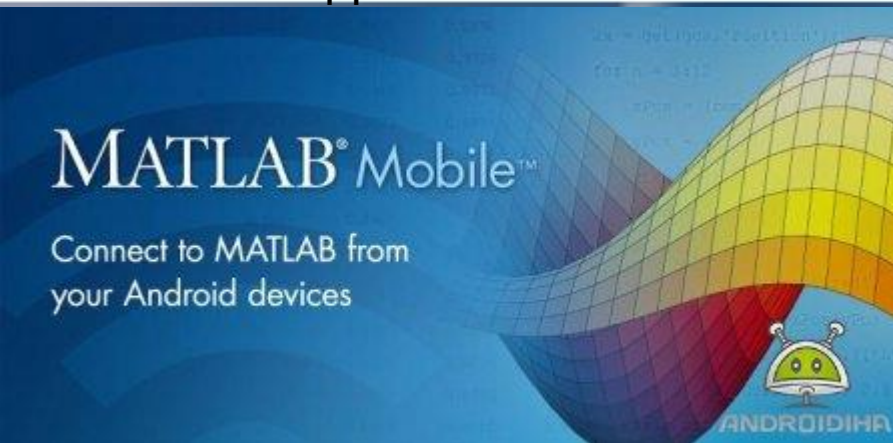


# Programmation Scientifique et Technique :

## Débuter avec Matlab

Jean-Philippe LAUFFENBURGER



# Sommaire

- **Préambule**
- **Bases**
- **Matrices**

## Remarques importantes

- Cette présentation introduit le « **Must Know** » (**choses à connaître impérativement**) des bases de Matlab et du calcul matriciel avec Matlab
- D'autres notions (affichage graphique, programmation structurelle, vérification fonctionnelle, etc. ) seront abordées au travers d'exercices de mise en application
- Cette présentation s'appuie sur des exemples et exercices qu'il FAUT FAIRE sous Matlab
- Cette présentation **ne remplace en rien l'aide de Matlab** accessible via les commandes **help, doc, demos**, etc.

## Prérequis

- Savoir naviguer sur Internet
- Savoir naviguer dans les dossiers gérés par le système d'exploitation Windows
- Savoir utiliser Moodle ([www.e-formation.uha.fr/moodle](http://www.e-formation.uha.fr/moodle)) et la gestion des cours en ligne

## Objectifs

- Acquérir le **vocabulaire du calcul** scientifique (déclaration, affectation, etc.)
- Acquérir une **"méthodologie" dans la conception/structuration** d'algorithmes
- Acquérir **les éléments de base** de Matlab pour en **faciliter la maîtrise progressive**

# Notations

- **>>** : invité de commande de Matlab (« Command Window Prompt ») pour la saisie de vos instructions dans la « **Command Window** »
- **ans** : variable de stockage par défaut
- **A** : matrice
- **a** : scalaire
- ***nom\_fonction*** : nom d'une fonction prédéfinie de Matlab à substituer
- **% chaîne de caractère** : décrit une zone de commentaires
- **%%** délimite une cellule (« cell ») dans un script
- Eléments d'une instruction **en vert** : données optionnelles

## Astuces et raccourci clavier

- Flèches ↑ ou ↓ : rappel des dernières instructions saisies dans la « **Command Window** »
- 1ères lettres d'une instruction + ↑ : rappel des instructions saisies et débutant par ces lettres
- *fx* : accès direct au menu contextuel d'aide
- Affichage automatique des syntaxes d'une fonction

```
>> plot(
plot(Y)
plot(X1,Y1,...,Xn,Yn)
plot(X1,Y1,LineSpec,...,Xn,Yn,LineSpec)
plot(X1,Y1,LineSpec,'PropertyName',PropertyValue)
plot(axes_handle,X1,Y1,LineSpec,'PropertyName',PropertyValue)
More Help...
```

- Fenêtre « **Command History** » : mémorise toutes les saisies au clavier
- *help nom\_fonction* : aide succincte
- *doc nom\_fonction* : aide complète + cas d'utilisation **via l'aide en ligne de Matlab** (« **helpdesk** »)

