Le logiciel de simulation MATLAB Introduction Partie I

Professeur Ali Tahri
Université des sciences et de la technologie d'Oran Mohamed Boudiaf

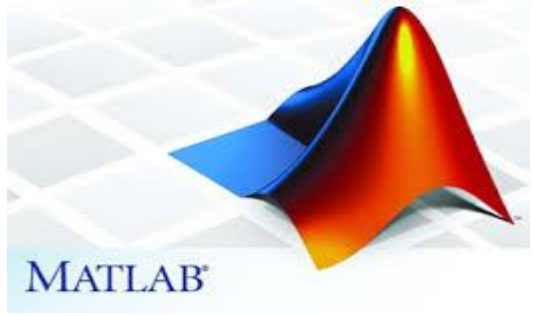
1. Introduction

Bienvenue à l'enseignement supérieur! Si vous voulez réussir, vous devez savoir comment cet endroit fonctionne. Une des principales règles que vous devez savoir est la différence entre les instructeurs que vous aviez avant votre entrée en université et ceux que vous aurez maintenant. Permettez-moi de prendre quelques minutes pour vous expliquer cela.

Fountain of Knowledge at Texas Tech University

Jusqu'à maintenant votre instruction a été dans les mains des enseignants, et le travail d'un enseignant est de vous assurer que vous apprenez. Les enseignants sont évalués sur la base des résultats d'apprentissage, généralement mesurée par des tests standardisés. Si vous n'apprenez pas, alors votre professeur est blâmé. Cependant, les choses sont très différentes pour un professeur d'université. Son travail ne consiste pas à vous faire apprendre. À l'université, l'apprentissage est votre travail - et vous seul. Le travail de votre professeur est de vous conduire à la fontaine de la connaissance. Que vous buvez profondément ou seulement se gargariser est entièrement à vous.

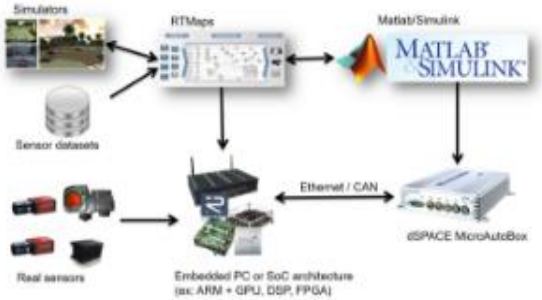
MATLAB (MATrix LABoratory) est un logiciel de programmation est de simulation.
Le logiciel MATLAB implémente le langage MATLAB. Il fournit une très vaste bibliothèque de fonctions pré-définies pour rendre la programmation technique une tâche plus faciles et plus efficaces. Avec cette très grande variété de fonctions, il est beaucoup plus facile de résoudre les problèmes techniques dans MATLAB que dans d'autres langages tels que Java, Fortran ou C ??.



Le plus grand avantage de Matlab est qu'il est facile à utiliser.
Ce cours est une introduction aux facilités que procure Matlab aux étudiants et aux programmateurs.

Aujourd'hui Matlab est utilisé par beaucoup d'entreprises pour le calcul et le prototypage rapide lors du developement de nouveaux systèmes.

