

IMT Atlantique

Dépt. Signal & Communications
Technopôle de Brest-Iroise - CS 83818
29238 Brest Cedex 3
Téléphone : +33 (0)2 29 00 13 04
Télécopie : +33 (0)2 29 00 10 12
URL : www.imt-atlantique.fr



Document support de TP DS-TP1

Karine AMIS, François-Xavier Socheleau

UE ATSA - Guide d'introduction à Matlab

Date d'édition : 16 avril 2025
Version : 1.1



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Sommaire

1. Introduction	3
2. Démarrage.....	3
3. Commandes générales	4
3.1. Gestion de fichiers.....	4
3.2. Constantes prédéfinies	4
3.3. Historique	4
3.4. Variables d'environnement.....	4
4. Déclaration d'un vecteur	4
4.1. Vecteur de taille connue n	4
4.2. Vecteur de taille inconnue.....	4
5. Opérations sur les matrices	5
5.1. Déclaration, éléments.....	5
5.2. Opérations sur les matrices (et les vecteurs).....	5
6. Quelques fonctions élémentaires sur les vecteurs et les matrices	5
7. Boucles	6
7.1. Boucle "for"	6
7.2. Boucle "while".....	6
8. Figures.....	6
9. Fonctions.....	6
10. Pour en savoir davantage.....	7
11. Exercices	8
11.1. Exercice 1	8
11.2. Exercice 2	8
11.3. Exercice 3	8
11.4. Exercice 4	8

Liste des figures

1. Illustration - Navigateur Matlab 3

Liste des tableaux



~ \$ matlab

3. Commandes générales

3.1. Gestion de fichiers

```
pwd      % affiche le nom du répertoire courant pour Matlab
cd rep   % change le répertoire courant pour Matlab qui devient rep
dir       % fournit le catalogue d'un répertoire
delete   % efface des fichiers ou des objets graphiques
```

3.2. Constantes prédéfinies

```
pi       % 3,1415 ...
eps      % 2.2204e-016
Inf      % nombre infini
NaN      % n'est pas un nombre ; exprime parfois une indétermination
```

3.3. Historique

Matlab conserve l'historique des commandes. Il est donc possible de récupérer des instructions déjà saisies (et ensuite de les modifier dans le but de les réutiliser) : \rightarrow , \leftarrow , \uparrow , \downarrow

3.4. Variables d'environnement

Matlab garde en mémoire les variables qui ont été créées (cf. l'onglet workspace). Pour les afficher et les effacer par la ligne de commande :

```
who       % donne la liste des variables présentes dans l'espace de travail
whos      % donne la liste des variables présentes dans l'espace de travail ainsi que leurs propriétés
what      % donne la liste des fichiers .m et .mat présents dans le répertoire courant
clear all % efface toutes les variables créées dans l'espace de travail
```

4. Déclaration d'un vecteur

4.1. Vecteur de taille connue n

Vecteur ligne :

```
x=[1 2 3 4 5 6];
x=[1:0.25:6];      % Déclaration de x contenant des éléments allant de 1 à 6 par pas 0,25
x=zeros(1,n);      % Déclaration de x et initialisation à  $0_{1 \times n}$ 
x=ones(1,n);       % Déclaration de x et initialisation à  $1_{1 \times n}$ 
```

Vecteur colonne :

```
x=[1;2;3;4;5;6];
x=zeros(n,1);      % Déclaration de x et initialisation à  $0_{n \times 1}$ 
x=ones(n,1);       % Déclaration de x et initialisation à  $1_{n \times 1}$ 
```

4.2. Vecteur de taille inconnue

$x=[]$; % Initialisation de x comme vecteur vide

Puis au cours d'un processus récursif, on vient insérer de nouvelles valeurs : si on souhaite insérer le vecteur a de longueur l dans le vecteur x de longueur alors égale à n (ici a et x sont des vecteurs ligne. Mais ce qui suit se généralise aux vecteurs colonne).

```
x=[x a];           % insertion à la fin
x=[a x];           % insertion au début
x=[x(1,1:k-1) a x(1,k:n)]; % insertion du vecteur a
                        à partir de la  $k$ -ième composante de  $x$ 
```

A l'issue de cette insertion, le vecteur x est de longueur $l + n$

N.B :

- la longueur d'un vecteur x est donnée par : `length(x)`
- $x(1,k)$ désigne la k -ième composante de x

- **Attention!!!!** Sous Matlab, la numérotation des indices commence à 1
- Ne jamais utiliser i et j comme indices car ce sont des lettres réservées : $i = j = e^{i\frac{\pi}{2}}$ ou alors utiliser $1i$ ou $1j$ pour désigner le nombre complexe $e^{i\frac{\pi}{2}}$.

