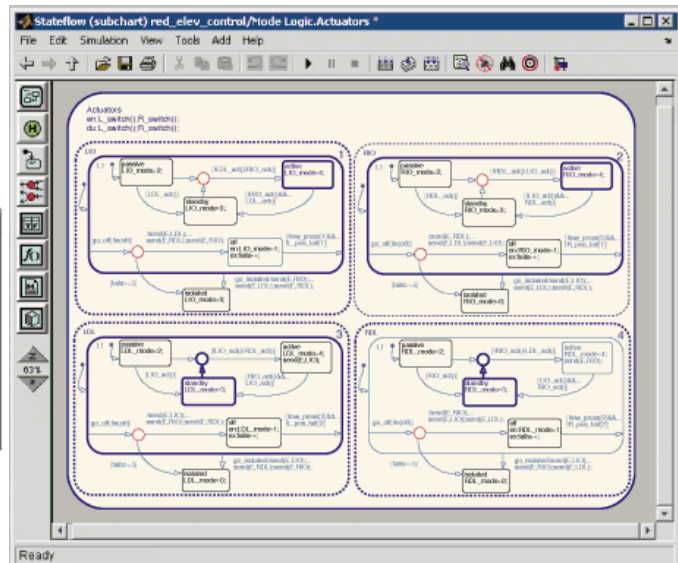
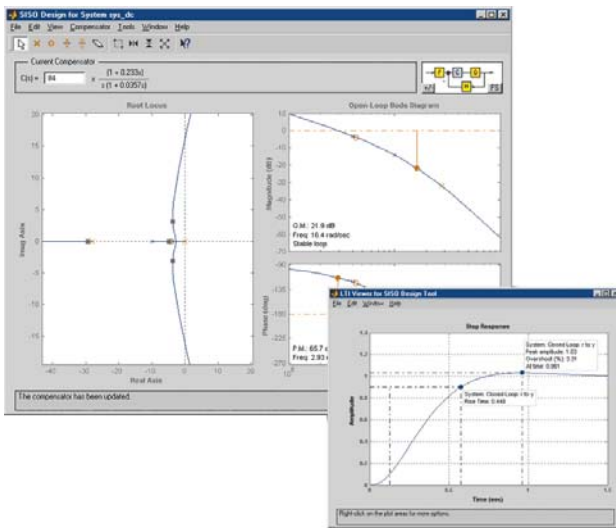
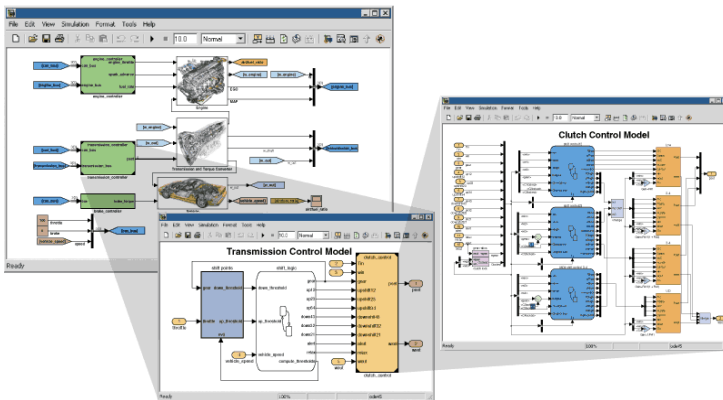
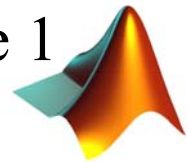


# Simulation des systèmes continus

## MATLAB<sup>®</sup> Séance 1



INSA de Lyon  
3GI  
2005-2006  
Lounis ADOUANE



## Préambule

Savoir maîtriser un outil de simulation pour systèmes physiques, économiques ou statistiques est un atout que tout ingénieur doit détenir. Le but de ces deux séances de TP est de vous initier à l'un de ces outils puissants et performants (en l'occurrence à MATLAB®). Bien évidemment, en deux séances nous n'allons pas pouvoir faire le tour de l'ensemble des fonctionnalités de MATLAB, néanmoins le TP va tâcher de vous orienter (via des questions et des exercices appropriés) vers les principaux outils et méthodes qu'offre MATLAB. Le but est de cerner globalement ses potentialités et sa philosophie de fonctionnement, afin de pouvoir aisément juger de sa pertinence lors de vos développements futurs de projets académiques ou professionnels.

