 7 \\ 3 & 0 & a & 10 \end{array}\right) \xrightarrow[L_3 \leftarrow L_3 - 3L_1]{L_2 \leftarrow L_2 - L_1} \left(\begin{array}{ccc|c} 1 & 2 & 3 & 2 \\ 0 & -3 & -1 & 5 \\ 0 & -6 & a-9 & 4 \end{array}\right) \xrightarrow{L_3 \leftarrow L_3 - 2L_2} \left(\begin{array}{ccc|c} 1 & 2 & 3 & 2 \\ 0 & -3 & -1 & 5 \\ 0 & 0 & a-7 & -6 \end{array}\right).$$

On a ainsi transformé le système linéaire initial dans le système linéaire triangulaire supérieur équivalent

$$\begin{cases} x + 2y + 3z = 2, \\ -3y - z = 5, \\ (a-7)z = -6. \end{cases}$$

Par conséquent,

- ⋆ si $a \neq 7$, $z = \dfrac{-6}{a-7}$, $y = \dfrac{5+z}{-3} = \dfrac{5a-41}{-3(a-7)}$ et $x = 2-2y-3z = \dfrac{2(8a-35)}{3(a-7)}$ est l'unique solution du système linéaire ;
- ⋆ si $a = 7$ il n'y a pas de solutions du système linéaire.

Exercice 3.52
En utilisant la méthode de GAUSS, résoudre le système linéaire en discutant suivant la valeur du paramètre $a \in \mathbb{R}$:

$$\begin{cases} x-y+2z=7, \\ x+2y+3z=2, \\ 3x+az=10. \end{cases}$$

Correction

$$\left(\begin{array}{ccc|c} 1 & -1 & 2 & 7 \\ 1 & 2 & 3 & 2 \\ 3 & 0 & a & 10 \end{array}\right) \xrightarrow[L_3 \leftarrow L_3-3L_1]{L_2 \leftarrow L_2-L_1} \left(\begin{array}{ccc|c} 1 & -1 & 2 & 7 \\ 0 & 3 & 1 & -5 \\ 0 & 3 & a-6 & -11 \end{array}\right) \xrightarrow{L_3 \leftarrow L_3-L_2} \left(\begin{array}{ccc|c} 1 & -1 & 2 & 7 \\ 0 & 3 & 1 & -5 \\ 0 & 0 & a-7 & -6 \end{array}\right).$$

On a ainsi transformé le système linéaire initial dans le système linéaire triangulaire supérieur équivalent

$$\begin{cases} x-y+2z=7, \\ 3y+z=-5, \\ (a-7)z=-6. \end{cases}$$

Par conséquent,

- ⋆ si $a \neq 7$, $z=\dfrac{-6}{a-7}$, $y=\dfrac{-5-z}{3}=\dfrac{-5a+41}{3(a-7)}$ et $x=7-y-2z=\dfrac{2(8a-35)}{3(a-7)}$ est l'unique solution du système linéaire ;
- ⋆ si $a=7$ il n'y a pas de solutions du système linéaire.

Exercice 3.53
En utilisant la méthode du pivot de GAUSS, résoudre le système linéaire en discutant suivant la valeur du paramètre $a \in \mathbb{R}$:

$$\begin{cases} x+z+w=0, \\ ax+y+(a-1)z+w=0, \\ 2x+ay+z+2w=0, \\ x-y+2z+aw=0. \end{cases}$$

Correction
Il s'agit d'un système homogène, il est alors inutile d'écrire le terme source dans la méthode du pivot de GAUSS. En appliquant cette méthode on obtient

$$\begin{pmatrix} 1 & 0 & 1 & 1 \\ a & 1 & a-1 & 1 \\ 2 & a & 1 & 2 \\ 1 & -1 & 2 & a \end{pmatrix} \xrightarrow[L_4 \leftarrow L_4-L_1]{\substack{L_2 \leftarrow L_2-aL_1 \\ L_3 \leftarrow L_3-2L_1}} \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & -1 & 1-a \\ 0 & a & -1 & 0 \\ 0 & -1 & 1 & a-1 \end{pmatrix} \xrightarrow[L_4 \leftarrow L_4+L_2]{\substack{L_3 \leftarrow L_3-aL_2 \\ }} \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & -1 & 1-a \\ 0 & 0 & a-1 & a(a-1) \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

On a ainsi transformé le système linéaire initial dans le système linéaire triangulaire supérieur équivalent

$$\begin{cases} x+z+w=0, \\ y-z+(1-a)w=0, \\ (a-1)z+a(a-1)w=0, \\ 0=0. \end{cases}$$

Par conséquent, si on pose $w=\kappa_1 \in \mathbb{R}$ une constante réelle quelconque, alors

- ⋆ si $a \neq 1$, $z=\frac{-a(a-1)w}{a-1}=-a\kappa_1$, $y=-(1-a)w+z=-\kappa_1$ et $x=-w-z=(a-1)\kappa_1$: tous les vecteurs de $\mathrm{Vect}\{(a-1,-1,-a,1)\}$ sont solution du système linéaire ;
- ⋆ si $a=1$, on pose $z=\kappa_2 \in \mathbb{R}$ une constante réelle quelconque et on a $y=-(1-a)w+z=-(1-a)\kappa_1+\kappa_2$ et $x=-w-z=-\kappa_1-\kappa_2$: tous les vecteurs de $\mathrm{Vect}\{(-1,1,1,0),(-1,0,0,1)\}$ sont solution du système linéaire.

Exercice 3.54
En utilisant la méthode du pivot de GAUSS, résoudre le système linéaire en discutant suivant la valeur du paramètre $b \in \mathbb{R}$:

$$\begin{cases} x+z+w=0, \\ (b+1)x+y+bz+w=0, \\ 2x+(b+1)y+z+2w=0, \\ x-y+2z+(b+1)w=0. \end{cases}$$

Correction
Il s'agit d'un système homogène, il est alors inutile d'écrire le terme source dans la méthode du pivot de GAUSS. En appliquant cette méthode on obtient

$$\begin{pmatrix} 1 & 0 & 1 & 1 \\ b+1 & 1 & b & 1 \\ 2 & b+1 & 1 & 2 \\ 1 & -1 & 2 & b+1 \end{pmatrix} \xrightarrow[L_4 \leftarrow L_4 - L_1]{\substack{L_2 \leftarrow L_2-(b+1)L_1 \\ L_3 \leftarrow L_3-2L_1}} \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & -1 & -b \\ 0 & b+1 & -1 & 0 \\ 0 & -1 & 1 & b \end{pmatrix} \xrightarrow[L_4 \leftarrow L_4+L_2]{L_3 \leftarrow L_3-(b+1)L_2} \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & -1 & -b \\ 0 & 0 & b & b(b+1) \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

On a ainsi transformé le système linéaire initial dans le système linéaire triangulaire supérieur équivalent

$$\begin{cases} x+z+w=0, \\ y-z-bw=0, \\ bz+b(b+1)w=0, \\ 0=0. \end{cases}$$

Par conséquent, si on pose $w=\kappa_1 \in \mathbb{R}$ une constante réelle quelconque, alors

- ⋆ si $b \neq 0$,
$$z=\frac{-b(b+1)w}{b}=-(b+1)\kappa_1, \qquad y=bw+z=-\kappa_1, \qquad x=-w-z=b\kappa_1;$$
tous les vecteurs de $\mathrm{Vect}\{(b+1,1,b+1,-1)\}$ sont solution du système linéaire ;
- ⋆ si $b=0$, on pose $z=\kappa_2 \in \mathbb{R}$ une constante réelle quelconque et on a $y=bw+z=\kappa_2$ et $x=-w-z=-\kappa_1-\kappa_2$: tous les vecteurs de $\mathrm{Vect}\{(-1,1,1,0),(-1,0,0,1)\}$ sont solution du système linéaire.