5. Opérations sur les matrices

5.1. Déclaration, éléments...

Si on connaît la taille de la matrice A (par exemple $n \times m$), on peut initialiser la matrice à :

```
A=[1 2 3;4 5 6];    % A =  $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ 
A=zeros(n,m);       % matrice nulle
A=ones(n,m);         % matrice "1"  $a_{ij} = 1$ 
A=eye(n);            % matrice identité si  $n = m$ 
```

La méthode décrite pour un vecteur de taille inconnue reste valable, modulo quelques précautions sur l'insertion des matrices

$A(k, l)$ désigne l'élément situé à l'intersection de la ligne k et de la colonne l .

5.2. Opérations sur les matrices (et les vecteurs)

$C=A*B$	$\% c_{qp} = \sum_k a_{qk} \cdot b_{kp}$
$C=A.*B$	$\% c_{qp} = a_{qp} \cdot b_{qp}$
$C=A./B$	$\% c_{qp} = \frac{a_{qp}}{b_{qp}}$
$C=\text{conj}(A)$	$\% c_{qp} = \bar{a}_{qp}$
$C=A'$	$\% c_{qp} = \bar{a}_{pq}$
$C=A.'$	$\% c_{qp} = a_{pq}$
$C=\text{abs}(A)$	$\% c_{pq} = a_{pq} $

Si C est complexe, $\text{real}(C)$ (resp. $\text{imag}(C)$) fournit la matrice des parties réelles (resp. imaginaires) de ses éléments.

6. Quelques fonctions élémentaires sur les vecteurs et les matrices

```
sqrt    % racine carrée
sin      % sinus
asin     % arc sinus
cos      % cosinus
acos     % arc cosinus
tan      % tangente
atan     % arc tangente
round    % arrondi à l'entier le plus proche
ceil     % arrondi à l'entier supérieur
floor    % arrondi à l'entier inférieur
abs      % valeur absolue
exp      % exponentiel
log      % logarithme népérien
log10    % logarithme base 10
conj     % conjugué
```

<code>max(x)</code>	% valeur maximum contenu dans le vecteur x
<code>min(x)</code>	% valeur minimum contenu dans le vecteur x
<code>sort(x)</code>	% tri par ordre croissant
<code>sum(x)</code>	% somme des éléments de x
<code>prod(x)</code>	% produit des éléments de x
<code>diff(x)</code>	% vecteur des différences entre deux éléments consécutifs de x
<code>mean(x)</code>	% moyenne des éléments de x
<code>median(x)</code>	% médiane
<code>std(x)</code>	% écart type
<code>conv(x,y)</code>	% convolution entre x et y
<code>xcorr(x,y)</code>	% intercorrélacion entre x et y
<code>rank(A)</code>	% rang de la matrice A
<code>det(A)</code>	% déterminant de la matrice A
<code>inv(A)</code>	% inverse de la matrice A
<code>[U,D]=eig(A)</code>	% vecteurs propres et valeurs propres de A

7. Boucles ...

7.1. Boucle “for”

```
for k=1:n:p, %parcours de 1 à p par pas de n
    Corps de la boucle
end,
```

7.2. Boucle “while”

```
while (condition) % Utilisation d'opérateurs binaires “||” ou “&”
    Corps de la boucle
end,
```

8. Figures

Soient x_1 , y_1 , x_2 et y_2 4 vecteurs tels que x_i et y_i soient de **même** longueur. On veut tracer sur la même figure y_1 en fonction de x_1 et y_2 en fonction de x_2 .

```
figure(n);           % n est le numéro de la figure n = 1, 2, ..
clf;                 % efface le contenu de la figure
plot(x1,y1,'bd-.',x2,y2,'ks-'); % trace  $y_1 = f(x_1)$  en pointillés (-.), avec des diamants
                                % (d) et en bleu (b). Ordre couleur-symbole-style de trait.
grid;                % Grille
xlabel('x');          % Nom pour l'axe des abscisses
ylabel('y=f(x)');     % Nom pour l'axe des ordonnées
legend('courbe 1','courbe 2'); % légende
axis([x_min x_max y_min y_max]); % dimension des axes X et Y
```

Si l'on veut rajouter une troisième courbe sur la même figure, on peut utiliser la commande `hold on` et tracer ensuite la courbe à l'aide de la commande `plot`. Il est également possible de zoomer ou de placer des curseurs sur les figures à l'aide de l'interface graphique.