## Pourquoi MATLAB® ?

Le logiciel MATLAB, contraction de MATrix LABORatory est un environnement de calcul numérique de haut niveau. Il est devenu de nos jours un standard pour la recherche scientifique et l'ingénierie, d'ailleurs, MATLAB est la référence mondiale dans le domaine du calcul technique. Il peut être utilisé autant pour le prototypage et le test rapide d'application que pour des applications plus sophistiquées et élaborées. En effet, il met à la disposition de l'utilisateur un environnement convivial et performant pour mener à bien des calculs numériques ou symboliques, obtenir des représentations graphiques et écrire des programmes. Des problèmes numériques complexes peuvent être ainsi résolus bien plus rapidement qu'avec des langages de programmation comme le C ou le FORTRAN.

L'architecture ouverte de MATLAB facilite son utilisation, la figure 1 nous donne un aperçu des différents éléments le composant. Le noyau MATLAB peut être complété par des boîtes à outils (Toolboxes) qui sont constituées de bibliothèques de fonctions spécialisées propres à un domaine particulier (voir figure. 1). On trouve des boîtes à outils dans des domaines aussi variés que la commande des systèmes, le traitement du signal, la chimie, la finance ou l'économie. Simulink est une boîte à outils particulière puisque elle permet de traduire la résolution des systèmes (principalement différentiels) de manière graphique à l'aide de schémas blocs. MATLAB est principalement utilisé sous sa forme d'interpréteur de commandes, il est toutefois possible de compiler les programmes afin d'améliorer la vitesse d'exécution.

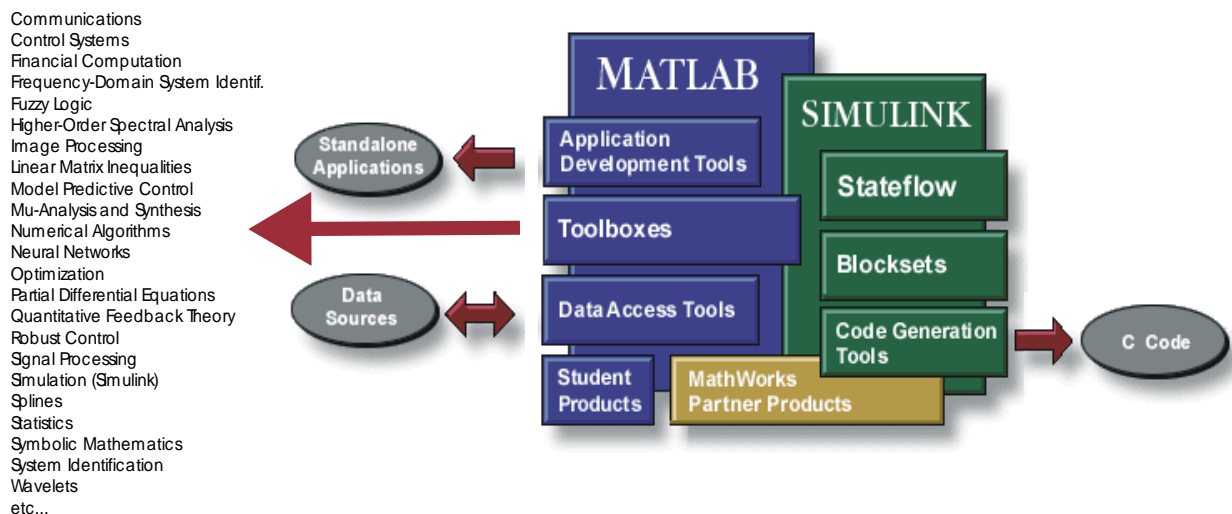


Figure. 1

## Principales fonctionnalités

1. du **calcul numérique** pour des résultats rapides et précis,
2. des **graphiques** pour visualiser et analyser des données,
3. un **langage** et un **environnement de programmation** interactifs,
4. l'outil **Simulink**, pour modéliser et simuler des systèmes en utilisant des schémas-blocs,
5. des outils pour concevoir des **interfaces utilisateur graphiques** (GUI pour « Graphical User Interface ») personnalisées,
6. la prise en charge d'**échange de données et de signaux** avec plusieurs type de matériels (oscilloscope, carte d'acquisition, carte de contrôle, etc.),
7. l'**intégration d'applications externes** écrites en C, C++, Fortran, Java, Excel, etc.
8. la **conversion d'applications** MATLAB en C et C++ avec Compiler.