Exercice 3.55
Discuter et résoudre le système

$$(S_a) \quad \begin{cases} (1+a)x+y+z=0, \\ x+(1+a)y+z=0, \\ x+y+(1+a)z=0, \end{cases}$$

d'inconnue $(x,y,z) \in \mathbb{R}^3$ et de paramètre $a \in \mathbb{R}$.

Correction
Comme le système contient un paramètre, on commence par calculer le déterminant de la matrice associée :

$$\begin{vmatrix} 1+a & 1 & 1 \\ 1 & 1+a & 1 \\ 1 & 1 & 1+a \end{vmatrix} = (1+a)^3+1+1-(1+a)-(1+a)-(1+a) = (1+a)^3-3(1+a)+2$$
$$=\Big((1+a)-1\Big)\Big((1+a)^2+(1+a)-2\Big)=\Big((1+a)-1\Big)\Big((1+a)+2\Big)\Big((1+a)-1\Big)=a^2(3+a).$$

Le système a une et une seule solution si et seulement si ce déterminant est non nul, donc

$$(S_a) \text{ a une et une seule solution si et seulement } a \in \mathbb{R}\setminus\{-3,0\}.$$

Notons $\mathscr{S}$ l'ensemble des solutions.

- ⋆ *Étude du cas* $a=-3$. Le système s'écrit

$$(S_{-3}) \quad \begin{cases} -2x+y+z=0, \\ x-2y+z=0, \\ x+y-2z=0, \end{cases}$$

On utilise la méthode du pivot de GAUSS :

$$\begin{cases} -2x & y & +z=0 \\ x & -2y & +z=0 \\ x & y & -2z=0 \end{cases} \xrightarrow[L_3 \leftarrow L_3+L_1/2]{L_2 \leftarrow L_2+L_1/2} \begin{cases} -2x & y & +z=0 \\ & -\frac{3}{2}y & +\frac{3}{2}z=0 \\ & \frac{3}{2}y & -\frac{3}{2}z=0 \end{cases} \xrightarrow{L_3 \leftarrow L_3+L_2} \begin{cases} -2x & -y & +z=0 \\ & -\frac{5}{2}y & +\frac{3}{2}z=0 \\ & & 0z=0 \end{cases}$$

donc $z=\kappa \in \mathbb{R}$, $y=z$ et $x=z$, ainsi

$$\mathscr{S} = \{(\kappa,\kappa,\kappa) \mid \kappa \in \mathbb{R}\}.$$

⋆ *Étude du cas* $a=0$. Le système s'écrit

$$(S_0) \quad \begin{cases} x+y+z=0, \\ x+y+z=0, \\ x+y+z=0, \end{cases}$$

donc $z=\kappa_1 \in \mathbb{R}$, $y=\kappa_2 \in \mathbb{R}$ et $x=-\kappa_1-\kappa_2$, ainsi

$$\mathscr{S} = \left\{(-\kappa_1-\kappa_2,\kappa_2,\kappa_1) \;\middle|\; (\kappa_1,\kappa_2) \in \mathbb{R}^2\right\}.$$

⋆ *Étude du cas* $a \in \mathbb{R} \setminus \{-3,0\}$. Il s'agit d'un système de Cramer homogène, donc l'unique solution est $(0,0,0)$:

$$\mathscr{S} = \{(0,0,0)\}.$$

Exercice 3.56

Discuter et résoudre le système

$$(S_a) \quad \begin{cases} x+ay+(a-1)z=0, \\ 3x+2y+az=3, \\ (a-1)x+ay+(a+1)z=a, \end{cases}$$

d'inconnue $(x,y,z) \in \mathbb{R}^3$ et de paramètre $a \in \mathbb{R}$.

Correction

Comme le système contient un paramètre, on commence par calculer le déterminant de la matrice associée :

$$\begin{vmatrix} 1 & a & a-1 \\ 3 & 2 & a \\ a-1 & a & a+1 \end{vmatrix} = 2(a+1)+a^2(a-1)+3a(a-1)-2(a-1)^2-a^2-3a(a+1) = a^2(a-4).$$

Le système a une et une seule solution si et seulement si ce déterminant est non nul, donc

$$(S_a) \text{ a une et une seule solution si et seulement } a \in \mathbb{R} \setminus \{0,4\}.$$

Notons $\mathscr{S}$ l'ensemble des solutions.

⋆ *Étude du cas* $a=0$. Le système s'écrit

$$(S_0) \quad \begin{cases} x-z=0, \\ 3x+2y=3, \\ -x+z=0, \end{cases}$$

donc $z=\kappa \in \mathbb{R}$, $y=\frac{3-3\kappa}{2}$ et $x=\kappa$, ainsi

$$\mathscr{S} = \left\{\left(\kappa, \frac{3-3\kappa}{2}, \kappa\right) \;\middle|\; \kappa \in \mathbb{R}\right\}.$$

⋆ *Étude du cas* $a=4$. Le système s'écrit

$$(S_4) \quad \begin{cases} x+4y+3z=0, \\ 3x+2y+4z=3, \\ 3x+4y+5z=4, \end{cases}$$

On utilise la méthode du pivot de GAUSS :

$$\begin{cases} x & +4y & +3z=0, \\ 3x & +2y & +4z=3, \\ 3x & +4y & +5z=4, \end{cases} \xrightarrow[L_3 \leftarrow L_3-3L_1]{L_2 \leftarrow L_2-3L_1} \begin{cases} x & +4y & +3z=0, \\ & -10y & -5z=3, \\ & -8y & -4z=4, \end{cases} \xrightarrow{L_3 \leftarrow 10L_3-8L_2} \begin{cases} x & +4y & +3z=0, \\ & -10y & -5z=3, \\ & & 0=16. \end{cases}$$