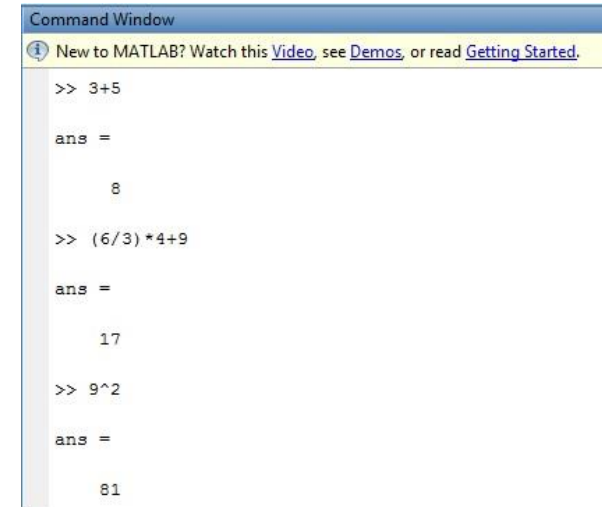
## Astuces et raccourci clavier de l'éditeur (« edit »)

- Ouverture de l'éditeur : `>> edit nom_fonction`
- CTL+R : mise en commentaire des lignes sélectionnées
- CTL+T : suppression des commentaires des lignes sélectionnées
- F5 : exécution du script/fonction en cours d'édition
- CTL+D : ouverture du fichier .m correspondant à la commande sélectionnée
- Chaîne de caractère + TAB : menu contextuel proposant les variables débutant par la chaîne de caractère saisie (« **auto completion** »)
- CTL+Entrée : exécution de la cellule active

# Utilisation simple : calcul numérique (1)

- Matlab réalise les calculs classiques  
⇒ Résultat dans la **Command Window**

```
>> 3+5           % Somme de 2 réels
>> (6/3)*4+9     % Associativité, distributivité, etc.
>> 9^2           % Elévation à la puissance
```



```
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.

>> 3+5

ans =

     8

>> (6/3)*4+9

ans =

    17

>> 9^2

ans =

    81
```

- En respectant les règles d'**associativité**, **distributivité**, etc.

$$\frac{e^{0.7\pi}}{3(\ln(5)+9)^2} \Rightarrow \exp(0.7\pi)/(3*(\log(5)+9)^2)$$

Variable particulière « **ans** »

Stockage du résultat de l'opération exécutée **sans affectation explicite**

```
>> 3+5
```

```
ans =
```

8

**Sans affectation explicite**

```
>> a=3+5
```

```
a =
```

8

**Avec affectation explicite**



## Utilisation simple : calcul numérique (2)

- Utilisation de la variable « **ans** »

```
>> 3+5  
ans =  
    8
```

% Somme de 2 réels

```
>> ans*ans  
ans =  
   64
```

% Multiplication du résultat de l'opération

```
>> 'toto'  
ans =  
    'toto'
```

% Saisie d'une chaîne de caractère

```
>> ans*2
```

% ?? (A compléter)

## Utilisation simple : calcul numérique (3)

- De nombreuses fonctions mathématiques sont prédéfinies (help **elfun**...)

- $\exp(x) \Leftrightarrow e^x$
- $\log(x) \Leftrightarrow \ln(x)$
- $\text{sqrt}(x) \Leftrightarrow \sqrt{x}$
- $\text{abs}(x) \Leftrightarrow |x|$  (le type de x définit la nature de l'opération !!)
- $\text{acos}(x) \Leftrightarrow \cos^{-1}(x)$
- $a^b \Leftrightarrow a^b$
- ...

```
>> cos(0.5)           % Angle en radians
>> cosd(30)           % Angle en degrés
>> cos(pi/3)
>> sqrt(sinh(0.2))    % Racine carrée du sinh
```

**A connaitre !!**

# Affectation et définition de variables (1)

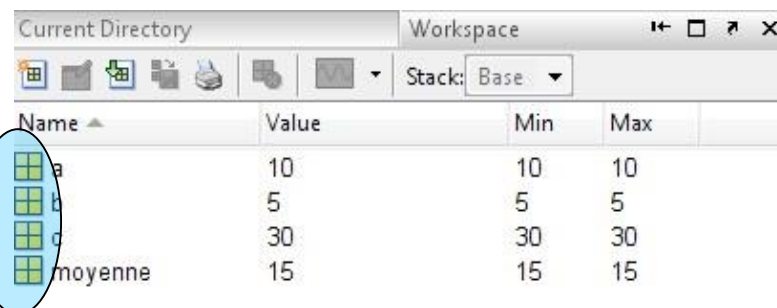
## (A connaître impérativement !!)

