République Algérienne Démocratique et Populaire Ministère de
L'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie d'Oran Mohamed
BOUDIAF faculté de génie électrique

Département d'électronique

L3_Télécommunication

TP1: Prise en main de Matlab: Rappels sur les commandes usuelles

- Aide (**help de Matlab**), Variables, Operations de base, Chaine de caractères, Affichage, Entrée/sortie, Fichiers (script/fonction),
- Mise à niveau pour l'exploitation des boites à outils de Matlab [**Toolbox** /Matlab, signal et **Simulink**].

Prise en main de l'environnement Matlab

Le logiciel Matlab est un logiciel de manipulation de données numériques et de programmation. L'environnement de travail Matlab est souple d'utilisation et évolutif car il permet de travailler soit interactivement en passant des commandes directement au clavier (comme une calculatrice), soit de réaliser des programmes (scripts) ou de définir des fonctions en plaçant ces commandes dans des fichiers texte, le nom de ces fichiers constituant des nouvelles commandes Matlab.

1. Help de Matlab

- Help pour savoir comment on utilise telle ou telle commande il faudra taper :

help nom_de_la_commande

- **Exemple du help** essayez :

help exit, help plot, help fft , help ifft , help sum,…..

2. Les variables dans Matlab, Opération de base

Dans un premiers temps, Matlab peut bien sûr s'utiliser comme une calculette. Ainsi
>> 29+13 , >> 2^3 , >> (((7+3)^2)*9)/8
Puisque Matlab respecte les règles usuelles des mathématiques. Il est aussi possible, comme dans tout langage de programmation, de définir des variables pour stocker un résultat ; ainsi après
>> a=29+13 , >> b=8+7 >> a+b
Si l'on veut éviter que Matlab affiche les résultats intermédiaires, on ajoute simplement un « ; » à la fin de la commande. On vérifiera que
>> a=29+13; >> b=8+7 ; >> a+b
effectue les mêmes opérations que précédemment, mais sans afficher les résultats intermédiaires.
Enfin, les calculs avec les variables complexes sont parfaitement possibles, en utilisant i ou j (avec bien évidement i^2 = j^2 = -1) :
>> a=1+2i ; >> b=2+3i ; >> a*b

Il est utile de savoir quelles sont les variables que nous avons utilisées depuis le début de la session, et pour cela il suffit d'écrire **>> who**
Pour en savoir plus sur elles **>> whos**
Enfin, pour détruire la variable a, on utilise **>> clear a**
ou encore pour détruire toutes les variables, on utilise **>> clear all**
Certaines variables très utiles sont déjà définies, comme **>> pi**
ou **>> ans** qui retourne la dernière réponse donnée par Matlab.
Il est aussi utile de savoir qu'en tapant la flèche ascendante on peut éviter de taper plusieurs fois des commandes identiques.
Pour déclarer une variable de type chaine de caractère, on écrit comme suit :
>> etat1='admis' ; >> etat2='ajourné' ;