Exemples d'entreprises qui utilisent Matlab pour le prototypage rapide



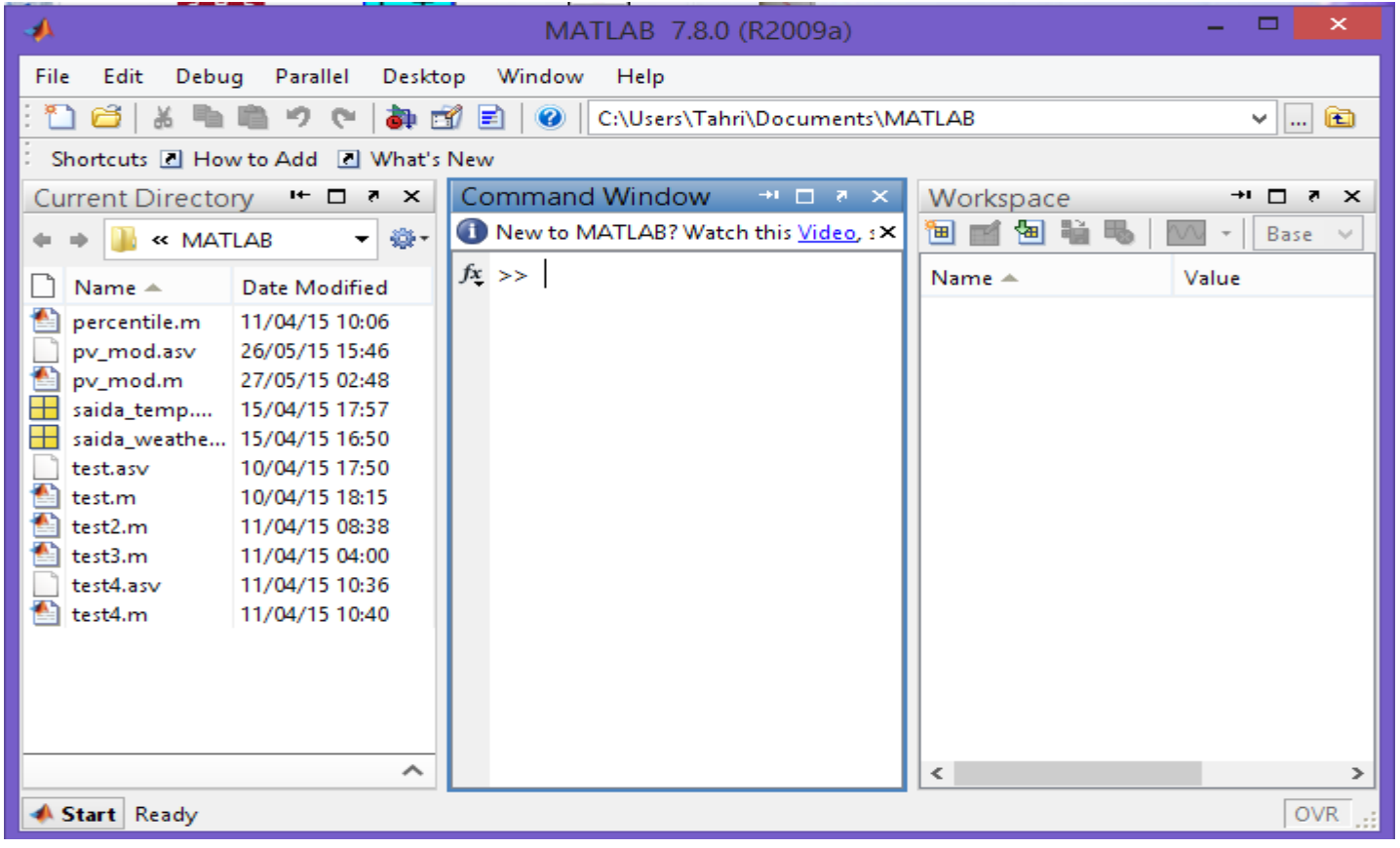
dSPACE: Simulation central

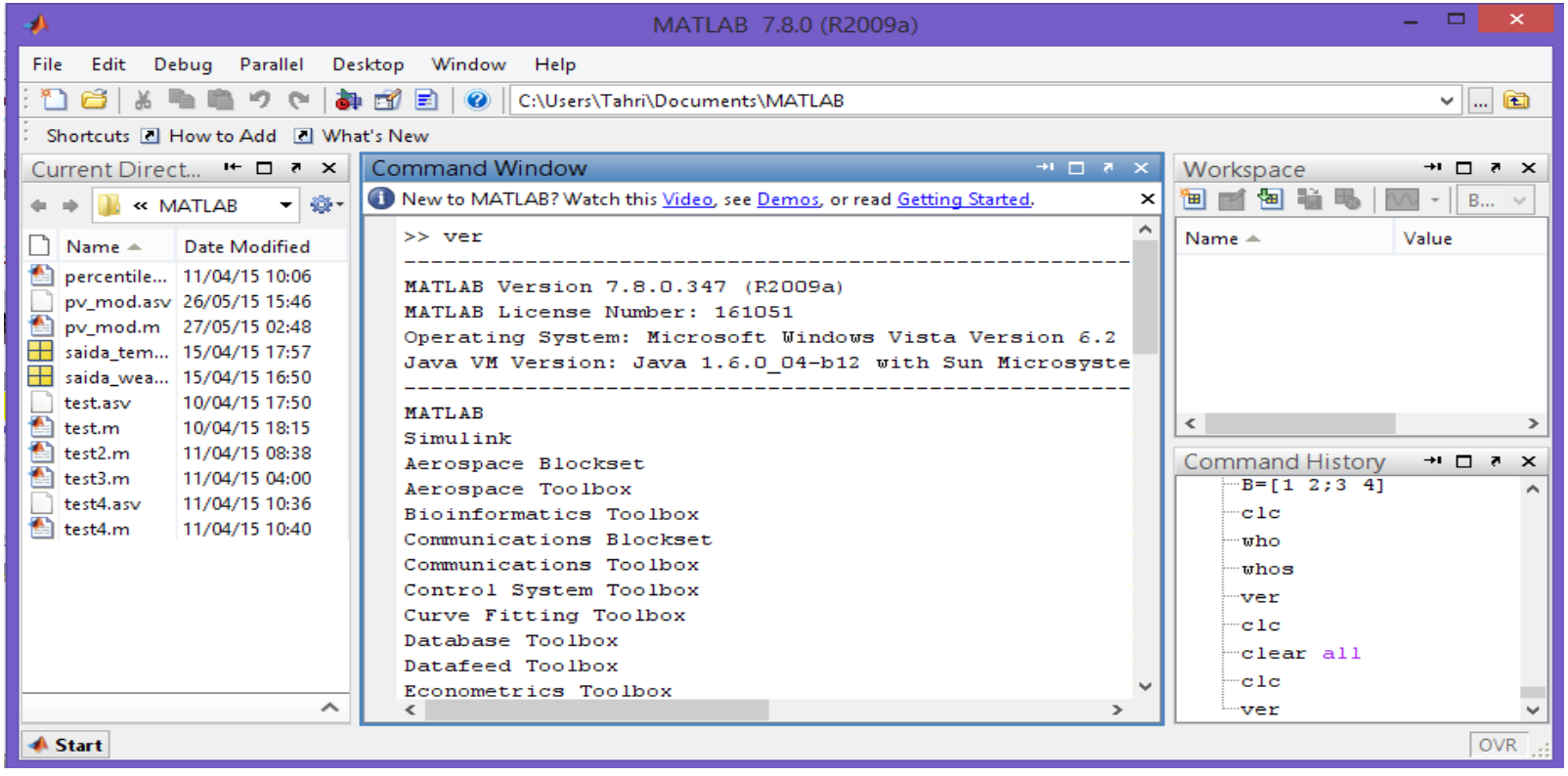
dSPACE's ASMs and ModelDesk bring together all the simulation models the user requires, so they can be run from one user interface