- Règle 1 : par défaut, toute **variable est une matrice**
- Règle 2 : **Pas de déclaration de types** (entier, réel, etc.)
- Règle 3 : type par défaut : **double**
- Opérateur **d'affectation** : **=**
- Exemples
  - `>>a = 10` % Variable scalaire entière
  - `>>b = 5;` % Intérêt du ; à la fin d'une instruction??
  - `>>c = 30`
  - `>>moyenne = (a+b+c)/3` % Affectation du résultat d'une opération

- Variables stockées dans le **Matlab Workspace**

```
>> C
c=30
```

La représentation graphique de la variable dans le « workspace » indique son genre (matrice, chaîne de caractère, etc.



Name	Value	Min	Max
a	10	10	10
b	5	5	5
c	30	30	30
moyenne	15	15	15

## Affectation et définition de variables (2)

- **Exemples**

```
>> aa = -1.234           % Nombre réel négatif
>> bb = 5/6;            % Nombre réel positif (exécution opération+affect.)
>> cc = 1.234+j*5.678    % Nombre complexe (j : variable prédéfinie de Matlab)
>> cc = 1.234+i*5.678    % Nombre complexe (i : variable prédéfinie de Matlab)

>> dd = cc'              % Complexe conjugué de cc
>> dd = conj(cc)         % Complexe conjugué de cc
```

- **Exercice**

**Comparer les résultats des instructions**

```
>> a = 10, b = 5.38, c = -3.456    % ??
```

```
>> a = 10; b = 5.38; c = -3.456;    % ??
```

```
>> a = 10; b = 5.38, c = -3.456    % ??
```

## Règles de nommage (**A connaître impérativement !!**)

- Matlab est **sensible à la casse** (différence majuscule-minuscule)

```
>> a = 5
```

```
>> A = 5
```

a et A sont 2 **variables différentes** initialisées à la même valeur

```
>> x = 12.3
```

```
>> y = X+9 ⇒ Undefined function X !!!
```

Did you mean:

```
>> y = x+9      % Matlab propose une alternative en fonction des données connues
```

- Matlab n'accepte **pas d'espace, pas de caractères spéciaux** (\$,\*,-,+, etc.) dans les noms :
  - de **variables**
  - de **m-files** et **fonctions** (fichiers .m)
  - (de **répertoires**)
- Matlab est un logiciel américain ⇒ **Ne pas utiliser d'accents**

## Règles de nommage à suivre

- Choisir des noms (variables, fichiers .m, etc.) français  
⇒ évite de choisir un nom déjà connu du logiciel
- **Matrices et vecteurs** débutent par une **majuscule**  
>> Xgps = linspace(0, 500, 500) % ??
- **Scalars** débutent par une **minuscule**  
>> t = 5;
- Utiliser « \_ » pour définir des noms de fichiers .m et de répertoires composés
- Choisir des **noms explicites selon le rôle** de la variable ou du fichier  
regression\_lineaire.m % script matlab réalisant une régression linéaire  
test1.m, toto.m, etc. % A bannir !!

## Scalaires, vecteurs et matrices (1)

### (A connaître impérativement !!)

- **Toute variable** est représentée par une **matrice** (m lignes\*n colonnes)
- Règle 1 : Opérateur de déclaration : **[ ]**
- Règle 2 : Séparateurs de ligne : **;** ou **enter**
- Règle 3 : Séparateurs de colonne : **,** ou **\_** (**espace**)

- **Exemples**

```
>> a = [1.24];           % scalaire avec notation matricielle
>> Vligne = [1, 2, 3, 4, 5]; % Vecteur ligne, séparateur virgule
>> Vligne = [1 2 3 4 5];   % Vecteur ligne, séparateur _
>> Vcol   = [32; -7; 12; 86; 23] % Vecteur colonne, séparateur ;
```

```
Vcol=
```

```
32
-7
12
86
23
```

## Scalaires, vecteurs et matrices (2)

- Exercice : déclaration de matrices

Ecrire

```
M1 = [2 4 6 8; 1 3 5 7; 11 13 15 17]
```

```
M2 = [2  4  6  8
      1  3  5  7
      11 13 15 17]
```

```
V1 = [2 4 6 8];
```

```
V2 = [1 3 5 7];
```

```
V3 = [11 13 15 17];
```

```
M3 = [v1; v2; v3]
```

