TP N°1 : Etude des systèmes dans L'espace d'état sous Matlab et Simulink

I. Introduction

Pendant longtemps, l'automaticien a utilisé exclusivement la notion de relation entrée-sortie. Or, pour les systèmes complexes, ce point de vue se révèle être une représentation incomplète des phénomènes.

La connaissance des variables internes d'un système peut présenter un intérêt certain en ce sens qu'elles permettent de déterminer l'état d'un système à un instant donné. Ces variables d'état sont donc l'ensemble des variables nécessaires à la caractérisation de l'état d'un système. Elles sont regroupées dans un vecteur appelé vecteur d'état.

II. Rappels

Les équations d'état d'un système sont constituées de l'ensemble des équations matricielles suivantes :

$$\begin{cases} \dot{\underline{x}}(t) = A\underline{x}(t) + B\underline{u}(t) & \text{équation d'état} \\ \underline{y}(t) = C\underline{x}(t) + D\underline{u}(t) & \text{équation d'observation} \end{cases}$$

Avec :

x : vecteur des variables d'état

u : vecteur des commandes

y : vecteur des sorties

A : matrice d'évolution de dimension $(n \times n)$(n :ordre du système)

B : matrice de commande de dimension $(n \times l)$(l :nombre d'entrées du système)

C : matrice d'observation de dimension $(m \times n)$ (m :nombre de sorties)

D : matrice de transmission directe de dimension $(m \times l)$

Ces équations sont établies pour les systèmes multivariables, dans le cas particulier des systèmes monovariable qui nous intéresse, on peut réécrire ces équations de la façon suivante :

$$\begin{cases} \dot{\underline{x}}(t) = A\underline{x}(t) + Bu(t) \\ y(t) = C\underline{x}(t) + Du(t) \end{cases}$$

Dans la plupart des systèmes rencontrés en automatique, il n'existe pas de lien direct entre l'entrée et la sortie du système. Par conséquent, dans la représentation d'état de tels systèmes, la matrice D de transmission directe est nulle.

Le choix des variables d'état conduit à des formes particulières pour les matrices, ces structures spécifiques sont appelées forme canonique, qu'on peut classer en trois formes [8] :

Forme canonique diagonale

La matrice d'état est diagonale et les éléments de la diagonale sont les valeurs propres.

Forme canonique commandable

Une ligne de la matrice d'état correspond aux coefficients du polynôme caractéristique de la fonction de transfert.

Forme canonique observable

Une colonne de la matrice d'état correspond aux coefficients du polynôme caractéristique de la fonction de transfert.

III. Présentation du logiciel Matlab

(MATrix LABoratory) est l'outil de référence pour la simulation numérique, notamment en ce qui concerne l'Automatique.

Matlab est une application Windows. En cliquant deux fois sur l'icône **Matlab**, s'ouvre alors la fenêtre principale. Cette fenêtre est divisée en plusieurs parties, comme le montre la Figure 1 ci-dessous.