Les thèmes qui seront abordés au cours des 2 séances de TP toucheront aux points allant de 1 à 5. Ces points couvrent l'essentiel de ce qu'il faut connaître sur MATLAB pour se rendre compte de ses potentialités et ainsi l'utiliser efficacement.

Vous allez d'abord aborder les fonctions élémentaires de MATLAB dans les domaines du calcul vectoriel et matriciel, de la représentation graphique et de la programmation. Ensuite vous découvrirez Simulink ainsi que les GUI. Vous allez utiliser succinctement aussi les boîtes à outils concernant l'automatique et les statistiques.

## I DECOUVERTE



Pour lancer le logiciel double-cliquez sur l'icone de MATLAB :

### I.1 Environnement graphique :

#### Espace de travail :

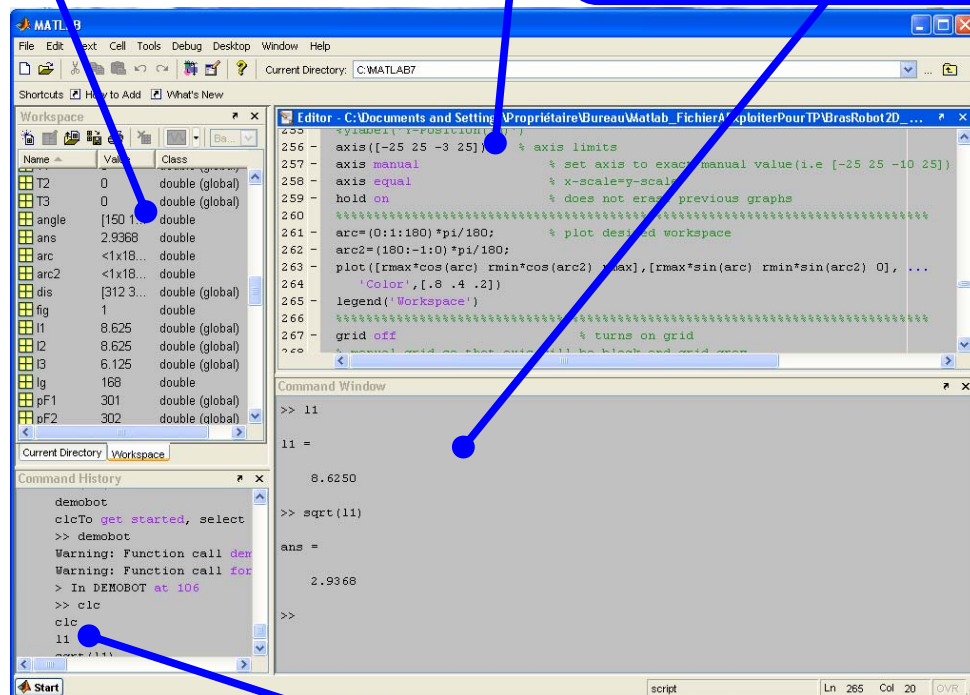
Toutes les variables définies seront listées ici. Un double clic sur une ligne permet d'accéder à la valeur de la variable.

#### Programme MATLAB :

Fichier « .m » qui correspond à un programme écrit en MATLAB.

#### Fenêtre de commande :

MATLAB est construit autour d'un langage interprété. Les calculs se font donc d'abord par la ligne de commande...



#### Historique des commandes :

Le double clic permet d'exécuter une commande.

La mise au point de calculs complexes, l'analyse de données ou de modèles nécessite souvent de tester d'abord les commandes dans la fenêtre de commande. Cet historique permet ensuite de retrouver ces commandes afin de les copier dans un fichier « .m ».

#### Bouton Start :

Il permet de lancer rapidement de nombreuses fonctionnalités. Pour lancer l'aide : Start -> Help

## I.2 L'aide :

Il y a plusieurs façons d'obtenir des informations sur les différentes composantes et fonctionnalités de MATLAB, parmi elles nous citons :