La dernière équation est impossible donc

$$\mathscr{S} = \emptyset.$$

⋆ *Étude du cas* $a \in \mathbb{R} \setminus \{-3, 0\}$. On utilise la méthode du pivot de GAUSS :

$$\begin{cases} x+ay+(a-1)z=0, \\ 3x+2y+az=3, \\ (a-1)x+ay+(a+1)z=a, \end{cases} \xrightarrow[L_3 \leftarrow L_3-(a-1)L_1]{L_2 \leftarrow L_2-3L_1} \begin{cases} x+ay+(a-1)z=0, \\ (2-3a)y+(3-2a)z=3, \\ (2-a)ay+(3-a)az=a, \end{cases}$$

$$\xrightarrow{L_3 \leftarrow L_3-\frac{(2-a)a}{(2-3a)}L_2} \begin{cases} x+ay+(a-1)z=0, \\ (2-3a)y+(3-2a)z=3, \\ -\frac{a^2(a-4)}{3a-2}z=\frac{4a}{3a-2}. \end{cases}$$

On obtient $z=-\frac{4}{a(a-4)}$, $y=-\frac{a-6}{a(a-4)}$, $x=\frac{a^2-2a-4}{a(a-4)}$, ainsi

$$\mathscr{S} = \left\{ \left(\frac{a^2-2a-4}{a(a-4)}, -\frac{a-6}{a(a-4)}, -\frac{4}{a(a-4)} \right) \right\}.$$

3.3.3 Calcul de la matrice inverse

Exercice 3.57
Calculer $\mathbb{A}^{-1}$ où $\mathbb{A}$ est la matrice $\begin{pmatrix} 1 & 0 & -1 \\ 4 & -1 & -2 \\ -2 & 0 & 1 \end{pmatrix}$.

Correction

$$[\mathbb{A}|\mathbb{I}_3] = \left(\begin{array}{ccc|ccc} 1 & 0 & -1 & 1 & 0 & 0 \\ 4 & -1 & -2 & 0 & 1 & 0 \\ -2 & 0 & 1 & 0 & 0 & 1 \end{array}\right) \xrightarrow[L_3 \leftarrow L_3+2L_1]{L_1 \leftarrow L_1 \\ L_2 \leftarrow L_2-4L_1} \left(\begin{array}{ccc|ccc} 1 & 0 & -1 & 1 & 0 & 0 \\ 0 & -1 & 2 & -4 & 1 & 0 \\ 0 & 0 & -1 & -2 & 0 & 1 \end{array}\right)$$

$$\xrightarrow[L_3 \leftarrow L_3]{L_1 \leftarrow L_1 \\ L_2 \leftarrow -L_2} \left(\begin{array}{ccc|ccc} 1 & 0 & -1 & 1 & 0 & 0 \\ 0 & 1 & -2 & 4 & -1 & 0 \\ 0 & 0 & -1 & -2 & 0 & 1 \end{array}\right) \xrightarrow[L_3 \leftarrow -L_3]{L_1 \leftarrow L_1-L_3 \\ L_2 \leftarrow L_2-2L_3} \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & 0 & -1 & -2 \\ 0 & 0 & 1 & -2 & 0 & -1 \end{array}\right) = [\mathbb{I}_3|\mathbb{A}^{-1}].$$

```
A=[1 0 -1; 4 -1 -2; -2 0 1]
inv(A)
```

Exercice 3.58
Calculer $\mathbb{A}^{-1}$ où $\mathbb{A}$ est la matrice $\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 2 \\ 2 & 0 & 1 \end{pmatrix}$.

Correction

$$[\mathbb{A}|\mathbb{I}_3] = \left(\begin{array}{ccc|ccc} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0 & 0 & 1 \end{array}\right) \xrightarrow[L_3 \leftarrow L_3-2L_1]{L_1 \leftarrow L_1 \\ L_2 \leftarrow L_2} \left(\begin{array}{ccc|ccc} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 0 \\ 0 & 0 & -1 & -2 & 0 & 1 \end{array}\right)$$

$$\xrightarrow[L_3 \leftarrow L_3]{L_1 \leftarrow L_1 \\ L_2 \leftarrow L_2} \left(\begin{array}{ccc|ccc} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 0 \\ 0 & 0 & -1 & -2 & 0 & 1 \end{array}\right) \xrightarrow[L_3 \leftarrow -L_3]{L_1 \leftarrow L_1+L_3 \\ L_2 \leftarrow L_2+2L_3} \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -4 & 1 & 2 \\ 0 & 0 & 1 & 2 & 0 & -1 \end{array}\right) = [\mathbb{I}_3|\mathbb{A}^{-1}].$$

```
A=[1 0 1; 0 1 2; 2 0 1]
inv(A)
```

Exercice 3.59
Calculer les inverses des matrices suivantes (si elles existent) :

$$\mathbb{A}=\begin{pmatrix}2 & -3\\ 4 & 5\end{pmatrix},\qquad \mathbb{B}=\begin{pmatrix}1 & 5 & -3\\ 2 & 11 & 1\\ 2 & 9 & -11\end{pmatrix},\qquad \mathbb{C}=\begin{pmatrix}1 & 5 & -3\\ 2 & 11 & 1\\ 1 & 4 & -10\end{pmatrix}.$$

Correction
$\mathbb{A}$ est inversible et on trouve

$$\mathbb{A}^{-1}=\frac{1}{22}\begin{pmatrix}5 & 3\\ -4 & 2\end{pmatrix}.$$

$\mathbb{B}$ est inversible et on trouve

$$\mathbb{B}^{-1}=\frac{1}{2}\begin{pmatrix}-130 & 28 & 38\\ 24 & -5 & -7\\ -4 & 1 & 1\end{pmatrix}.$$

$\mathbb{C}$ n'est pas inversible.

```
A=[2 -3; 4 5]
inv(A)
B=[1 5 -3; 2 11 1; 2 9 -11]
inv(B)
C=[1 5 -3; 2 11 1; 1 4 -10]
inv(C)
```

Exercice 3.60
Soit $\mathbb{A}$ la matrice

$$\mathbb{A}=\begin{pmatrix}1 & 0 & 0 & -1\\ 1 & 1 & -1 & -1\\ 1 & 2 & -1 & -2\\ 1 & 2 & 0 & -2\end{pmatrix}.$$

1. Calculer $\det(\mathbb{A})$.
2. Si $\det(\mathbb{A})\neq 0$, calculer $\mathbb{A}^{-1}$.

Correction

1. Pour calculer le déterminant de la matrice $\mathbb{A}$ on développe par rapport à la première ligne

$$\det(\mathbb{A})=1\cdot\det(\mathbb{A}_{11})-0\cdot\det(\mathbb{A}_{12})+0\cdot\det(\mathbb{A}_{13})-(-1)\cdot\det(\mathbb{A}_{14})=\det\begin{pmatrix}1 & -1 & -1\\ 2 & -1 & -2\\ 2 & 0 & -2\end{pmatrix}+\det\begin{pmatrix}1 & 1 & -1\\ 1 & 2 & -1\\ 1 & 2 & 0\end{pmatrix}.$$