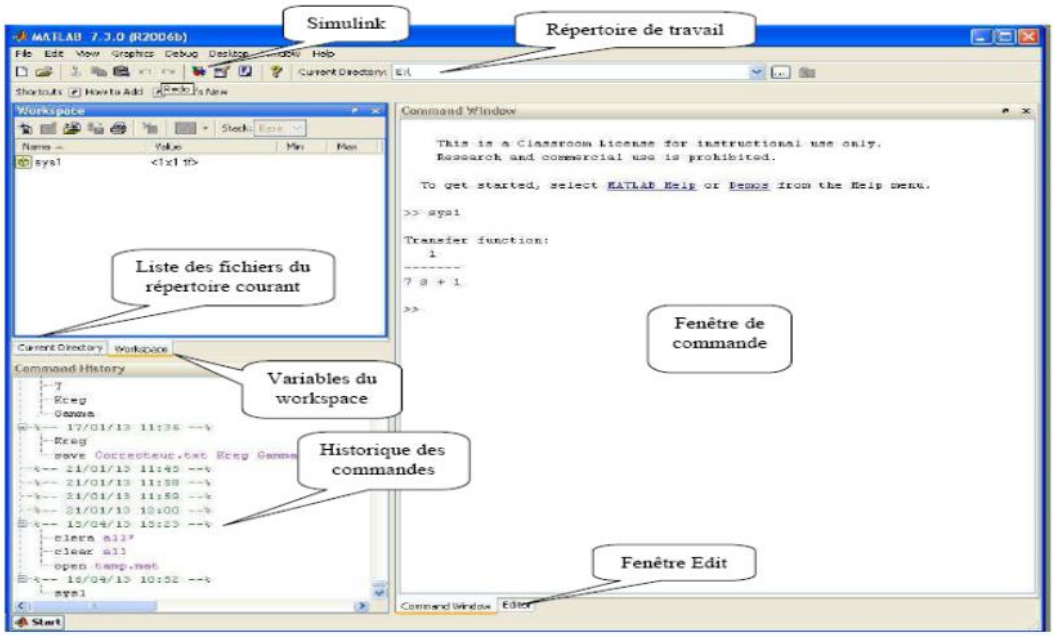


Figure 1 : IHM (Interface Homme Machine) de **Matlab**

Programmation sous Matlab :

- Lancer Matlab sous windows, une fenêtre (command window) est alors affichée.
- Cliquer sur File et choisir "New" ensuite "Mfile", une nouvelle fenêtre est ouverte dans laquelle on peut écrire notre programme.
- Donner un nom au programme à enregistrer.
- Pour l'exécution du programme, taper son nom dans la fenêtre (Command window) ou cliquer sur l'icône **run**  dans la barre d'outils de Matlab.

Une boîte à outils de Matlab correspond à un ensemble de fonctions disponibles dans la fenêtre de travail que l'on peut appeler à l'invite ≫

De même, pour chaque commande particulière, vous avez accès à deux niveaux d'aide :

1. Un 1er niveau d'aide par : ≫ help 'nom de la commande'
2. Un 2eme niveau d'aide, avec navigation hypertexte par : ≫helpwin 'nom de la commande' .

Exemple: ≫help **ss** ou ≫helpwin **ss**

Matlab possède une boîte à outils réservée à l'étude des systèmes de commande « control system Toolbox », ainsi qu'un environnement graphique de simulation des systèmes dynamiques **Simulink** sous forme de schémas en blocs.

Pour lancer Simulink, on clique sur l'icône  dans la barre d'outils de Matlab, ou bien par la commande ≫ simulink à l'invite, dans la fenêtre de travail de Matlab.

Ceci nous donnera accès à différents menus ou boîtes de Simulink ; ainsi on ouvrira une nouvelle fenêtre Simulink à partir de laquelle on pourra créer un nouveau fichier modèle et placer les éléments constitutifs du système à étudier.

La librairie de Simulink comprend de différentes sections permettant d'aborder de nombreux aspects de la commande des systèmes.

Dans notre polycopié on portera notre attention sur les éléments suivants comme le montre la Figure I.2 :

- **Step** : se trouve dans **Sources**, générateur de l'échelon.
- **Transfert fcn** : se trouve dans **Continuous**, permet de définir une fonction de transfert. Changez les paramètres pour qu'elle corresponde à la fonction de transfert demandée.
- **State-space :** se trouve dans **Continuous**, permet de définir une représentation d'état. changer les matrices A,B ,C,D pour qu'elle correspondre à la représentation d'état

demandée.

• **Gain :** se trouve dans **Math Operations**, permet de multiplier l'entrée par un gain k(scalaire ou vecteur)

• **Mux** : se trouve dans **Signal routing**, permet de multiplexer plusieurs signaux dans un fil.

• **Sum** : se trouve dans **Math Operations**, permet de réaliser le comparateur. Il faut choisir les signes + et -.