### a- À partir de la fenêtre de commande :

- >> help help % instruction pour savoir comment obtenir de l'aide.
- NB :** - Lorsque le symbole ">>" apparaît à gauche de la fenêtre de commande alors cela indique que l'interpréteur est prêt à recevoir une commande.
- Tout ce qui est après le symbole «%» indique un commentaire sous MATLAB.
- >> help % liste les thèmes pour lesquels une aide en ligne est disponible.
- >> help thème % liste les fonctions relatives au thème mentionné pour lesquelles une aide est disponible.
- >> help nom\_fonction % présente la page d'aide concernant la fonction choisie.
- >> lookfor mot\_clé % liste les commandes et fonctions MATLAB contenant le mot\_clé demandé dans leur % documentation.
- >> helpdesk % obtient une documentation en format HTML.
- >> helpwin % présente une fenêtre de navigation (figure.2) pour une aide interactive. La partie gauche de la % fenêtre permet, à partir d'onglets, de rechercher de l'aide en fonction du contenu thématique % (Contents), de la liste alphabétique des thèmes et mots clés (Index) ou à partir d'un moteur de % recherche (Search). Un onglet permet également d'accéder à des démonstrations classées par % thèmes (Demos).

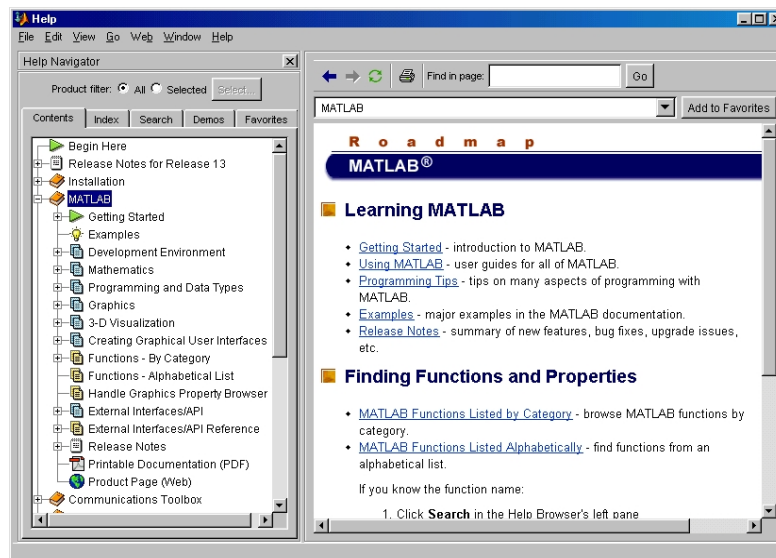



Figure. 2

**b- À partir d'Internet :** plus particulièrement à partir du site de MathWorks<sup>1</sup>. Il existe aussi des démonstrations en ligne intéressantes au niveau de l'INSA de Lyon, pour cela il faut se rendre sur le site du Centre de Mathématique <http://maths.insa-lyon.fr/> rubrique : Enseignant/Matlab-Wserv.

## II TRAVAIL A RENDRE

Après la fin des deux séances de TP vous devez me rendre un compte-rendu sous format papier (1 par binôme). Ces documents doivent absolument être déposés dans mon casier (secrétariat GI) la semaine qui suit la deuxième séance de TP.

A chaque fois que l'icône  apparaît à gauche du texte, ceci signifie qu'une explication sur la syntaxe, que la description d'une commande ou qu'un programme à réaliser doit figurer dans votre compte-rendu. Ceux-ci doivent être donnés dans l'ordre avec l'indication du numéro noté à côté de l'icône. Vous devez également me fournir les principaux programmes (.m), (.mdl), etc. que vous allez réaliser (à m'envoyer à mon adresse email avec comme objet : TP MATLAB [Vos noms]).

<sup>1</sup> <http://www.mathworks.com/> (société qui développe et qui commercialise MATLAB)

### III PREMIERS PAS

#### III.1 Calcul matriciel

MATLAB est particulièrement performant pour le calcul matriciel car sa structure interne de données est basée sur les matrices. N'importe quelle chaîne de caractères exceptés celles contenant des symboles mathématiques (tels que : espace, :, /, %, etc.) peut être utilisée pour nommer une variable. L'utilisation d'une variable sous MATLAB ne nécessite de définir ni son type (booléen, entier, réel, etc.) ni même sa dimension (scalaire, vecteur, matrice) et cela contrairement à la plupart des langages de programmation tels que le C, le PASCAL ou le FORTRAN. Ainsi, une variable dans MATLAB peut être définie au départ comme un scalaire, puis devenir un vecteur, et finir comme matrice. Il est à noter que MATLAB distingue les majuscules des minuscules (la variable 'a' est différente de 'A').