On note que la première colonne de la sous-matrice $\mathbb{A}_{11}$ est l'opposée de la deuxième colonne, ainsi le déterminant de $\mathbb{A}_{11}$ est nul et il ne reste plus qu'à calculer le déterminant de $\mathbb{A}_{14}$ (par exemple en utilisant la règle de Sarrus).

$$\det(\mathbb{A})=0+\det\begin{pmatrix}1 & 1 & -1\\ 1 & 2 & -1\\ 1 & 2 & 0\end{pmatrix}=1.$$

2. Calculons $\mathbb{A}^{-1}$

$$[\mathbb{A}|\mathbb{I}_4]=\left(\begin{array}{cccc|cccc}1 & 0 & 0 & -1 & 1 & 0 & 0 & 0\\ 1 & 1 & -1 & -1 & 0 & 1 & 0 & 0\\ 1 & 2 & -1 & -2 & 0 & 0 & 1 & 0\\ 1 & 2 & 0 & -2 & 0 & 0 & 0 & 1\end{array}\right)\xrightarrow[L_4\leftarrow L_4-L_1]{\substack{L_2\leftarrow L_2-L_1\\ L_3\leftarrow L_3-L_1}}\left(\begin{array}{cccc|cccc}1 & 0 & 0 & -1 & 1 & 0 & 0 & 0\\ 0 & 1 & -1 & 0 & -1 & 1 & 0 & 0\\ 0 & 2 & -1 & -1 & -1 & 0 & 1 & 0\\ 0 & 2 & 0 & -1 & -1 & 0 & 0 & 1\end{array}\right)$$

$$\xrightarrow[L_4\leftarrow L_4-2L_2]{\substack{L_1\leftarrow L_1\\ L_3\leftarrow L_3-2L_2}}\left(\begin{array}{cccc|cccc}1 & 0 & 0 & -1 & 1 & 0 & 0 & 0\\ 0 & 1 & -1 & 0 & -1 & 1 & 0 & 0\\ 0 & 0 & 1 & -1 & 1 & -2 & 1 & 0\\ 0 & 0 & 2 & -1 & 1 & -2 & 0 & 1\end{array}\right)\xrightarrow[L_4\leftarrow L_4-2L_3]{\substack{L_1\leftarrow L_1\\ L_2\leftarrow L_2+L_3}}\left(\begin{array}{cccc|cccc}1 & 0 & 0 & -1 & 1 & 0 & 0 & 0\\ 0 & 1 & 0 & -1 & 0 & -1 & 1 & 0\\ 0 & 0 & 1 & -1 & 1 & -2 & 1 & 0\\ 0 & 0 & 0 & 1 & -1 & 2 & -2 & 1\end{array}\right)$$

$$\xrightarrow[\substack{L_2\leftarrow L_2+L_4\\ L_3\leftarrow L_3+L_4}]{L_1\leftarrow L_1+L_4}\left(\begin{array}{cccc|cccc}1 & 0 & 0 & 0 & 0 & 2 & -2 & 1\\ 0 & 1 & 0 & 0 & -1 & 1 & -1 & 1\\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 1\\ 0 & 0 & 0 & 1 & -1 & 2 & -2 & 1\end{array}\right)=[\mathbb{I}_4|\mathbb{A}^{-1}].$$

```
A=[1 0 0 -1; 1 1 -1 -1; 1 2 -1 -2; 1 2 0 -2]
det(A)
inv(A)
```

3.3.4 Méthode de Gauss pour des systèmes rectangulaires (sur ou sous déterminés)

Exercice 3.61
Résoudre le système

$$(S) \quad \begin{cases} -2x+y+z=0, \\ x-2y+z=0, \end{cases}$$

d'inconnue $(x,y,z)\in\mathbb{R}^3$.

Correction
(S) est équivalent au système

$$\begin{cases} -2x+y+z=0, \\ -3y+3z=0, \end{cases}$$

qui admet une infinité de solutions de la forme (κ,κ,κ) pour $\kappa\in\mathbb{R}$.

Exercice 3.62
Soit le système linéaire

$$(S) \quad \begin{cases} x_1+x_2-2x_3+4x_4=6, \\ -3x_1-3x_2+6x_3-12x_4=b. \end{cases}$$

1. Pour quelle valeur de b le système est-il possible ?
2. Donner à b la valeur trouvée au point précédent et calculer la solution complète du système.

Correction
(S) est équivalent au système

$$\begin{cases} x_1+x_2-2x_3+4x_4=6, \\ 0=b+18. \end{cases}$$

1. (S) est possible si et seulement si $b=-18$.
2. Si $b=-18$, (S) admet ∞^3 solutions de la forme $(x_1,x_2,x_3,x_4)=(6-a+2b-4c,a,b,c)$ avec $a,b,c\in\mathbb{R}$.

Exercice 3.63
Résoudre le système

$$(S) \quad \begin{cases} x+2y+z=-1, \\ 2x+y-z=1, \\ -x+y+2z=-2, \\ x+y+z=4. \end{cases}$$

Correction
(S) étant un système de 4 équations à 3 inconnues, on considère le sous-système carré d'ordre 3

$$(S') \quad \begin{cases} x+2y+z=-1, \\ 2x+y-z=1, \\ -x+y+2z=-2, \end{cases}$$

qu'on peut résoudre par la méthode du pivot de GAUSS

$$\begin{cases} x+2y+z=-1, \\ 2x+y-z=1, \\ -x+y+2z=-2, \end{cases} \xrightarrow[L_3\leftarrow L_3+L_1]{L_2\leftarrow L_2-2L_1} \begin{cases} x+2y+z=-1, \\ -3y-3z=3, \\ 3y+3z=-3, \end{cases} \xrightarrow{L_3\leftarrow L_3+L_2} \begin{cases} x+2y+z=-1, \\ -3y-3z=3, \\ 0=0, \end{cases}$$

qui admet une infinité de solutions de la forme $(1+\kappa,-1-\kappa,\kappa)$ pour $\kappa\in\mathbb{R}$. Cherchons parmi ces solutions celles qui vérifient l'équation de (S) qui n'apparaît pas dans (S') : pour $(x,y,z)=(1+\kappa,-1-\kappa,\kappa)$ on a $x+y+z=1+\kappa-1-\kappa+\kappa=\kappa$ donc $x+y+z=4$ si et seulement si $\kappa=4$ ainsi (S) admet l'unique solution $(5,-5,4)$.