Comparer M1, M2, M3

- En cas de ligne de commande longue (... à la fin de la saisie d'une ligne)

```
>> M = [1, 3, 5, 7, 9, 11, 13; ...
      2, 4, 6, 8, 10, 12, 14; ...
      1+2, 3+4, 5+6, 7+8, 9+10, 11+12, 13+14];
```



## Définition de matrices particulières (1)

- **Fonctions prédéfinies** (fonctions dites « built-in ») pour la création de matrices

zeros(), ones(), eye(), diag() magic(), pascal(), rand(), etc. (**help elmat**)

- **Syntaxe**

*nom\_fonction*(m,n) % m=nb lignes, n=nb colonnes

*nom\_fonction*(m) % matrice carrée m=n

- **Exemples**

```
>> A = magic(5)           % Matrice carré magique 5*5
>> A = rand(7)           % Matrice 7*7 de nombres aléatoires
>> A = diag([1 3 5 7])   % Matrice diagonale
>> A = [zeros(3) ones(3)] % ??
>> A = [magic(3); ones(3)] % ??
>> A = [zeros(3) ones(3); magic(3) [1:3; 2:4; 3:5]] % ??
```

- **Cf. demos->Mathematics->...**

## Définition de matrices particulières (2)

- **Matrices (ou vecteurs) intégrant une suite logique (1)**

```
>> V = [1 2 3 4 5 6 7 8 9 10]; % Méthode laborieuse!!
```

- **Syntaxe**

```
V = [debut:itérateur:fin]
```

```
V = debut:itérateur:fin % [ ] optionnel ici
```

- **Si itérateur = 1**

```
V = debut:fin % Omission de l'itérateur unitaire
```

- **Exemples**

```
>> A = [0 : 0.2 : 1] % Itérateur réel
```

```
A =
```

```
0 0.2 0.4 0.6 0.8 1
```

```
>> A = 0:10 % Itérateur unitaire omis
```

```
A =
```

```
0 1 2 3 4 5 6 7 8 9 10
```

```
>> A = [1:5; 2:6; 3:7]; % ??
```

## Définition de matrices particulières (3)

- Matrices (ou vecteurs) intégrant une suite logique (2)

- Syntaxe

$v = \text{ linspace}(\text{debut}, \text{fin}, \text{nb\_points})$   
entre debut et fin

% vecteur de points équidistribués

- Attention**

Sélectionner la syntaxe en fonction de l'énoncé !!

### Exemples

Créer un vecteur Temps de 0 à 5s avec un intervalle de 0,01s

⇒  $\text{Temps} = 0:0.01:5;$

Temps : matrice  $1 \times 501$  !!

Intervalle de 0.01s !!

Créer un vecteur Temps de 0 à 5s contenant 500 points

⇒  $\text{Temps} = \text{linspace}(0,5,500);$

Temps : matrice  $1 \times 500$  !!

intervalle de 0.010020040080160s !!

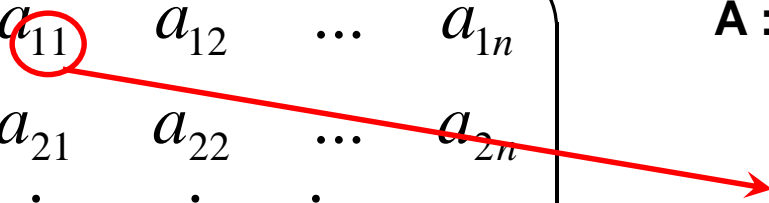
- $\text{logspace}()$  % ??

## Accès aux éléments d'une matrice (1)

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

**A : m lignes \* n colonnes**

**Début de l'indexation à 1**



- Opérateur d'accès : **( , )**
- Syntaxe :
  - Accès à **un élément** : **A(i, k)**  $\Rightarrow$  i<sup>ème</sup> ligne, k<sup>ème</sup> colonne
  - Accès à des **sous-matrices** (ou vecteurs) : **A(i:iterateur:j, k:iterateur:l)**
  - Accès à une **ligne complète** : **A(i, :)**  $\Rightarrow$  éléments de la ligne i
  - Accès à **plusieurs colonnes** : **A(:, k:iterateur:l)**
  - Accès à **des éléments** d'une ligne : **A(i, [k, l])**  $\Rightarrow$  éléments k et l de la ligne i

## Accès aux éléments d'une matrice (2)

- **Exercice**

Créer la matrice A telle que :

$$A = \begin{pmatrix} 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 \\ -1 & 9 & 6 & 13 \\ -6 & 17 & 0 & 4 \end{pmatrix}$$