3. Les vecteurs en Matlab

Pour définir un vecteur v, par exemple v = [11 22 33 44 55]
, on utilise >> v= [11 22 33 44 55] (vecteur ligne)
La commande >> v'
donne la transposée de ce vecteur, c'est-à-dire ici un vecteur colonne. On peut aussi écrire
>> z =[11 ; 22 ; 33 ; 44 ; 55 ;]
pour obtenir directement un vecteur colonne z, mais il est bien plus simple d'écrire
>> z=v' ou encore >> z=[11 22 33 44 55]'
pour obtenir z
Il est aussi possible d'utiliser ces notations avec notre opérateur « : » pour définir un nouveau vecteur.
Par exemple si l'on écrit [1 :1 :5], on définit un vecteur allant de 1 à 5 avec une incrémentation unité (un pas de 1)
.Ainsi >> w=[1 :1 :5] ou encore >> w=[1 :5]
nous permettent de définir le vecteur w = (1 2 3 4 5)
Les mêmes opérations sont bien sûr possibles sur les vecteurs colonnes, par exemple
>> vec=[1 :2 :10]' définit le vecteur vec
Pour accéder à une valeur particulière d'un vecteur, on écrira par exemple
>> v(2)
qui donne simplement accès au 2em élément, c. à d. ici la valeur 22. Il est aussi possible d'utiliser les « : » pour sélectionner plusieurs éléments d'un vecteur ! Voici quelques exemples :
>> v(1 :3)
donne accès aux éléments qui vont de 1 à 3 (avec un pas de 1 sans plus de précision, donc les éléments 1, 2 et 3). Cette commande retourne le vecteur (11 22 33). On peut aussi écrire
>> v(1 :2 :5)
qui donne accès aux éléments qui vont de 1 à 5, mais en incrémentant avec un pas de 2. Dans ce cas, le vecteur (11 33 55) est retourné. Si l'on écrit
>> v
tout le vecteur est retourné, mais l'on obtient le même résultat par
>> v(:)
le symbole « : » sans autres précisions signifiant tout le vecteur.

4. Toolboxes

Ce sont des boîtes à outils. Celles-ci sont des bibliothèques de fonctions dédiées à des domaines particuliers. Nous pouvons citer par exemple : l'Automatique, le traitement du signal, l'analyse statistique, l'optimisation....

Quelque Exemples des Toolbox:

- Control System Toolbox
- Symbolic Math Toolbox
- Signal Processing Toolbox
- Image Processing Toolbox
- Aerospace Toolbox
- Bioinformatics Toolbox
- Financial Toolbox

Exemple : Le toolbox "signal processing" de Matlab comporte une collection très riche de fonctions de traitement du signal. On peut citer les sous familles suivantes :

- Transformées : fft, hilbert, ifft, …
- Analyse et implémentation de filtres : FIR, IIR, …
- Translation fréquentielle (modulation),
- Traitement statistique du signal : corrcoef, cov, xcorr,,…
- Modélisation paramétrique de série temporelle : ar, arima, arburg, arcov, aryule, ident, …

5. Simulink

Simulink est un logiciel muni d'une interface graphique pour la modélisation, la simulation et l'analyse des systèmes dynamiques. Etant intégré à MATLAB, les deux environnements sont parfaitement compatibles et les différentes fonctionnalités de ce dernier sont alors directement accessibles. Simulink est basé sur une interface graphique qui permet une construction aisée et conviviale de schémas-blocs. Chaque bloc composant le système est sélectionné depuis un ensemble de bibliothèques prédéfinies.

L'utilisation du SIMULINK suit en général les étapes suivantes :

1. Etablir (dessiner) le modèle du système en utilisant les blocs présent en librairie ;
2. Placer des sources de signaux aux entrées du modèle : générateurs numériques ou analogiques
3. Placer des instruments de visualisation en sortie du modèle : Scope,…
4. Paramétrer et lancer la simulation du fonctionnement du modèle : double clic sur le bloc puis modifier les paramètres et Simulation/Start pour lancer la simulation ;
5. Observer les résultats à l'aide des instruments de visualisation

- **Exemple 1: Création et simulation d'un simple modèle sur Simulink**

  Réaliser le montage suivant puis visualiser ces deux signaux : signal sinusoïdal sans et avec amplification.

  A1=3 avec gain égale à 3.