Exercice 3.64
Déterminer si le système suivant a une solution non nulle. Dans le cas affirmatif trouver la(les) solution(s) et expliquer pourquoi :

$$(S)\quad\begin{cases}x-2y+2z=0,\\2x+y-2z=0,\\3x+4y-6z=0,\\3x-11y+12z=0.\end{cases}$$

Correction
(S) étant un système de 4 équations à 3 inconnues, on considère le sous-système carré d'ordre 3

$$(S')\quad\begin{cases}x-2y+2z=0,\\2x+y-2z=0,\\3x+4y-6z=0,\end{cases}$$

qu'on peut résoudre par la méthode du pivot de GAUSS

$$\begin{cases}x-2y+2z=0,\\2x+y-2z=0,\\3x+4y-6z=0,\end{cases}\xrightarrow[L_3\leftarrow L_3-3L_1]{L_2\leftarrow L_2-2L_1}\begin{cases}x-2y+2z=0,\\5y-6z=0,\\10y-12z=0,\end{cases}\xrightarrow{L_3\leftarrow L_3-2L_2}\begin{cases}x-2y+2z=0,\\5y-6z=0,\\0=0,\end{cases}$$

qui admet une infinité de solutions de la forme $(2\kappa,6\kappa,5\kappa)$ pour $\kappa\in\mathbb{R}$. Cherchons parmi ces solutions celles qui vérifient l'équation de (S) qui n'apparaît pas dans (S') : pour $(x,y,z)=(2\kappa,6\kappa,5\kappa)$ on a $3x-11y+12z=6\kappa-66\kappa+60\kappa=0$ donc $3x-11y+12z=0$ pour tout $\kappa\in\mathbb{R}$ ainsi (S) admet une infinité de solutions de la forme $(2\kappa,6\kappa,5\kappa)$ pour $\kappa\in\mathbb{R}$.

Exercice 3.65
Équilibrer l'équation

$$x_1\,\mathrm{CuSCN}+x_2\,\mathrm{KIO_3}+x_3\,\mathrm{HCl}\to x_4\,\mathrm{CuSO_4}+x_5\,\mathrm{KCl}+x_6\,\mathrm{HCN}+x_7\,\mathrm{ICl}+x_8\,\mathrm{H_2O}$$

Correction
Méthode "par tentatives". Chaque élément chimique en jeu fournit une équation :

$$\text{Le sulfure de cuire CuS donne } x_1=x_4 \tag{3.1}$$
$$\text{Le cyanure CN donne } x_1=x_6 \tag{3.2}$$
$$\text{Le potassium K donne } x_2=x_5 \tag{3.3}$$
$$\text{Le iode I donne } x_2=x_7 \tag{3.4}$$
$$\text{Le chlore Cl donne } x_3=x_5+x_7 \tag{3.5}$$
$$\text{L'oxygène O donne } 3x_2=4x_4+x_8 \tag{3.6}$$
$$\text{L'hydrogène H donne } x_3=x_6+2x_8 \tag{3.7}$$

On a alors

$$x_1\overset{(3.1)}{=}x_4\overset{(3.2)}{=}x_6, \tag{3.8}$$
$$x_2\overset{(3.3)}{=}x_5\overset{(3.4)}{=}x_7, \tag{3.9}$$
$$x_3\overset{(3.5)}{=}x_5+x_7\overset{(3.8)}{=}2x_2 \tag{3.10}$$

De plus

$$x_8\overset{(3.6)}{=}3x_2-4x_4\overset{(3.8)}{=}3x_2-4x_1 \tag{3.11}$$

$$2x_8 \overset{(3.7)}{=} x_3 - x_1 \overset{(3.8)}{=} 2x_2 - x_1 \tag{3.12}$$

$$2x_2 - x_1 \overset{(3.12)}{=} 2x_8 \overset{(3.11)}{=} 2(3x_2 - 4x_1)$$

Cela donne

$$4x_2 = 7x_1 \tag{3.13}$$

d'où

$$4x_8 \overset{(3.12)}{=} 2(2x_2 - x_1) = 4x_2 - 2x_1 \overset{(3.13)}{=} 7x_1 - 2x_1 = 5x_1 \tag{3.14}$$

Enfin on a

$$2x_3 \overset{(3.7)}{=} 2(x_6 + 2x_8) \overset{(3.8)-(3.14)}{=} 2x_1 + 5x_1 = 7x_1$$

Conclusion :

$$\begin{aligned}
x_2 &= \frac{7}{4}x_1 \\
x_3 &= \frac{7}{2}x_1 \\
x_4 &= x_1 \\
x_5 &= x_2 = \frac{7}{4}x_1 \\
x_6 &= x_1 \\
x_7 &= x_2 = \frac{7}{4}x_1 \\
x_8 &= \frac{5}{4}x_1
\end{aligned}$$

On choisit $x_1 = 4$ (pour ne pas avoir de fractions) ainsi

$$4\mathrm{CuSCN} + 7\mathrm{KIO_3} + 14\mathrm{HCl} \rightarrow 4\mathrm{CuSO_4} + 7\mathrm{KCl} + 4\mathrm{HCN} + 7\mathrm{ICl} + 5\mathrm{H_2O}$$

Méthode de Gauss : on réécrit les équations dans l'ordre suivant

$$\left\{\begin{array}{rrrrrrrrr}
x_1 & & & -x_4 & & & & & = 0 \\
 & x_2 & & & & -x_7 & & & = 0 \\
 & & x_3 & & -x_5 & & -x_7 & & = 0 \\
x_1 & & & & & & -x_6 & & = 0 \\
 & x_2 & & & -x_5 & & & & = 0 \\
 & 3x_2 & & -4x_4 & & & & -x_8 & = 0 \\
 & & x_3 & & & -x_6 & & -2x_8 & = 0
\end{array}\right. \xrightarrow{L_4 \leftarrow L_4 - L_1} \left\{\begin{array}{rrrrrrrrr}
x_1 & & & -x_4 & & & & & = 0 \\
 & x_2 & & & & -x_7 & & & = 0 \\
 & & x_3 & & -x_5 & & -x_7 & & = 0 \\
 & & & x_4 & & -x_6 & & & = 0 \\
 & x_2 & & & -x_5 & & & & = 0 \\
 & 3x_2 & & -4x_4 & & & & -x_8 & = 0 \\
 & & x_3 & & & -x_6 & & -2x_8 & = 0
\end{array}\right. \xrightarrow[L_6 \leftarrow L_6 - 3L_2]{L_5 \leftarrow L_5 - L_2}$$