• **Scope** : se trouve dans **Sinks**, c'est un scope rudimentaire pour avoir rapidement un tracé des courbes.

• **ToWorkspace**: il se trouve dans **Sinks**, permet de récupérer le résultat de la simulation dans une variable exploitable sur Matlab (ligne de commande).

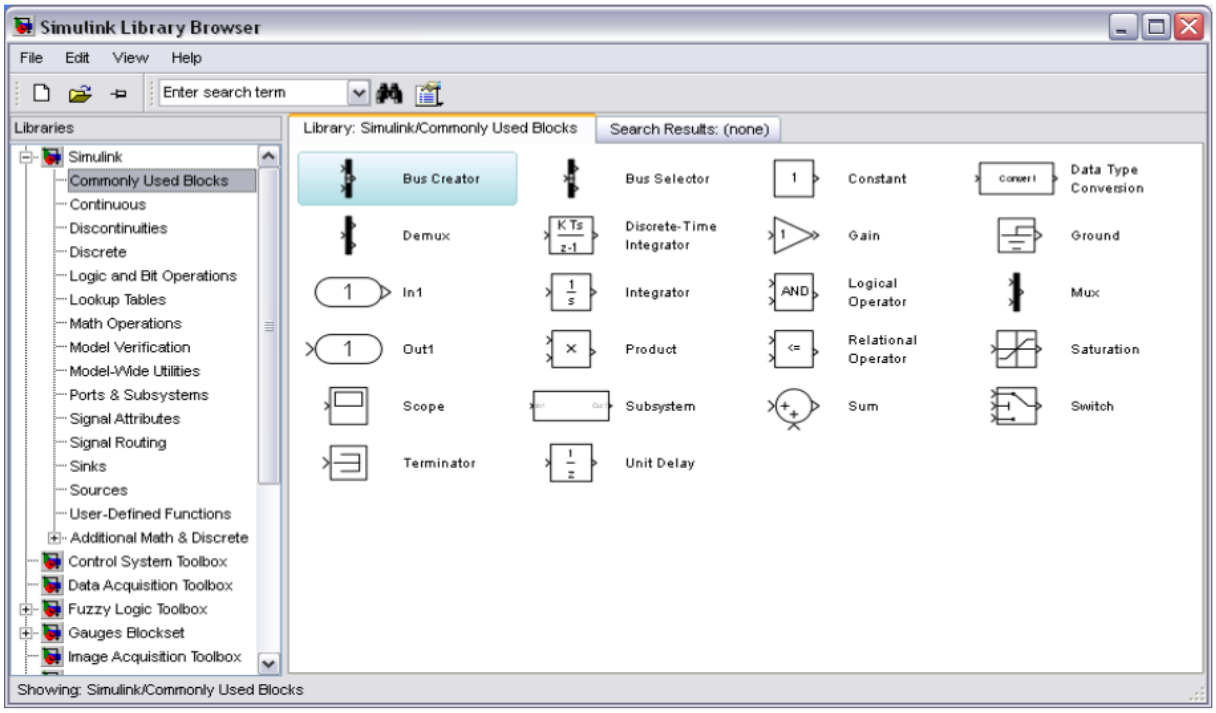


Figure 2 : interface Simulink

IV. Modélisation du système :

Le système est un moteur de position, l'actionneur principal dans le système de control est un moteur CC voir figure 3, il fournit directement un mouvement de rotation et, couplé à travers des roues pour fournir un mouvement de translation.

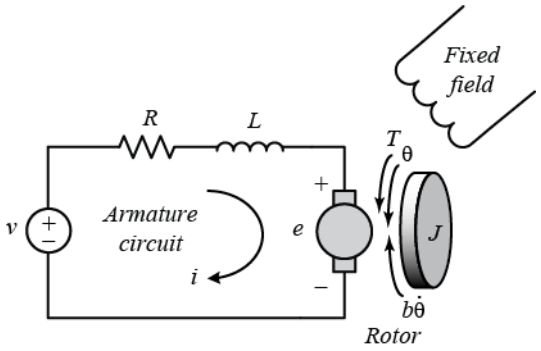


Figure 1 – Schéma fonctionnel d'un Moteur à Courant Continu.