Expliquer les résultats de ces commandes :

- `>> A(1, 2)`
- `>> B = A(3:4, 3:4)`
- `>> A(1, [3, 1])`
- `>> 2*A(2, :)`
- `>> B = A(1:3,1)`
- `>> A(1:3, 3:4)`
- `>> B = A(1:3,1)'`

## Initialisation d'une matrice

- Solution 1 : cf. définitions de matrices particulières

- Solution 2 : **Affectation explicite**

```
>> C(1, 4) = -8           % Affectation de C14
```

```
>> C(3, :) = 1           % ??
```

```
>> C(:, 5) = [3; 3; 3]    % ??
```

## Suppression de lignes ou colonnes

- Syntaxe : affectation d'une **matrice « vide »**

```
>> D(1, :) = [ ]         % Suppression de la 1ère ligne
```

```
>> D(:, 2:4) = [ ]       % ??
```

```
>> D(1, 5) = [ ]        % ??
```

# Opérations matricielles... calcul matriciel

- **RAPPEL : toute donnée est une matrice**

**OPERATION MATHEMATIQUE = OPERATION MATRICIELLE !!**

- $A*B$                     % Produit matriciel
- $A/B$                     % Division matricielle ( $A*B^{-1}$ ), division à « droite »
- $A \backslash B$                   % Division à « gauche » ( $A^{-1}*B$ )
- $A^3$                     %  $A*A*A$  (A doit être carrée !!)
- $\text{inv}(B)$                 % Inverse de B

- **Exemple**

```
>> v1 = [1; 2; 3]; % Vecteur de type ...
>> v2 = [4, 5, 6]; % Vecteur de type ...
>> A = v1*v2      % A matrice de taille ... A = ...
```

```
>> B = v2*v1      % B matrice de taille ... B = ...
```

## Opérations matricielles... calcul par élément

- Manipulation des éléments d'une matrice
- Opération effectuée **élément par élément**
- Opérateurs concernés

- Multiplication  $\Rightarrow$  **.\***
- Division  $\Rightarrow$  **./** et **.\**
- Exponentiation  $\Rightarrow$  **.^**

- **Règle** : opérateur précédé d'un « . »

- **Exemple**

>> v1\*v1                      % ??

>> v1.\*v1                    % Multiplication élément par élément ( $a_{ij} * a_{ij}$ )

>> v1'.\*v2                   % ??

>> 2.^v1                    % ??



## Opérations matricielles... extension implicite

- **Exercice**

Créer la matrice A telle que :

$$A = \begin{pmatrix} 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 \\ -1 & 9 & 6 & 13 \\ -6 & 17 & 0 & 4 \end{pmatrix}$$

Expliquer le résultat de cette commande :

- **>> A+2**
- <https://fr.mathworks.com/matlabcentral/answers/306689-matlab-2016b-mat-dimensions-for-addition-and-subtraction>
- [https://fr.mathworks.com/help/matlab/matlab\\_prog/compatible-array-sizes-for-basic-operations.html](https://fr.mathworks.com/help/matlab/matlab_prog/compatible-array-sizes-for-basic-operations.html)
- <https://fr.mathworks.com/help/matlab/ref/bsxfun.html>

## Opérations matricielles... **extension implicite**

- Nouveauté depuis Matlab R2016b !!

**Pas de compatibilité avec les versions précédentes !!**

- Que donne ces instructions ??

```
>> clear all; close all; clc;           %...  
>> x=1:3                                %...  
>> y=1:5                                %...  
>> tailleX = size(x), tailleY = size(y) %...  
>> z=x+y'                               %...  
>> tailleX = size(x), tailleY = size(y), tailleZ = size(z) %...
```

## Pour poursuivre la prise en main

- Initiation à Matlab de Laurent OTT :  
[http://www.scalab.cnrs.fr/images/Pages\\_perso/lott/Matlab\\_s1.pdf](http://www.scalab.cnrs.fr/images/Pages_perso/lott/Matlab_s1.pdf)
- Initiation à Matlab pour la résolution de problèmes numériques (Mines ALBI) :  
[http://nte.mines-albi.fr/MATLAB/co/Matlab\\_web.html](http://nte.mines-albi.fr/MATLAB/co/Matlab_web.html)
- Rejoignez la « Matlab Academy » (incluant Matlab On Ramp)...  
<https://matlabacademy.mathworks.com/>

# Place à la PRATIQUE...