$$\left\{\begin{array}{rrrrrrrrr}
x_1 & & & -x_4 & & & & & = 0 \\
 & x_2 & & & & -x_7 & & & = 0 \\
 & & x_3 & & -x_5 & & -x_7 & & = 0 \\
 & & & x_4 & & -x_6 & & & = 0 \\
 & & & & -x_5 & & +x_7 & & = 0 \\
 & & & -4x_4 & & & +3x_7 & -x_8 & = 0 \\
 & & x_3 & & & -x_6 & & -2x_8 & = 0
\end{array}\right. \xrightarrow{L_7 \leftarrow L_7 - L_3} \left\{\begin{array}{rrrrrrrrr}
x_1 & & & -x_4 & & & & & = 0 \\
 & x_2 & & & & -x_7 & & & = 0 \\
 & & x_3 & & -x_5 & & -x_7 & & = 0 \\
 & & & x_4 & & -x_6 & & & = 0 \\
 & & & & -x_5 & & +x_7 & & = 0 \\
 & & & -4x_4 & & & +3x_7 & -x_8 & = 0 \\
 & & & & x_5 & -x_6 & +x_7 & -2x_8 & = 0
\end{array}\right. \xrightarrow{L_6 \leftarrow L_6 + 4L_4}$$

$$\left\{\begin{array}{rrrrrrrrr}
x_1 & & & -x_4 & & & & & = 0 \\
 & x_2 & & & & -x_7 & & & = 0 \\
 & & x_3 & & -x_5 & & -x_7 & & = 0 \\
 & & & x_4 & & -x_6 & & & = 0 \\
 & & & & -x_5 & & +x_7 & & = 0 \\
 & & & & & -4x_6 & +3x_7 & -x_8 & = 0 \\
 & & & & x_5 & -x_6 & +x_7 & -2x_8 & = 0
\end{array}\right. \xrightarrow{L_7 \leftarrow L_7 + L_5} \left\{\begin{array}{rrrrrrrrr}
x_1 & & & -x_4 & & & & & = 0 \\
 & x_2 & & & & -x_7 & & & = 0 \\
 & & x_3 & & -x_5 & & -x_7 & & = 0 \\
 & & & x_4 & & -x_6 & & & = 0 \\
 & & & & -x_5 & & +x_7 & & = 0 \\
 & & & & & -4x_6 & +3x_7 & -x_8 & = 0 \\
 & & & & & -x_6 & +2x_7 & -2x_8 & = 0
\end{array}\right. \xrightarrow{L_7 \leftarrow L_7 - \frac{1}{4}L_6}$$

$$\begin{cases} x_1 - x_4 = 0 \\ x_2 - x_7 = 0 \\ x_3 - x_5 - x_7 = 0 \\ x_4 - x_6 = 0 \\ -x_5 + x_7 = 0 \\ -4x_6 + 3x_7 - x_8 = 0 \\ \frac{5}{4}x_7 - \frac{7}{4}x_8 = 0 \end{cases}$$

On pose donc $x_8 = 5\kappa$ ainsi $x_7 = \frac{7}{5}x_8 = 7\kappa$, $x_6 = \frac{x_8 - 3x_7}{-4} = 4\kappa$, $x_5 = x_7 = 7\kappa$, $x_4 = x_6 = 4\kappa$, $x_3 = x_5 + x_7 = 14\kappa$, $x_2 = x_7 = 7\kappa$ et $x_1 = x_4 = 4\kappa$.

On peut choisir par exemple $\kappa = 1$ et on a

$$4\mathrm{CuSCN} + 7\mathrm{KIO_3} + 14\mathrm{HCl} \rightarrow 4\mathrm{CuSO_4} + 7\mathrm{KCl} + 4\mathrm{HCN} + 7\mathrm{ICl} + 5\mathrm{H_2O}$$

3.4 Pour aller plus loin

Exercice 3.66 (Représentation et manipulation de polynômes)

Dans cette exercice nous allons construire des fonctions qui se trouvent déjà dans Octave, on pourra comparer donc le résultat obtenu avec celui d'Octave. Attention, vous devez programmer vous même les fonctions indiquées. Toute utilisation de fonctions toutes prêtes ne sera pas prise en compte.

Soit $\mathbb{R}_n[x]$ l'ensemble des polynômes de degré inférieur ou égale à n, $n \in \mathbb{N}^*$. Tout polynôme de cet espace vectoriel s'écrit de manière unique comme

$$p_n(x) = \sum_{i=0}^{n} a_i x^i = a_0 + a_1 x + \cdots + a_n x^n, \qquad \text{où } a_i \in \mathbb{R} \text{ pour } i = 0, \dots n.$$

Les $n+1$ valeurs réels $a_0, a_1, \dots, a_n$ sont appelés les **coordonnées de p_n dans la base canonique** [a] de $\mathbb{R}_n[x]$ et on peut les stocker dans un vecteur $\mathbf{p}$:

$$\mathbf{p} = \text{coord}(p_n, \mathscr{C}_n) = (a_0, a_1, a_2, \dots, a_n) \in \mathbb{R}^{n+1}$$

Dans Octave nous utiliserons le vecteur $\mathbf{p}$ pour manipuler un polynôme et nous construirons des fonctions pour opérer sur les polynômes à partir de cette représentation. Par exemple, pour construire le polynôme $p_2(x) = 2 - x + x^2$ nous écrirons

```
p=[2 -1 1]
```

Dans le **script** `script_pol.m` on écrira les instructions utilisées pour tester les **`function`** suivantes :

1. Implémenter une fonction appelée **`eval_pol`** permettant d'évaluer le polynôme p (la fonction polynomiale) en des points donnés. La syntaxe doit être `function y=eval_pol(p,x)` où x est une valeur numérique ou un vecteur. Dans le second cas on doit obtenir un vecteur contenant les valeurs de la fonction polynomiale aux différents points spécifiés dans le vecteur $\mathbf{x}$. Par exemple, pour évaluer le polynôme $p(x) = 1 + 2x + 3x^2$ en $\mathbf{x} = (-1, 0, 1, 2)$ nous écrirons

   ```
   p=[1 2 3]
   y=eval_pol(p,[-1,0,1,2])
   ```

   et on veut obtenir le vecteur $\mathbf{y} = p(\mathbf{x}) = (2, 1, 6, 17)$. En effet on a

   $$\begin{aligned} p(x) &= 1 + 2x + 3x^2 \\ \mathbf{p} &= \text{coord}(p, \mathscr{C}_2) = (1, 2, 3) \end{aligned} \qquad \begin{aligned} p(-1) &= 1 + 2 \times (-1) + 3 \times ((-1)^2) = 1 - 2 + 3 = 2 \\ p(0) &= 1 + 2 \times 0 + 3 \times (0^2) = 1 + 0 + 0 = 1 \\ p(1) &= 1 + 2 \times 1 + 3 \times (1^2) = 1 + 2 + 3 = 6 \\ p(2) &= 1 + 2 \times 2 + 3 \times (2^2) = 1 + 4 + 12 = 17 \end{aligned}$$

2. Implémenter une fonction appelée **`plot_pol`** prenant en entrée un polynôme p (*i.e.* le vecteur qui contient ses coordonnées) et deux réels a et $b > a$ et qui trace le graphe de p pour $x \in [a, b]$. La syntaxe de l'instruction doit être `plot_pol(p,a,b)`. Par exemple, pour tracer le graphe du polynôme $p(x) = 1 + 2x + 3x^2$ sur l'intervalle $[-2; 2]$ nous écrirons