Quelques fonctions utilisables à partir de la matrice  $A = \begin{bmatrix} 0 & 0 & 1 & 5 \\ 4 & 5 & 2 & -1 \\ 3 & -2 & 1 & 4 \\ -2 & -4 & 7 & 2 \end{bmatrix}$

- Entrez la matrice A. Deux méthodes sont possibles :

```
>> A=[0 0 1 5 ; 4 5 2 -1 ; 3 -2 1 4 ; -2 -4 7 2]
```

ou alors :

```
>> A=[0 0 1 5
      4 5 2 -1
      3 -2 1 4
      -2 -4 7 2]
```

- Saisir la même valeur de A (flèche ↑ du clavier) en y ajoutant un point virgule à la fin « ; ». Qu'est-ce qui change ?



1

- Expliquez les commandes suivantes :

```
>> B=A(2,:)
>> C=B*A
>> D=A.*A
>> A(:,2)
>> B=A ; B(5,1)=10
>> C= 0 : 0.2 :1
```

- Définissez la matrice B telle que  $B=A+2 \times I$  (avec I matrice identité) :

```
>> B=A+2*eye(4)
```



2

- Utilisez l'aide pour expliquez la syntaxe et la signification de la commande eye ?

- Définissez C la matrice inverse de B :

```
>> C=inv(B)
```

Remarque : Par défaut l'affichage des valeurs se fait avec une précision de  $10^{-4}$ . L'instruction *format* permet de changer cette précision. Par exemple :

```
>> format bank
>> C
>> format short
>> C
```

- Calculez D le déterminant de C :

```
>> D=det(C)
```

- Définissez E la transposée de C :

```
>> E=C'
```



3

- Que font les commandes « a = zeros(n) » et « b = ones(n) » où n est un entier naturel ?



Tout polynôme peut s'écrire sous la forme d'un vecteur ligne contenant les coefficients du polynôme rangés par puissance décroissante.

Par exemple le polynôme  $P = x^3 - 6x^2 - 72x + 27$  a pour écriture :

```
>> P=[1 -6 -72 27]
```



4

- Expliquez le rôle de la commande suivante. Donnez le résultat obtenu :

```
>> roots(P)
```

```
>> polyval(P,3)
```

### III.2 Graphiques

MATLAB, en plus de ses possibilités de calcul numérique, produit aussi des graphiques en 2 et 3 dimensions de haute qualité. Nous allons nous focaliser dans ce qui suit sur les graphiques 2D.

Il est très facile de représenter graphiquement les résultats de calculs.

- Pour dessiner une courbe il faut d'abord déterminer les points de l'axe des x pour lesquels les valeurs de la fonction seront calculées. Créez un vecteur :

```
>> x = 0 : pi/4 : 5*pi
```

- Pour chaque valeur de x défini précédemment il faut ensuite calculer la valeur de la fonction :

```
>> y = sin (x/pi)
```



5

- Tracez une courbe en utilisant les commandes :

```
>> plot(x,y); grid on;
```



6

- En utilisant la fonction « polyval » confirmez graphiquement la position des racines du polynôme P défini ci-dessus.

- Par défaut une seule courbe peut être tracée sur une figure. Pour tracer plus de figures dans une même fenêtre, veillez à ne pas fermer la fenêtre contenant la figure précédente et tapez les commandes suivantes :

```
>> hold on
```

```
>> z = y + 0.2
```

```
>> plot(x,z)
```

La commande « plot » permet aussi de tracer directement plusieurs courbes et de spécifier leurs propriétés :

```
>> hold off
```

```
>> plot(x,y,'r+',x,z,'bo')
```



7

- Utilisez le raccourci « Show Plot Tools » afin de :

- donner un titre : « premières fonctions »,
- donner un nom à chaque axe : « abscisse » et « ordonnée »,
- donner une légende donnant la définition de chaque courbe.



Show Plot Tools

- changer les propriétés de la courbe la plus haute :

- Line properties : style = solid line, width = 2.0, color = green
- Marker properties : style = diamond, size = 6.0, edge color = black, face color = yellow



8

Afin d'insérer la figure précédente dans votre rapport :

- Menu « File » → « Save as ... » puis sélectionnez le format, le répertoire et le nom de fichier, ou bien :

- Si vous utilisez Word :

- dans la fenêtre MATLAB « Figure No. 1 » : Menu « Edit » → « Copy Figure »
- dans Word : Menu « Edition » → « Coller »

Les modifications qui ont été faites sur la figure peuvent être réalisées avec des commandes. Par exemple :

```
>> title('titre de la figure')
>> xlabel('abscisses')
>> ylabel('ordonnées')
>> legend('courbe1','courbe2')
```





9

- Donnez les commandes permettant de créer plusieurs fenêtres de figures puis de choisir dans quelle fenêtre sera dessinée une courbe.