Image: Custom models can be easily included in the open Automotive Simulation Models from dSPACE

Today's development process for automotive control algorithms is highly complex. Modern control algorithms require intensive interaction between distributed functions on different electronic control units (ECUs). A typical development process starts with the PC-based simulation of single control functions and ends with a full-blown ECU network test on a hardware-in-the-loop simulator (HIL). During each development step, functions have to be verified by simulating them together with a model of the device under control (e.g., combustion engine, brake hydraulics, electric motor). dSPACE Automotive Simulation Models (ASM) are the ideal toolkit for this. They are MATLAB/Simulink-based models for simulating essential automotive components and properties, such as combustion engines, electric motors, vehicle dynamics, electrical systems, and traffic for passenger vehicles as well as commercial vehicles.

2. MATLAB


MATLAB 7.8.0 (R2009a)
File Edit Debug Parallel Desktop Window Help
C:\Users\Tahri\Documents\MATLAB
Shortcuts How to Add What's New
Current Directory
« MATLAB
Name Date Modified
percentile.m 11/04/15 10:06
pv_mod.asv 26/05/15 15:46
pv_mod.m 27/05/15 02:48
saida_temp.... 15/04/15 17:57
saida_weathe... 15/04/15 16:50
test.asv 10/04/15 17:50
test.m 10/04/15 18:15
test2.m 11/04/15 08:38
test3.m 11/04/15 04:00
test4.asv 11/04/15 10:36
test4.m 11/04/15 10:40
Command Window
New to MATLAB? Watch this Video,
>>
Workspace
Base
Name Value
Start Ready
OVR