   ```
   p=[1 2 3]
   plot_pol(p,-2,2)
   ```

3. Implémenter une fonction appelée **`sum_pol`** renvoyant la somme de deux polynômes (attention, si les deux polynômes n'ont pas même degré, il faudra ajouter des zéros en fin du polynôme de plus petit degré afin de pouvoir calculer l'addition des deux vecteurs représentatifs). Par exemple, pour $\mathbf{p} = (1, 2, 3)$ et $\mathbf{q} = (1, -2)$, on veut obtenir $\mathbf{s} = (2, 0, 3)$:

   $$\begin{aligned} p(x) &= 1 + 2x + 3x^2 \\ q(x) &= 1 - 2x \\ s(x) = p(x) + q(x) &= 2 + 3x^2 \end{aligned} \qquad \begin{aligned} \mathbf{p} &= \text{coord}(p, \mathscr{C}_2) = (1, 2, 3) \\ \mathbf{q} &= \text{coord}(q, \mathscr{C}_1) = (1, -2) \implies \mathbf{q} = \text{coord}(q, \mathscr{C}_2) = (1, -2, 0) \\ \mathbf{s} &= \text{coord}(p + q, \mathscr{C}_2) = (2, 0, 3) \end{aligned}$$

4. Implémenter une fonction appelée **`prod_pol`** renvoyant le produit de deux polynômes.

Exemple, pour $\mathbf{p}=(1,0,3)$ et $\mathbf{q}=(1,-2)$, on veut obtenir $\mathbf{u}=(1,-2,3,-6)$.

$$\begin{aligned} p(x) &= 1+3x^2 & \mathbf{p} &= \text{coord}(p,\mathscr{C}_2)=(1,0,3)\\ q(x) &= 1-2x & \mathbf{q} &= \text{coord}(q,\mathscr{C}_2)=(1,-2,0)\\ u(x)=p(x)\times q(x) &= 1\times p(x)-2x\times p(x)=1-2x+3x^2-6x^3 & \mathbf{u} &= \text{coord}(p\times q,\mathscr{C}_3)=(1,-2,3,-6)\end{aligned}$$

5. Implémenter une fonction appelée **`derivee_pol`** renvoyant la dérivée d du polynôme p donné en entrée (attention, si $p\in\mathbb{R}_n[x]$, alors $d\in\mathbb{R}_{n-1}[x]$).
   Exemple, pour $\mathbf{p}=(1,2,6)$, on veut obtenir $\mathbf{d}=(2,12)$.

$$\begin{aligned} p(x) &= 1+2x+6x^2 & \mathbf{p} &= \text{coord}(p,\mathscr{C}_2)=(1,2,6)\\ d(x)=p'(x) &= 2+12x & \mathbf{d} &= \text{coord}(d,\mathscr{C}_1)=(2,12)\end{aligned}$$

6. Implémenter une fonction appelée **`primitive_pol`** renvoyant la primitive v du polynôme p donné en entrée ayant 0 pour racine (attention, si $p\in\mathbb{R}_n[x]$, alors $v\in\mathbb{R}_{n+1}[x]$).
   Exemple, pour $\mathbf{p}=(1,2,6)$, on veut obtenir $\mathbf{v}=(0,1,1,2)$.

$$\begin{aligned} p(x) &= 1+2x+6x^2 & \mathbf{p} &= \text{coord}(p,\mathscr{C}_2)=(1,2,6)\\ v(x)=\int_0^x p(t)\,\mathrm{d}t=\int_0^x 1+2t+6t^2\,\mathrm{d}t &= x+x^2+2x^3 & \mathbf{v} &= \text{coord}(v,\mathscr{C}_3)=(0,1,1,2)\end{aligned}$$

7. Implémenter une fonction appelée **`integrale_pol`** renvoyant l'intégrale d'un polynôme entre deux valeurs a et b.
   Exemple, pour $\mathbf{p}=(1,2,6)$, $a=1$ et $b=2$, on veut obtenir $c=18$:

$$\begin{aligned} p(x) &= 1+2x+6x^2\\ c=\int_a^b p(t)\,\mathrm{d}t=\int_0^b p(t)\,\mathrm{d}t-\int_0^a p(t)\,\mathrm{d}t &= v(b)-v(a)=b+b^2+2b^3-a-a^2-2a^3=18.\end{aligned}$$

8. Implémenter une fonction appelée **`print_pol`** prenant en entrée un polynôme p (*i.e.* le vecteur qui contient ses coordonnées) et qui écrit dans la fenêtre de commande le polynôme dans la base canonique.
   Exemple, pour $\mathbf{p}=(1,2,-3,0,7)$, on veut afficher le message `1+2x-3x^2+7x^4`.

a. La base canonique de l'espace vectoriel $\mathbb{R}_n[x]$ est l'ensemble $\mathscr{C}_n=\{1,x,x^2,\dots,x^n\}$

Correction
Dans le fichier `script_pol.m` on écrit les instructions qui permettent de tester les différents points de cet exercice.

1. Dans le fichier *`eval_pol`* on écrit

```
function [y]=eval_pol(p,x)
  y=zeros(size(x));
  for k=1:length(p)
     y+=p(k)*x.^(k-1);
  end
end
```

et on teste cette fonction par exemple comme suit

```
y=eval_pol([1 2 3],[-1 0 1 2])
```

2. Dans le fichier *plot_pol.m* on écrit

```
function plot_pol(p,a,b)
  x=linspace(a,b,100);
  y=eval_pol(p,x);
  plot(x,y);
end
```

et on teste cette fonction par exemple comme suit

```
plot_pol([-1 0 1],-2,2)
```

3. Sans perte de généralité, supposons que $n>m$, alors

$$\begin{aligned} p(x)&=\sum_{i=0}^{n}a_ix^i=\sum_{i=0}^{m}a_ix^i+\sum_{i=m+1}^{n}a_ix^i & &\text{coord}(p,\mathscr{C})=(a_0,a_1,a_2,\dots,a_m,a_{m+1},\dots,a_n)\\ q(x)&=\sum_{i=0}^{m}b_ix^i=\sum_{i=0}^{m}b_ix^i+\sum_{i=m+1}^{n}0\times x^i & &\text{coord}(q,\mathscr{C})=(b_0,b_1,b_2,\dots,b_m)\end{aligned}$$

$$(p+q)(x)=\sum_{i=0}^{m}(a_i+b_i)x^i+\sum_{i=m+1}^{n}a_ix^i \qquad \text{coord}(p+q,\mathscr{C})=(a_0+b_0,a_1+b_1,a_2+b_2,\dots,a_m+b_m,a_{m+1},\dots,a_n)$$