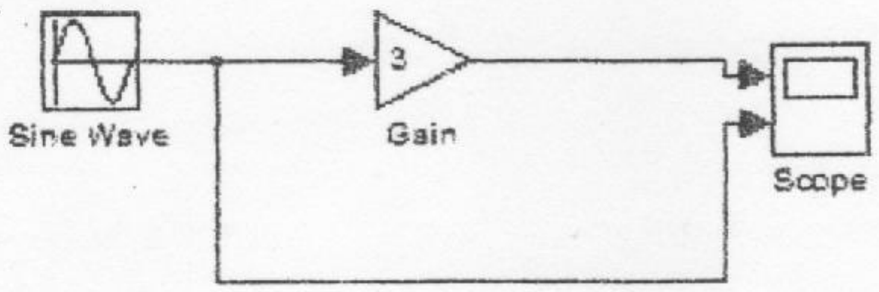


Figure 1 montage de l'exemple 1

- **Exemple 2: Simulation et visualisation de différents signaux (sinusoïdal, rampe et échelon) sur Simulink**
  Réaliser le montage suivant puis visualiser ces trois signaux : sinusoïdal, rampe et échelon.

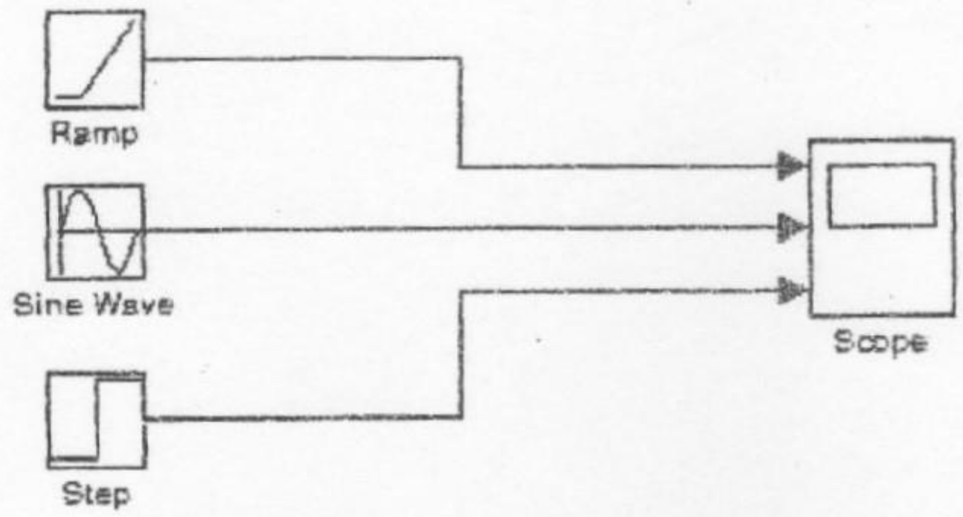


Figure 2 montage de l'exemple 2

6. Affichage des graphes

La fonction plot est utilisée pour tracer le graphe des valeurs d'un tableau (y) en fonction d'un autre tableau (x):

```
>> x = 0 : pi/90 : pi ;          >> y = sin(x) ;          >> w=cos(x);
>>plot(x,y)
```

Pour afficher la grille:

```
>> grid
```

Pour donner un label à l'axe de x :

```
>> xlabel('x')
```

Pour donner un label à l'axe de y:

```
>> ylabel('sin(x)')
```

Pour donner un titre au graphe

```
>> title('la fonction sinus')
```

Pour tracer deux ou plusieurs signaux sur le même graphe, on utilise

```
>> hold on
```

Pour la représentation discrète du signal continu on utilise la fonction suivante :

```
>>stem(x,y)
```

Afin de différencier entre plusieurs graphes sur la même figure, on peut changer la couleur de notre tracé comme suit : `>> plot(x,w,'k')`