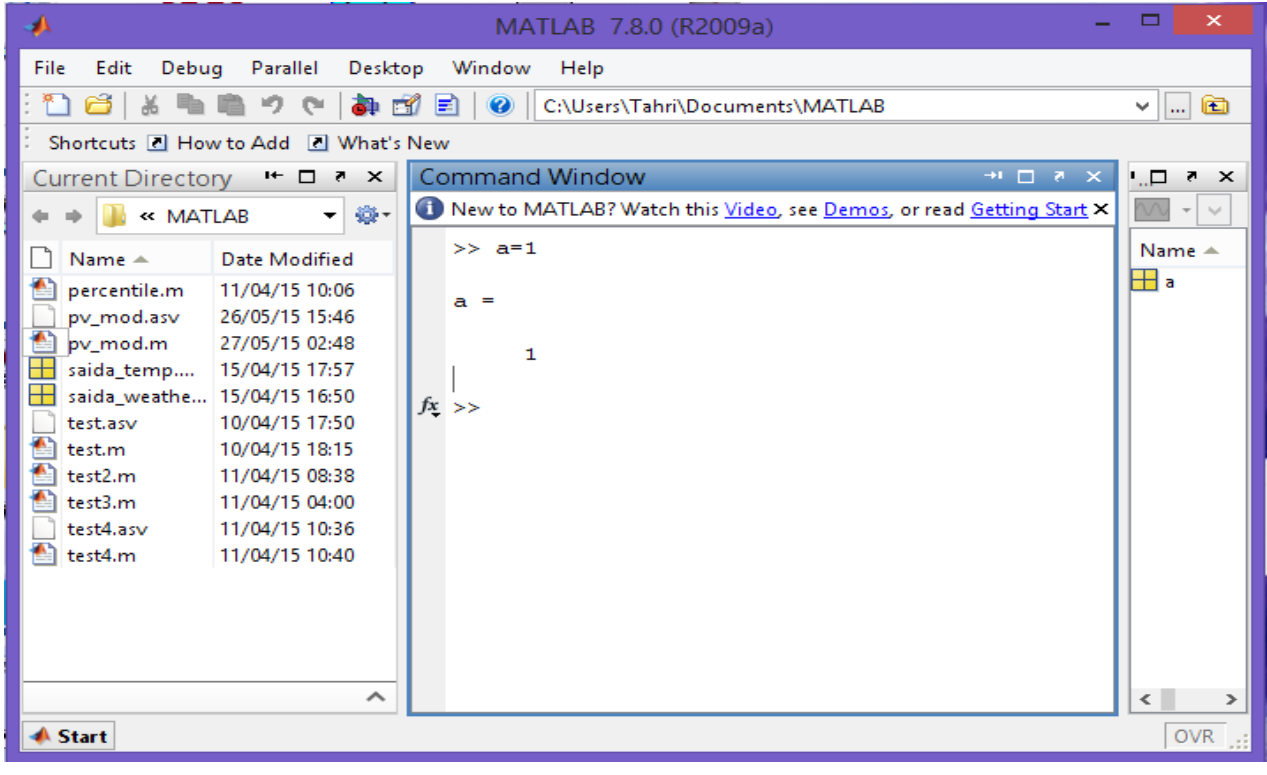


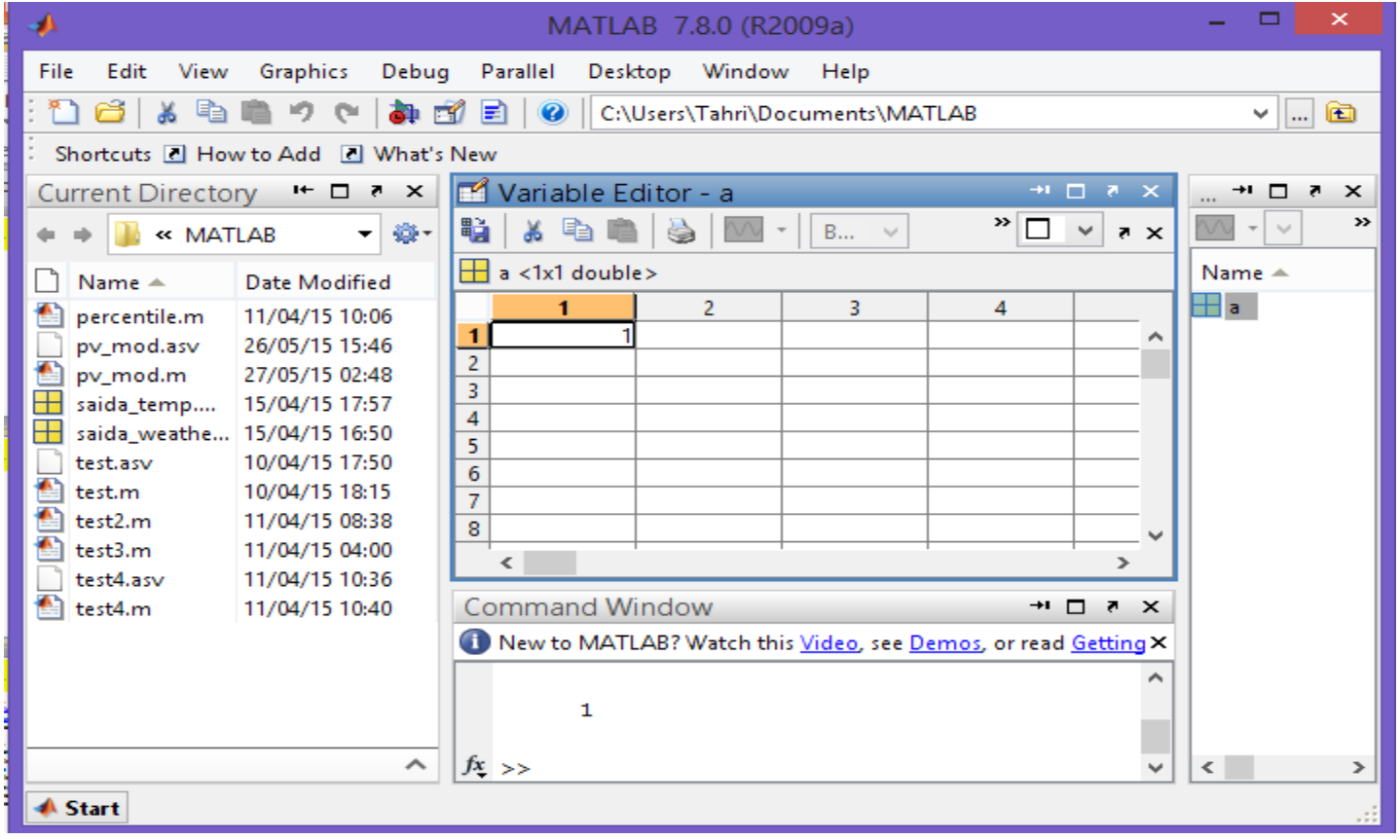
La commande ver permet de vous donner des informations sur la version de la copie du logiciel Matlab ainsi que toutes ses composantes.

2.1 Scalaire

a est un scalaire.

```
>> a=1

a =

     1
```





Après double click sur la variable a, il est évident que la variable a est considérée comme une matrice de dimension (1,1).

2.2 Vecteur

A vecteur de dimension (1,2), une ligne, deux colonnes.

```
>> A=[1,1]

A =

    1    1
```

La commande clc permet d'éffacer l'écran du workspace.

```
>> clc
```

La commande length permet de visualiser la dimension d'un vecteur.

```
>> length(A)

ans =

    2
```

```
>> m=length(A)

m =

    2
```