Dans le fichier `sum_pol.m` on écrit

```
function s=sum_pol(p,q)
  n=length(p);
  m=length(q);
  A=zeros(2,max(n,m));
  A(1,1:n)=p;
  A(2,1:m)=q;
  s=sum(A);
end
```

et on teste cette fonction par exemple comme suit

```
s=sum_pol([1 2 3],[4 5 6])
s=sum_pol([1 2 3],[4 5])
s=sum_pol([1 2],[4 5 6])
```

4. Dans le fichier *prod_pol.m* on écrit

```
function s=prod_pol(p,q)
  n=length(p);
  m=length(q);
  A=zeros(m,n+m-1);
  for i=1:m
    A(i,i:n+i-1)=q(i)*p;
  end
  s=sum(A);
end
```

et on teste cette fonction par exemple comme suit

```
u=prod_pol([1],[4 5 6])
u=prod_pol([1 2],[4 5 6])
u=prod_pol([1 2 3],[4 5 6])
u=prod_pol([1 2 3 4],[4 5 6])
```

5. Remarquons que

$$p(x)=\sum_{i=0}^{n}a_ix^i \qquad d(x)=p'(x)=\sum_{i=0}^{n}i\,a_ix^{i-1}$$
$$\text{coord}(p,\mathscr{C}_n)=(a_0,a_1,a_2,\dots,a_n) \qquad \text{coord}\,(d,\mathscr{C}_{n-1})=(a_1,2a_2,\dots,na_n)$$

Dans le fichier *derivee_pol.m* on écrit

```
function d=derivee_pol(p)
  n=length(p);
  d=p(2:end).*(1:n-1);
end
```

et on teste cette fonction par exemple comme suit

```
d=derivee_pol([1])
d=derivee_pol([1 2])
d=derivee_pol([1 2 3])
d=derivee_pol([1 2 1 1])
```

6. Remarquons que

$$p(x)=\sum_{i=0}^{n}a_ix^i \qquad v(x)=\int_0^x p(t)\,\mathrm{d}t=\sum_{i=0}^{n}a_i\int_0^x t^i\,\mathrm{d}t=\sum_{i=0}^{n}a_i\frac{x^{i+1}}{i+1}$$
$$\text{coord}(p,\mathscr{C}_n)=(a_0,a_1,a_2,\dots,a_n) \qquad \text{coord}\,(v,\mathscr{C}_{n+1})=\left(0,\frac{a_0}{0+1},\frac{a_1}{1+1},\frac{a_2}{2+1},\dots,\frac{a_n}{n+1}\right)$$

Dans le fichier *primitive_pol.m* on écrit

```
function prim=primitive_pol(p)
  n=length(p);
  prim(1)=0;
  prim([2:n+1])=p([1:n])./[1:n];
end
```

et on teste cette fonction par exemple comme suit

```
v=primitive_pol([1])
v=primitive_pol([1 2])
v=primitive_pol([1 2 3])
v=primitive_pol([1 2 1 1])
```

7. Dans le fichier *integrale_pol.m* on écrit

```
function integr=integrale_pol(p,a,b)
  prim=primitive_pol(p);
  n=length(prim); % = 1+length(p)
  aa([1:n])=a.^([0:n-1]);
  prima=sum(prim.*aa);
  bb([1:n])=b.^([0:n-1]);
  primb=sum(prim.*bb);
  integr=primb-prima;
end
```

et on teste cette fonction par exemple comme suit

```
w=integrale_pol([1 1], 1, 2)
```

8. Dans le fichier *print_pol.m* on écrit

```
function str=print_pol(p)
  n=length(p);
  str='';
  if n==1;
      str=strcat(num2str(p(1)));
  else
      strsign=char((p>0)*'+' + (p<0)*' ' + (p
          ==0)*'0');
      if p(1)~=0
          str=num2str(p(1));
      end
      if p(2)~=0
          str=strcat(str,strsign(2),num2str(p(2)
              ),'x');
      end
      for i=3:n
        if p(i)~=0
          str=strcat(str,strsign(i),num2str(p(i)
              ),'x^',num2str(i-1));
        end
      end
   end
end
```

et on teste cette fonction par exemple comme suit

```
print_pol([1 2 -3 -7 5])
print_pol([1 0 -3  0 5])
print_pol([1 2 -3 7])
print_pol([1 2 -3])
print_pol([1 2])
print_pol([1])
```

Exercice 3.67

Considérons un système linéaire sous la forme matricielle $\mathbb{A}\mathbf{x} = \mathbf{b}$ où $\mathbb{A}$ est une matrice de $\mathbb{R}^{n\times n}$ non singulière et $\mathbf{b}$ est un vecteur colonne de $\mathbb{R}^n$.
Implémenter une fonction appelée **`mygauss`** qui transforme la matrice augmentée $[\mathbb{A}|\mathbf{b}]$ en une matrice triangulaire supérieure par la méthode de GAUSS et, à chaque étape, affiche les opérations sur les lignes ainsi que la matrice modifiée. Enfin, elle résout le système linéaire triangulaire par remontée.
La syntaxe doit être `function [x]=mygauss(A,b)`
Écrire un script appelé **`TESTmygauss.m`** pour tester cette fonction sur l'exemple suivant : pour

$$\mathbb{A} = \begin{pmatrix} 1 & 0 & 3 \\ 2 & 2 & 2 \\ 3 & 6 & 4 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} 4 \\ 6 \\ 13 \end{pmatrix}$$

on doit obtenir

$$\mathbf{x} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Correction
Dans le fichier `mygauss.m` on écrit

```
function [x]=mygauss(A,b)
   printf("Matrice augmentee : [A|b]\n")
   Ab = [A,b]
   [n,m]=size(A);
   tol=1.0e-9;
   for k=1:n-1
      printf(strcat("\nEtape ",num2str(k),"\n"))
      for i=k+1:n
            L(i,k)=Ab(i,k)/Ab(k,k);
            printf(strcat("\tL_",num2str(i)," <- L_",num2str(i)," - (",num2str(L(i,k)),") L_",num2str(k
                ),"\n"))
            Ab(i,k:n+1)=Ab(i,k:n+1)-L(i,k)*Ab(k,k:n+1);
      end
      Ab
   end
```

```
   printf("\nResolution du systeme triangulaire ainsi obtenu\n")
   U=triu(Ab(:,1:n));
   y=Ab(:,n+1);
   x(n)=y(n)/U(n,n);
   for i=n-1:-1:1
     x(i)=(y(i)-dot(U(i,i+1:n),x(i+1:n)))/U(i,i);
   end
end
```

et on teste cette fonction par exemple comme suit

```
clear all
A=[1 0 3; 2 2 2; 3 6 4];
b=[4; 6; 13];
x=mygauss(A,b)'
% Pour verifier notre resultat on peut
% comparer au resultat d'Octave
xOctave=A\b
% ou verifier qua Ax=b
printf(strcat("||Ax-b||=",num2str(norm(A*x-b)),"\n"))
```