Caractéristique de système :

(J) Moment d'inertie de rotor 3.2284 E-6 kg.m^2

(b) Constant de frottement visqueuse de moteur 3.5077 E-6 N.m.s

(Kb) Constant de force électromotrice 0.0274 V/rad/sec

(Kt) constant de couple moteur 0.0274 N.m/Amp

(R) resistance électrique 4 Ohm

(L) inductance électrique 2.75E-6H

Dans toute la suite de ce TP, le système considéré n'a qu'une entrée et qu'une sortie, tel que nous supposons que l'entrée du système est la source de tension (V) appliquée à l'induit du moteur, alors que la sortie est la position de l'arbre (thêta).

Nous supposons en outre un modèle de frottement visqueux, c'est-à-dire que le couple de frottement est proportionnel à la vitesse angulaire de l'arbre.

1. **Système d'équations**

En général, **le couple généré** par un moteur à courant continu est proportionnel au courant d'induit et à la force du champ magnétique. Dans cet exemple, nous supposerons que le champ magnétique est constant et, par conséquent, que le couple moteur est proportionnel uniquement au courant d'induit i par un facteur constant Kt comme indiqué dans l'équation ci-dessous. C'est ce qu'on appelle un moteur commandé par induit.

$$T = K_t i \quad \textbf{(1)}$$

La force contre-électromotrice, e, est proportionnelle à la vitesse angulaire de l'arbre par un facteur constant Kb

$$e = K_b \dot{\theta} \quad \textbf{(2)}$$

En unités SI, les constantes du couple moteur et de la force contre-électromotrice sont égales, c'est-à-dire Kt = Kb ; par conséquent, nous utiliserons K pour représenter à la fois la constante de couple du moteur et la constante de force contre-électromotrice.

À partir de la figure ci-dessus, nous pouvons dériver les équations directrices suivantes basées sur la 2e loi de Newton et la loi de tension de Kirchhoff.

$$J\ddot{\theta} + b\dot{\theta} = Ki \tag{3}$$

$$L\frac{di}{dt} + Ri = V - K\dot{\theta} \tag{4}$$

2. Fonction de transfert :

En appliquant la transformée de Laplace, les équations de modélisation ci-dessus peuvent être exprimées en fonction de la variable de Laplace s.

$$s(Js + b)\Theta(s) = KI(s) \tag{5}$$

$$(Ls + R)I(s) = V(s) - Ks\Theta(s) \tag{6}$$

Nous arrivons à la fonction de transfert en boucle ouverte suivante en éliminant I(s) entre les deux équations ci-dessus, où la vitesse de rotation est considérée comme la sortie et la tension d'induit est considérée comme l'entrée

$$P(s) = \frac{\dot{\Theta}(s)}{V(s)} = \frac{K}{(Js + b)(Ls + R) + K^2} \qquad [\frac{rad/sec}{V}] \tag{7}$$

Cependant, au cours de cet exemple, nous examinerons la position en tant que sortie. Nous pouvons obtenir la position en intégrant la vitesse, il suffit donc de diviser la fonction de transfert ci-dessus par s.

$$\frac{\Theta(s)}{V(s)} = \frac{K}{s((Js + b)(Ls + R) + K^2)} \qquad [\frac{rad}{V}] \tag{8}$$

3. Equations d'état :

Les équations différentielles ci-dessus peuvent également être exprimées sous forme variable d'état en choisissant la position du moteur, la vitesse du moteur et le courant d'induit comme variables d'état ; ou la tension d'induit est considéré comme entrée et la position de rotation est choisie comme sortie.