2.3 Matrice

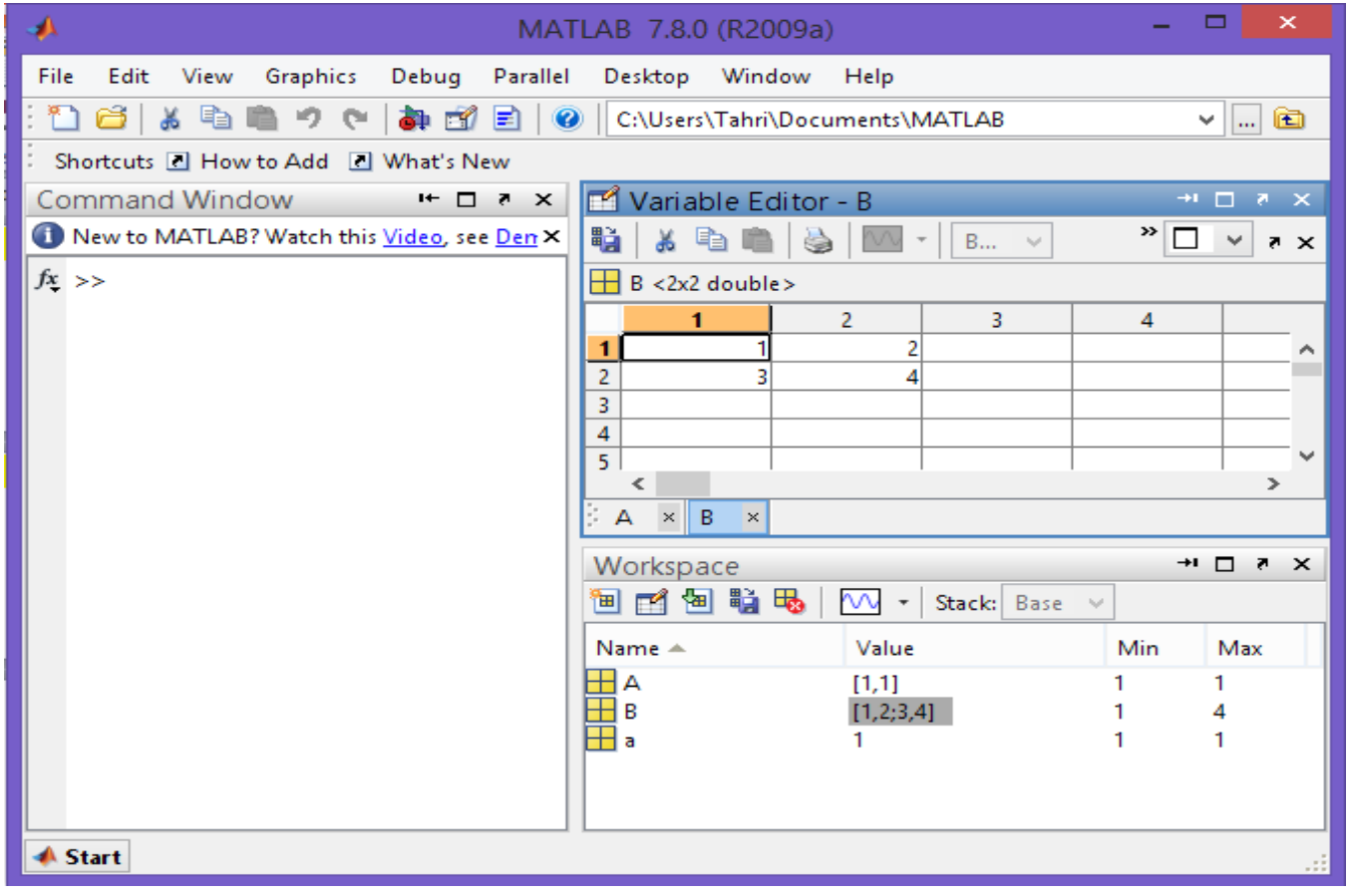
B matrice de dimension (2,2), deux lignes, deux colonnes.

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

B =

    1    2
    3    4
```

La commande size permet de visualiser la dimension d'une matrice.

```
>> n=size(B)

n =

    2    2
```

```
[l,c]=size(B)
l =
     2
c =
     2
```



Remarquer chaque variable est définit dans le workspace

2.4 Fonctions prédéfinies

Toutes les fonctions courantes et beaucoup parmi les moins courantes existent. La plupart d'entre elles fonctionnent en complexe. On retiendra que pour appliquer une fonction à une valeur, il faut mettre cette dernière entre parenthèses.

Exemple :
>> sin(pi/12)
ans =
0.16589613269342

Voici une liste non exhaustive :
- fonctions trigonométriques et inverses : sin, cos, tan, asin, acos, atan
- fonctions hyperboliques (on rajoute "h") : sinh, cosh, tanh, asinh, acosh, atanh
- racine, logarithmes et exponentielles : sqrt, log, log10, exp
- fonctions erreur : erf, erfc
- fonctions de Bessel et Hankel : besselj, bessely, besseli, besselk, besselh.

Matlab dispose de plusieurs fonctions prédéfinies qui facilitent le travail.

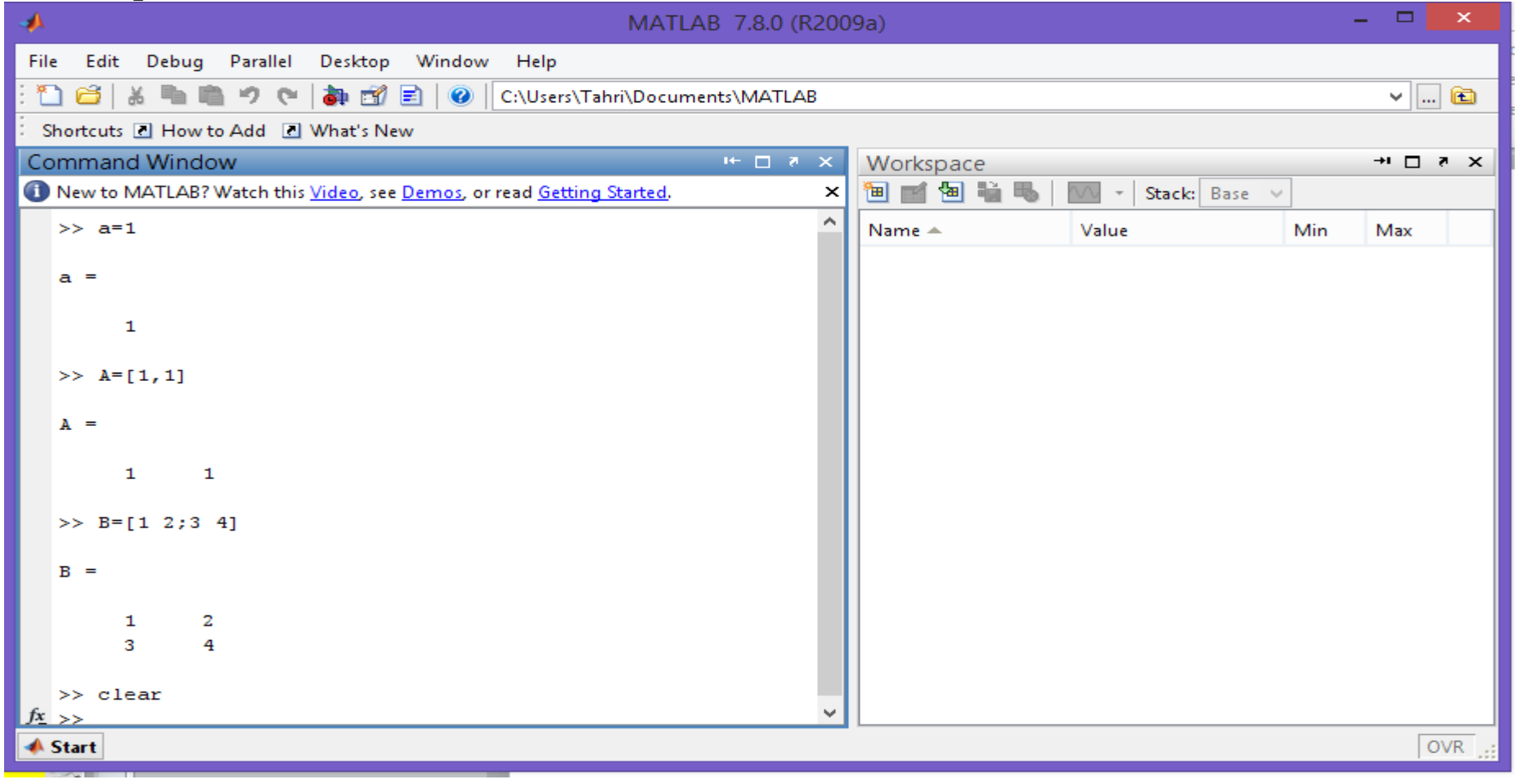
La commande who permet de voir quelles sont les variables définies dans le workspace.