A la place de `plot`, on peut aussi utiliser `stem` ou `bar` (pour les diagrammes par exemple). Pour en savoir plus : “`help plot`”.

9. Fonctions...

Soit *FonctionAlpha* le nom de la fonction que l'on souhaite créer. Dans le répertoire dans lequel le programme principal est situé (le cas échéant il faut préciser le chemin pour atteindre la fonction), écrire

un fichier appelé *FonctionAlpha.m* (les noms du fichier et de la fonction doivent être les mêmes) du type suivant :

```
function [y1 y2 ... yn]=FonctionAlpha(x1,x2,...,xp);
```

Corps de la fonction

Les x_i sont les valeurs d'entrée et y_i celles de sortie.

10. Pour en savoir davantage...

Vous disposez de l'aide en ligne : il suffit de taper dans la fenêtre de commande Matlab "`help nom_de_la_fonction`" dont vous souhaitez des détails. Attention, dans l'aide en ligne, le nom de la fonction est en majuscules, mais son utilisation requiert des minuscules.

Si vous voulez savoir s'il existe une fonction qui réalise ce que vous souhaitez, utilisez la commande "`lookfor mot_cle`". Sinon utilisez l'outil d'aide (cliquez sur le point d'interrogation en haut à droite).

Le site Web de Matlab peut s'avérer utile :

<https://fr.mathworks.com/help/matlab/>

11. Exercices

11.1. Exercice 1

Soit le système d'équations suivant

$$\begin{cases} 3x_1 + 2x_2 + x_3 &= 4 \\ -x_1 + 5x_2 + 2x_3 &= -1 \\ x_1 + x_2 + 5x_3 &= \sqrt{2} \end{cases}$$

- Ecrire le système sous la forme matricielle $Ax = b$ (où vous définissez A et b) et calculer le rang de la matrice A avec Matlab.
- Résoudre le système avec Matlab.

11.2. Exercice 2

Soit un signal $s(t) = \sin^2(2\pi f_0 t)$, avec $f_0 = 10$ Hz. Tracer une figure représentant une version échantillonnée de ce signal à une fréquence $\nu_e = 500$ Hz sur une fenêtre temporelle de 1 s. Ne pas oublier la légende et le nom des axes.

11.3. Exercice 3

Soit $s(k) = \text{rect}_{10}(k)$. A l'aide de la fonction `xcorr`, tracer la fonction d'autocorrélation de $s(k)$ (en rouge). Ne pas oublier la légende et le nom des axes.

11.4. Exercice 4

- Créer une fonction intitulée *AnalyseSigComplexe* qui prend en entrée un signal échantillonné à valeur complexe et qui retourne son module, sa phase, sa partie réelle, sa partie imaginaire, son énergie et sa puissance moyenne.
- Appliquer cette fonction sur le signal

$$s(k) = \ln(k+1)e^{i\pi \frac{k}{10}} \text{rect}_{100}(k)$$

et afficher le module, la partie réelle et la partie imaginaire sur une même figure.

OUR WORLDWIDE PARTNERS UNIVERSITIES - DOUBLE DEGREE AGREEMENTS

3 CAMPUS, 1 SITE



IMT Atlantique Bretagne–Pays de la Loire – <http://www.imt-atlantique.fr/>

Campus de Brest

Technopôle Brest-Iroise
CS 83818
29238 Brest Cedex 3
France
T +33 (0)2 29 00 11 11
F +33 (0)2 29 00 10 00

Campus de Nantes

4, rue Alfred Kastler
CS 20722
44307 Nantes Cedex 3
France
T +33 (0)2 51 85 81 00
F +33 (0)2 99 12 70 08

Campus de Rennes

2, rue de la Châtaigneraie
CS 17607
35576 Cesson Sévigné Cedex
France
T +33 (0)2 99 12 70 00
F +33 (0)2 51 85 81 99

Site de Toulouse

10, avenue Édouard Belin
BP 44004
31028 Toulouse Cedex 04
France
T +33 (0)5 61 33 83 65



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

© IMT Atlantique, 2021
Imprimé à IMT Atlantique
Dépôt légal : Novembre 2021
ISSN :