### III.3 Programmation

Jusqu'à maintenant vous avez utilisé uniquement le *mode interactif* de MATLAB, qui exécute les instructions au fur et à mesure qu'elles lui sont données par l'utilisateur dans la fenêtre de commande. Néanmoins, MATLAB dispose aussi d'un *mode exécutif* qui permet d'exécuter un programme inclus dans un fichier appelé un m-file (en relation à son extension « .m »). Il en existe deux types, les *scripts* et les *fonctions*. Ces fichiers vous permettront ainsi d'exécuter plusieurs fois les commandes sans avoir à les entrer à chaque fois.

- Écrivez un premier programme :
  - avant de créer un nouveau programme, modifiez le répertoire courant en cliquant sur le bouton  à droite de la zone indiquant le répertoire de travail courant (utilisez votre répertoire personnel) : 
  - dans la barre de bouton de MATLAB cliquez sur le premier icône (New M-File)
  - dans la nouvelle fenêtre tapez les commandes suivantes :
 

```
a=3
x=1:10
y=a*x-2
plot(x,y)
grid on
```
  - sauvegardez ce programme «Script1». Pour enregistrer un programme existant et l'exécuter : touche F5.

Une fonction en MATLAB est un m-file particulier, on lui passe des arguments et il retourne des valeurs de sortie. Les fonctions sont très utiles dans la mesure où chaque utilisateur peut étendre les possibilités de MATLAB à son domaine d'application. Il est à noter que les variables utilisées à l'intérieur d'une fonction sont locales et cela afin de limiter la taille mémoire des programmes exécutés.

La forme générale de la déclaration d'une fonction est :

```
function nom_variable_retour = nom_fonction ( nom_paramètres )
```

Quelques remarques :

- le m-file et la fonction doivent avoir le même nom
- il peut y avoir plusieurs arguments (il suffit de les séparer par des virgules) ou aucun
- il peut y avoir plusieurs valeurs de retour (les mettre entre crochets et les séparer par des virgules) ou aucune.

**Exemple :** Implémentez (en mot à mot) la fonction ci-dessous.

```
function y = Fonction1(m,n)
%Génération aléatoire d'une matrice à coefficients entiers
%Fonction1(m,n) renvoie une matrice de dimensions m×n
%à coefficient compris entre 0 et 9
y = floor (10*rand(m,n)) ;
```


**Remarque:** La fonction « Fonction1 » peut être testée directement dans la fenêtre de commande avec :  
Fonction1(argument<sub>1</sub>, argument<sub>2</sub>),

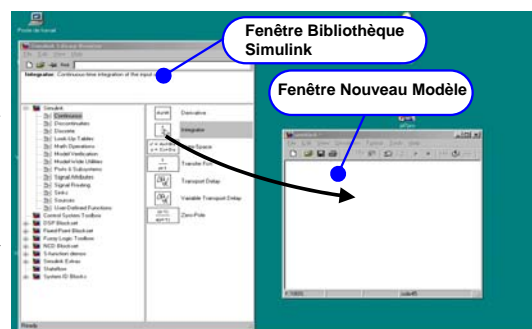


- 10 – Exécutez la commande  
`>> help Fonction1`  
 que constatez vous ?
- 11 – Créez une fonction « Fonction2 » telle que :  $y = x^2 - \sqrt{x} + 2x - 1$
- 12 – Écrivez un programme « Script2 » qui :  
 – dessine la fonction précédente entre 0 et 20,  
 – donne la valeur de x pour laquelle y vaut 0,  
 – donne la valeur de x pour laquelle y est minimum et qui calcule cette valeur y.  
Commentaire : les fonctions à utiliser sont « fplot », « fzero » et « fminbnd ».
- 13 – Écrivez un programme « Script3 » qui :  
 – crée et sauvegarde (dans un fichier Donnees.txt) un vecteur contenant 100 valeurs entières comprises aléatoirement entre 0 et 20,  
 – construit un histogramme de ces valeurs contenant 10 classes centrées autour des valeurs impaires,  
 – donne un titre à cette figure : « histogramme des notes »,  
 – affiche sur le graphique le nombre de valeurs,  
 – affiche sur le graphique la moyenne des valeurs (arrondie à  $10^{-2}$  près) ainsi qu'une ligne verticale correspondant à cette moyenne.  
Commentaire : cherchez l'aide sur les commandes « fprintf », « mean » et « num2str ».
- Les programmes un peu plus évolués nécessitent des éléments de structuration permettant les tests de valeurs, les boucles, les interruptions prioritaires, etc.
- 14 – Quels sont les opérateurs relationnels utilisables ?  
 – Expliquez la syntaxe des contrôles de flux « if », « for » et « while ».