```
>> who
Your variables are:

A  B  a
```

La commande whos permet de voir quelles sont les variables définies dans le workspace, leurs dimensions et leurs natures.

```
>> whos
  Name      Size            Bytes  Class     Attributes

  A         1x2                16  double
  B         2x2                32  double
  a         1x1                 8  double
```

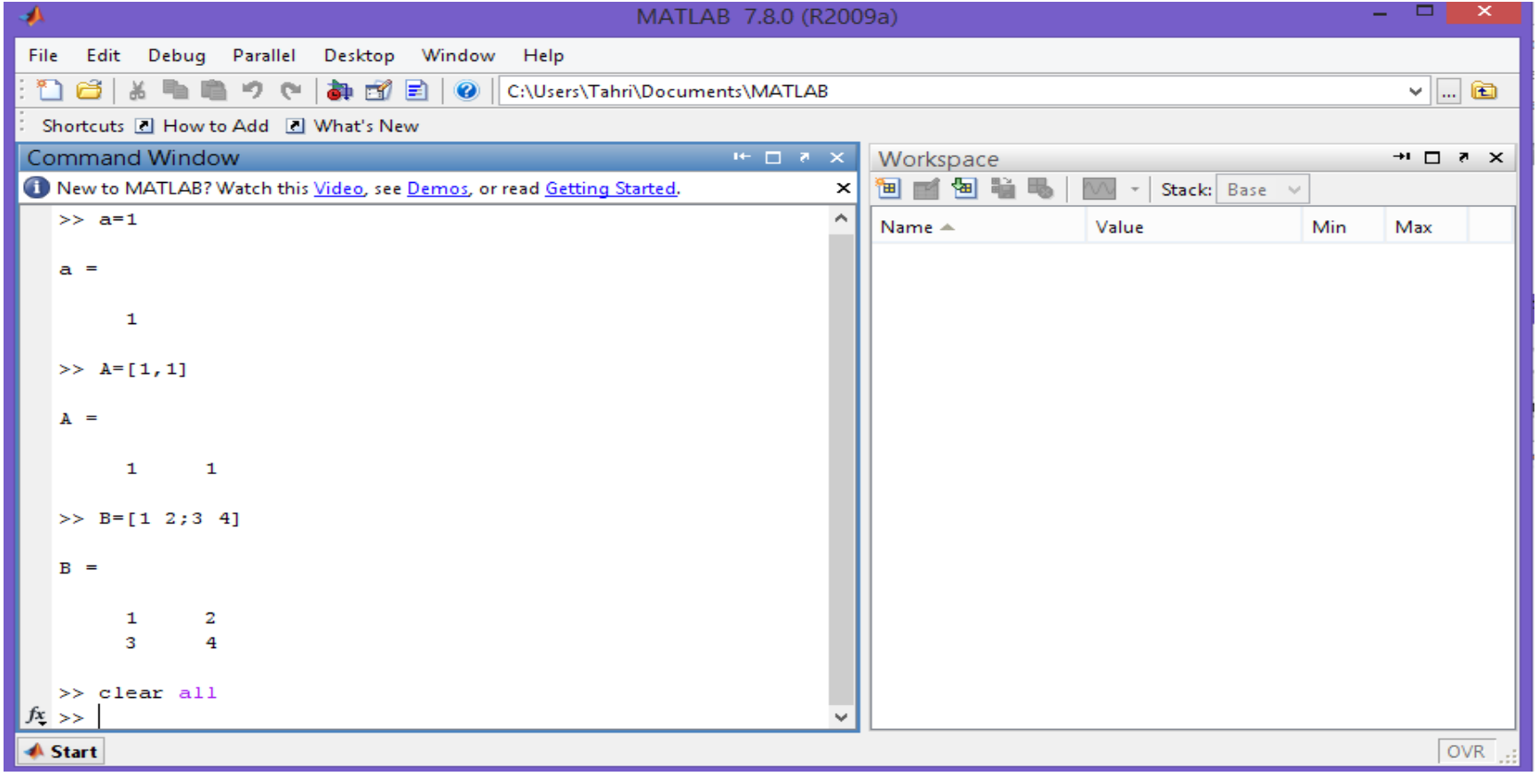
La commande clear permet d'effacer toutes les variables définies dans le workspace.



La commande clc permet d'effacer l'écran du workspace.

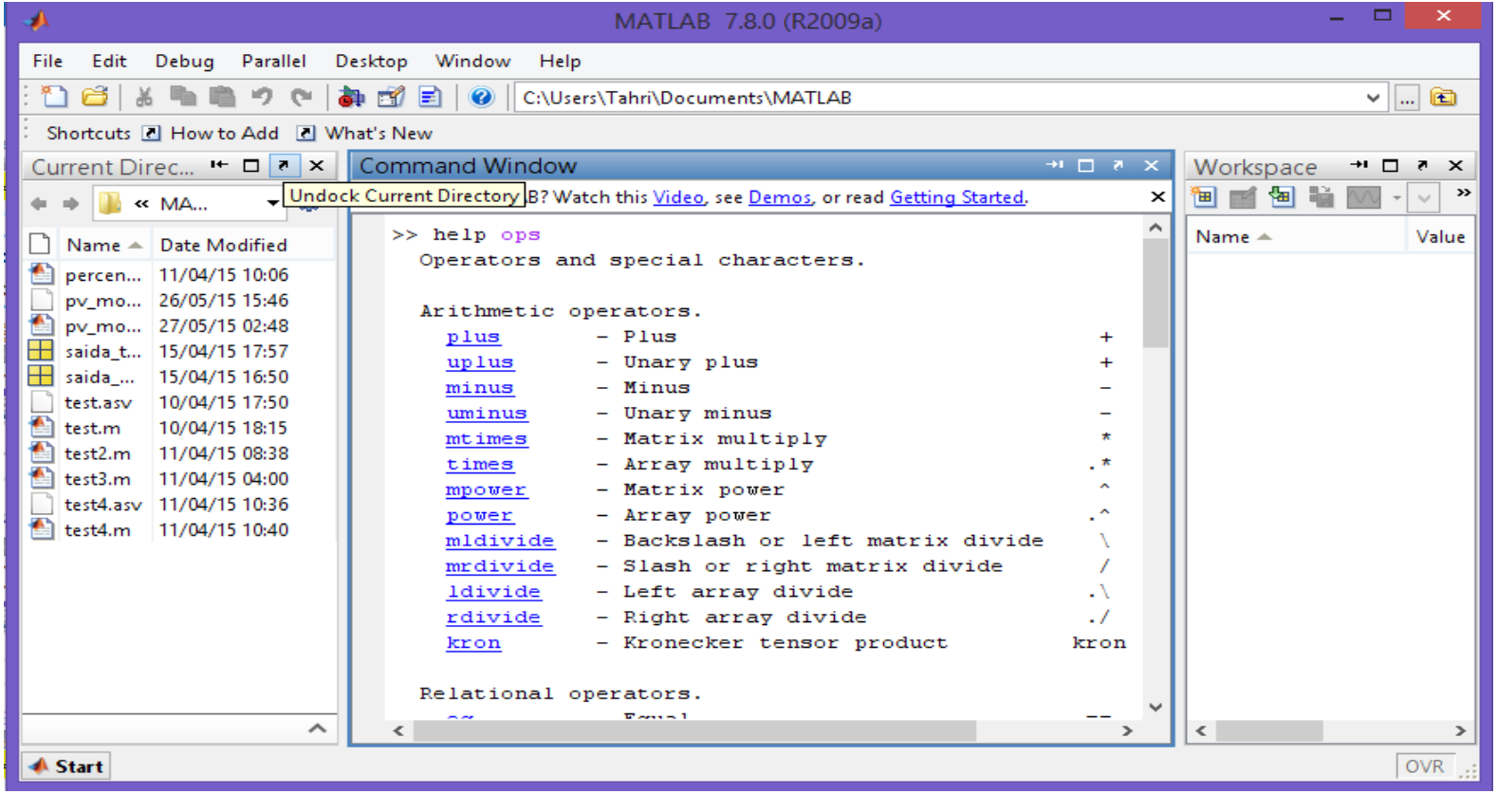
```
>> clc
```

La commande clear all permet aussi d'effacer toutes les variables définies dans le workspace.



2.5 Les opérations

La commande help ops permet de visualiser toutes les opérations qui existent en Matlab.



Operateurs de base

Les opérateurs ci-dessous sont valables pour les scalaires et pour les matrices.
Matlab effectuera la bonne opération en fonction du type des termes de l'opérateur.

```
% — Définitions de matrices spéciales ———————————————————— %

'     % Transposition (matrice)
+     % Addition
-     % Soustraction
*     % Multiplication
/     % Division à droite
\     % Division à gauche
^     % Puissance
```

2.5.1 L'addition

```
>> c=1+2

c =

    3
```

2.5.2 La soustraction

```
>> d=1-2

d =

   -1
```

2.5.3 La multiplication

```
>> e=2*3

e =

    6
```

2.5.4 La division à gauche et à droite

```
>> f=9/2
```

C'est la division à droite.