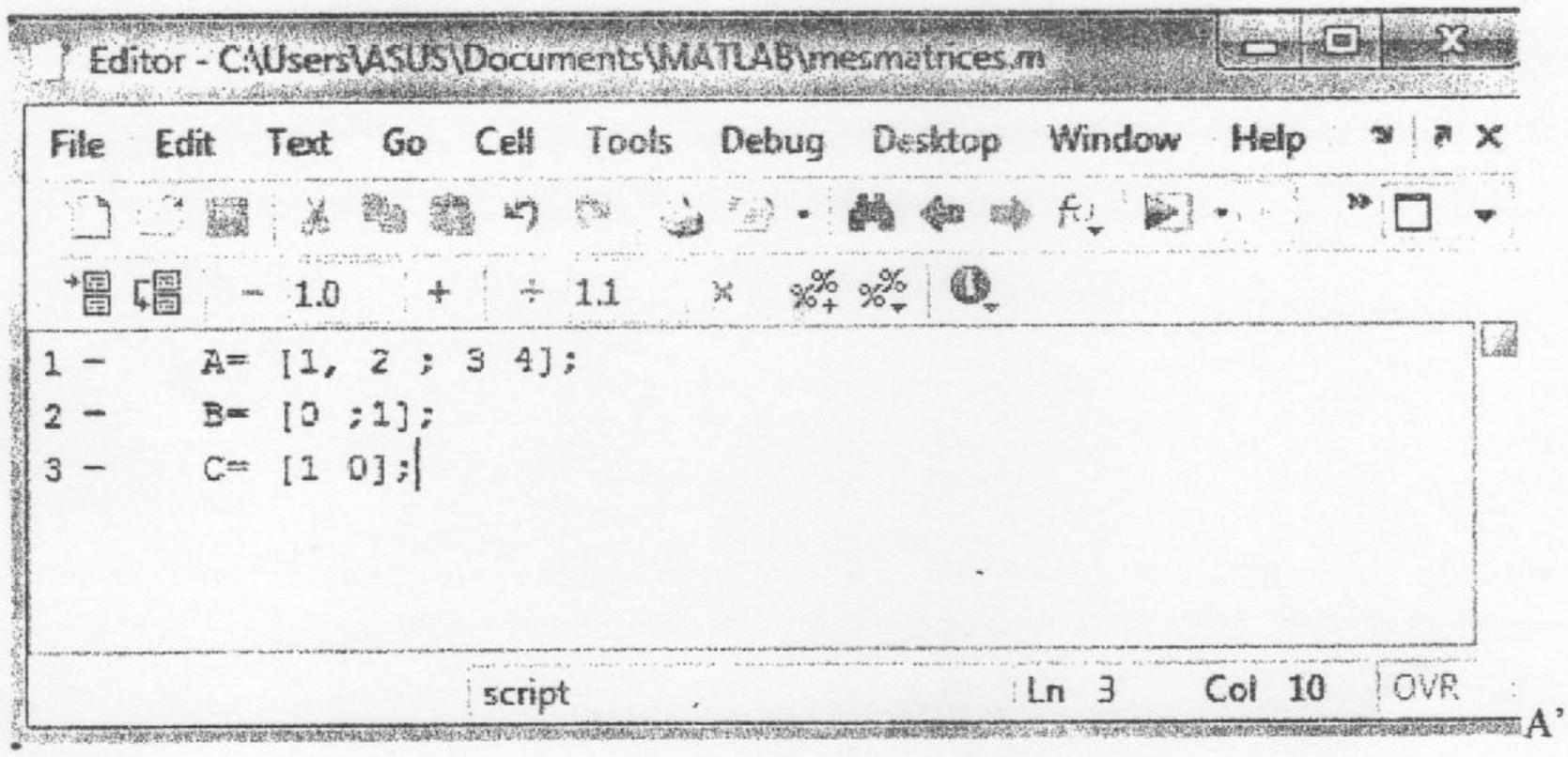
Avec k : noir, r : rouge, b : bleu,.....(voir sur le help)

7. Editer un script

On peut placer une suite d'instructions MATLAB dans un script (fichier d'extension .m).

Exemple d'un simple script :

Créer le script suivant puis sauvegardé le en le nomant mesmatrices.



A'

Pour exécuter, dans la fenêtre MATLAB, faire >>mesmatrices

Puis essayer A, A', B, ... ces matrices sont dans le **workspace** de Matlab