### III.4 Simulink

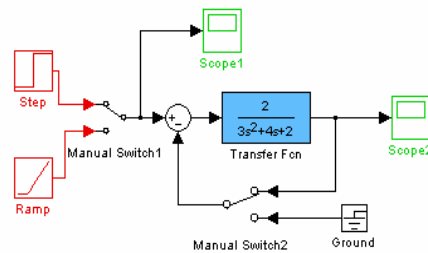
Simulink est l'extension graphique de MATLAB qui permet de modéliser, de simuler et d'analyser des systèmes dynamiques via l'utilisation de schémas-blocs.

- Pour ouvrir la bibliothèque de schémas-blocs, cliquez sur le bouton .
- Faire « Ctrl+N » pour ouvrir un nouveau modèle.
- Pour ajouter un nouveau schéma-bloc dans le modèle, il faut d'abord que celui-ci apparaisse dans la partie droite de la fenêtre de la bibliothèque. Par exemple, dans la partie gauche de la fenêtre, cliquez sur le symbole « + » de la ligne « Simulink » puis en-dessous cliquez sur « Continuous ». Dans la partie droite sept schémas-blocs apparaissent alors.
- Ajoutez maintenant le schéma-bloc « Integrator » en cliquant sur son symbole et en le glissant dans la fenêtre du nouveau modèle.



- 15 Pour apprendre à utiliser Simulink construisez le modèle donné dans l'aide Simulink :
- Dans la fenêtre Simulink cliquez sur le menu « Help » puis sur « Simulink Help ».
  - Cliquez ensuite sur « Getting Started » puis sur « Building a Model ».
  - Suivez les instructions pour construire le modèle donné que vous nommerez «Simulink1 ».

- 16 Maintenant que vous savez construire un modèle Simulink, construisez le modèle élémentaire ci-contre, que vous sauvegarderez sous le nom « Simulink2». Pour cela vous devrez utiliser les schémas-blocs suivants :



- « Transfer fcn » dans la liste « Continuous »,
- « Sum» dans la liste « Math Operations »,
- «Switch » de la liste « Signal Routing »,
- « Ground », « Step » et «Ramp » dans la liste « Source »,
- « Scope » dans la liste « Sinks ».

- 17
- Quel est le rôle potentiel du schéma Simulink2 implémenté ? Expliquer le rôle de chaque bloc graphique du schéma.
  - Réalisez un programme « Script4.m» qui réalise les mêmes fonctionnalités décrites dans le modèle Simulink2.
  - Trouvez les pôles de cette fonction de transfert, représentez le diagramme de Bode et de Nyquist du système étudié, commentez.
  - Expliquez à quoi correspond l’outil sisotool (single input single output tool) de la boite à outils «Control System Toolbox».

Commentaire : Utilisez les fonctions contenues dans le «Control System Toolbox» → help control.

## COMMANDES USUELLES

**Tableau 1 : les commandes indispensables**

help	donne toute la liste des aides
help <i>sujet</i>	affiche l’aide sur le <i>sujet</i>
<i>commande</i> ;	le ; permet d’interdire l’affichage du résultat de la commande à l’écran
clc	efface le contenu de la fenêtre de commande
quit, exit	extinction de MATLAB