```
f =

    4.5000
```

Matlab dispose de la division à gauche.

Soit le système linéaire suivant.
Trouver x1 et x2

$$\begin{bmatrix} 3 & 9 \\ 2 & 7 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

```
>> A=[3 9;2 7];B=[1;2];
>> A\B
```

→ La division à gauche

$$A \quad X \quad = B$$

```
ans =

   -3.6667
    1.3333
```

$$A^{-1}AX = A^{-1}B$$

$$X = A^{-1}B$$

2.5.5 La transposition de matrice

```
>> A=[1 2;3 4]
A =
    1    2
    3    4
>> A'
ans =

    1    3
    2    4
```

2.5.6 L'inversion d'une matrice

```
>> inv(A)

ans =

   -2.0000    1.0000
    1.5000   -0.5000
```

2.5.7 Déterminant d'une matrice

```
>> A=[1 2;3 4]
A =
    1    2
    3    4
>> det(A)

ans =

   -2
```

2.6 Opérations élément par élément

Ajouter un point devant un des opérateurs classiques signifie que l'on effectue l'opération élément par élément.

```
% — Opérations élément par élément ————————————————————— %

.+    % Addition
.-    % Soustraction
.*    % Multiplication
./    % Division
.^    % Puissance
```

Plus disponible dans les nouvelles versions de Matlab.

```
>> A=[1 2;3 4]
A =
    1    2
    3    4
>> A.^2
ans =

    1    4
    9   16
```

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

A =
    1    2
    3    4


B =
    5    6
    7    8

>> A.*B

ans =

    5   12
   21   32
```

2.7 Création de matrices particulières

Matlab permet de créer de manière simplifiée des matrices particulières.
m et n sont les dimensions de la matrice

```
% — Définitions de matrices spéciales ———————————————————— %

eye(m,n)          % Matrice unité
ones(m,n)         % Matrice dont tous les éléments sont égaux à un
zeros(m,n)        % Matrice dont tous les éléments sont égaux à zéro
rand(m,n)         % Matrice d'éléments aléatoires
diag(m,n)         % Matrice diagonale
meshgrid(m :n,k :l)  % Renvoie deux matrices de grilles définissant un quadrillage
```

```
>> A=[1 2;3 4]
A =
    1    2
    3    4
>> diag(A)
ans =
    1
    4
```

```
>> eye(2,2)
ans =
    1    0
    0    1
```

```
>> ones(2,2)
ans =
    1    1
    1    1
```

```
>> zeros(2,1)
ans =
    0
    0
```

```
>> rand(2,3)
ans =
    0.1270    0.6324    0.2785
    0.9134    0.0975    0.5469
```

2.8 Chaine de caractères

Matlab enregistre des variables sous forme de chaine de caractères

```
>> nom='Mohamed'

nom =

Mohamed
```

La fonction disp (display) permet de visualiser le contenu d'une variable

```
>> disp(nom)
Mohamed
```

num2str(x) retourne une chaine de caractères qui correspond à un nombre stocké en x

str2num(s) retourne un nombre correspondant à la chaine de caractère s

str2double(s) retourne un nombre correspondant à la chaine de caractère s

length(s) retourne le nombre de caractère dans la chaine de caractère s

lower(s) retourne la chaine de caractères en minuscule

upper(s) retourne la chaine de caractères en majuscule

sName(4) retourne le 4ème caractère de la chaine de caractère s

sName(4:6) retourne du 4ème au 6ème caractère de la chaine de caractère s

3. Constantes prédéfinies

Matlab dispose de constantes prédéfinies, voici la liste.

```
eps       % Floating-point relative accuracy
i         % Imaginary unit
Inf       % Infinity
intmax    % Largest value of specified integer type
intmin    % Smallest value of specified integer type
j         % Imaginary unit
NaN       % Not-a-Number
pi        % Ratio of circle's circumference to its diameter
realmax   % Largest positive floating-point number
realmin   % Smallest positive normalized floating-point number
```

ATTENTION : ces variables ne sont pas protégées, donc si vous les affectez, elles ne gardent pas leur valeur initiale. C'est souvent le problème pour i et j que l'on utilise souvent spontanément comme indices de boucles, de telle sorte qu'on ne peut plus ensuite définir de complexe.

```
>> 1+2i
ans =

   1.0000 + 2.0000i
>> eps

ans =

  2.2204e-016

   >> pi

   ans =

       3.1416
```

```
>> 1/0

ans =

    Inf


>> 0/0

ans =

    NaN
```

4. Le typage de données

Matlab effectue le typage de données, autrement dit il adapte le type des opérateurs de manière à permettre aux opérations de s'effectuer.

```
% — Fonctions de conversion ———————————————————————— %
cast                          % Cast variable to different data type
double                        % Convert to double precision
int8, int16, int32, int64     % Convert to signed integer
single                        % Convert to single precision
typecast                      % Convert data types without changing underlying data
uint8, uint16, uint32, uint64 % Convert to unsigned integer
```

```
% — Fonctions de conversion string vers nombre ———————————————— %
base2dec        %Convert base N number string to decimal number
bin2dec         %Convert binary number string to decimal number
cast            %Cast variable to different data type
hex2dec         %Convert hexadecimal number string to decimal number
hex2num         %Convert hexadecimal number string to double-precision number
str2double      %Convert string to double-precision value
str2num         %Convert string to number
unicode2native  %Convert Unicode characters to numeric bytes
```

Merci

Pour

votre attention