**Tableau 2 : commandes sur fichiers et répertoires**

cd <i>nomrépertoire</i>	change de répertoire de travail (cd .. : remonter d’un répertoire)
pwd	affiche le nom du répertoire courant
dir, ls	affiche le contenu du répertoire courant
delete <i>nomfichier</i>	supprime le fichier <i>nomfichier</i>
load <i>nomfichier</i>	import les variables du fichier <i>nomfichier.mat</i> (format matlab)
load <i>nomfichier.dat</i>	importe la matrice <i>nomfichier</i> du fichier <i>nomfichier.dat</i> (format texte)
load <i>nomfichier X Y</i>	importe uniquement les variables X et Y du fichier <i>nomfichier.mat</i>
<i>nomfichier</i>	exécute le script contenu dans le fichier <i>nomfichier.m</i> (commandes, définitions, variables)
save <i>nomfichier</i>	sauvegarde toutes les variables dans le fichier <i>nomfichier.mat</i>
save <i>nomfichier X Y</i>	sauvegarde uniquement les variables X et Y dans le fichier <i>nomfichier.mat</i>

**Tableau 3 : gérer les variables**

length(A)	retourne le nombre de composants du vecteur A
size(A)	retourne le nombre de lignes et de colonnes de la variable A
who	liste les variables utilisées
whos	donne la liste des variables ainsi que leur dimension
clear	supprime toutes les variables de l'espace de travail
clear X Y	supprime seulement les variables X et Y
=, <, >, <=, >=	opérateurs de comparaison de variables (aide : <i>help eq, lt, gt, le, ge</i> )
x = y	affectation : x prend la valeur de y (aide : <i>help punct</i> )
a:b	intervalle [a, a+1 ... b] (aide : <i>help colon</i> )
x=y(:,k)	le vecteur x est égal toute la k <sup>ème</sup> colonne de la matrice y
x=y(a:b,k)	le vecteur x est égal à la k <sup>ème</sup> colonne de la matrice y, mais uniquement du composant numéro a au composant numéro b
max(X)	donne le plus grand élément du vecteur X
max(y(a :b,k))	donne le plus grand élément de la k <sup>ème</sup> colonne de la matrice y, mais uniquement de la ligne numéro a à la ligne numéro b
min(X)	si X est un vecteur, min(X) donne le plus petit élément de ce vecteur
min(y(a :b,k))	donne le plus petit élément de la k <sup>ème</sup> colonne de la matrice y, mais uniquement de la ligne numéro a à la ligne numéro b

**Tableau 4 : fonctions graphiques**

plot(X,Y,'cs')	dessine le graphe de Y en fonction de X suivant la couleur c et en utilisant le symbole s. c peut prendre entre autres les valeurs : y (jaune), m (magenta), c (cyan), r (rouge), g (vert), b (bleu), w (blanc), k (noir) et i (invisible). s peut être entre autres : . (point), - (tiret), -. (tiret-point), + (plus), * (étoile), o (cercle), x (croix).
title('texte')	ajoute le <i>texte</i> comme titre de la figure
xlabel('texte')	ajoute le <i>texte</i> comme légende de l'axe des abscisses
ylabel('texte')	ajoute le <i>texte</i> comme légende de l'axe des ordonnées
gtext('texte')	placera le <i>texte</i> à partir du point sélectionné à la souris
legend('texte1', 'texte2'...)	place une légende sur la figure courante. <i>texte1</i> , <i>texte2</i> , etc. sont les légendes de différentes courbes.
axis([X <sub>min</sub> X <sub>max</sub> Y <sub>min</sub> Y <sub>max</sub> ])	retrace la figure dans les limites imposées sur chaque axe
grid on ou grid off	ajoute ou enlève une grille sur la figure
figure	crée une nouvelle figure
figure(num)	la figure portant le numéro <i>num</i> devient la figure courante (celle où seront dessinées les courbes). La figure est créée si elle n'existait pas.
hold on ou hold off	autorise ou interdit le dessin des prochaines fonctions sur la figure sans effacer les précédentes (hold off est le mode par défaut).
close	ferme la figure courante
close(num)	ferme la figure portant le numéro <i>num</i>
close all	ferme toutes les figures

**Tableau 5 : programmation**

x = input('text')	permet de saisir la valeur de x au clavier
disp('xxx')	affiche à l'écran la chaîne de caractères xxx
pause	attend un appui sur une touche quelconque du clavier avant d'exécuter l'instruction suivante.
for i=min : max ... end	répétition en boucle. Pour i = min à i = max faire ...
break	termine immédiatement le traitement d'une boucle for ou while
if cond1 ... elseif cond2 ... else ... end	exécution conditionnelle. Si condition <i>cond1</i> faire ... sinon si <i>cond2</i> faire ...sinon faire...
% texte	commentaires