$$\frac{d}{dt}\begin{bmatrix}\theta\\ \dot{\theta}\\ i\end{bmatrix}=\begin{bmatrix}0 & 1 & 0\\ 0 & -\frac{b}{J} & \frac{K}{J}\\ 0 & -\frac{K}{L} & -\frac{R}{L}\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\\ i\end{bmatrix}+\begin{bmatrix}0\\ 0\\ \frac{1}{L}\end{bmatrix}V \tag{9}$$

$$y=\begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\\ i\end{bmatrix} \tag{10}$$

V. Exigences de conception (cahier de charge) :

Nous voudrons pouvoir positionner le moteur très précisément, ainsi l'erreur en régime permanent de la position du moteur devrait être nulle lorsqu'on lui donne une position commandée. Nous voudrons également que l'erreur en régime permanent due à une perturbation constante soit également nulle. L'autre exigence de performance est que le moteur atteigne sa position finale très rapidement sans dépassement excessif.

Dans ce cas, nous voulons que le système ait un temps de stabilisation de 40 ms et un dépassement inférieur à 16%.
Si nous simulons l'entrée de référence par une entrée de pas d'unité, alors la sortie de position du moteur devrait avoir

- Temps de stabilisation inférieur à 40 millisecondes
- Dépassement inférieur à 16%
- Pas d'erreur en régime permanent, même en présence d'une entrée de perturbation échelonnée

VI. Etude et Analyse de la représentation d'état des systèmes

❖ Préparation : (à remettre au début du TP)

1) Donner la représentation d'état de système dessus,
2) Parmi les éléments A, B, C, D, x, u, y, quels sont ceux qui varient en fonction de la représentation d'état choisie? Pourquoi les autres éléments restent-ils identiques quel que soit la représentation d'état?
3) La représentation d'état d'un système est-elle unique ?

4) Ecrire les équations générales permettant, de passer de la représentation d'état d'un système à la représentation externe (Fonction de transfert), en utilisant la transformation de Laplace,
5) En prenant la transformée de Laplace de l'équation différentielle du système, donner sa fonction de transfert qui relie l'entrée avec la sortie. Quel est l'ordre du système ?

❖ **Répondre à toutes les questions suivantes pour chaque système**

1) Créer un fichier de type M-file et en utilisant la fonction Matlab **ss**, construire la représentation d'état continue du système,
2) Indiquer la dimension de chacune des matrices *A, B, C,* et *D,* en utilisant la fonction Matlab **size**,
3) Calculer la fonction de transfert $H(p) = C.\ [pI\ -\ A]\text{-}1\ \cdot\ B$ en utilisant les fonctions Matlab **syms, inv,eye**,
4) En utilisant la fonction Matlab **ss2tf**, passer de la représentation d'état (représentation interne) du système à sa fonction de transfert (représentation externe),
5) En utilisant la fonction Matlab **tf2ss**, passer de la fonction de transfert (représentation externe) du système à sa représentation d'état (représentation interne),
6) Trouver la représentation d'état sous la forme canonique diagonale en utilisant la fonction Matlab **canon** avec l'option 'modal',
7) Trouver la représentation d'état sous la forme canonique observable en utilisant la fonction Matlab **canon** avec l'option 'companion',
8) Comment vérifier si deux modèles d'état sont équivalents du point de vue du comportement entrée-sortie ?
9) Analyser la position des pôles et des zéros du système, et déterminer s'il est stable en boucle ouverte ou pas, tout en justifiant. Pour cela utiliser la fonction Matlab **pzmap**
10) Calculer les valeurs propres de la matrice d'état en utilisant la fonction Matlab **eig**, Faire le point en répondant à cette question : Que peut-on dire sur les valeurs propres de A et les pôles du système ?
11) Calculer le gain statique du système en utilisant la fonction Matlab **dcgain**,
12) Analyser les performances temporelles du système (fonction de transfert et représentation d'état) en utilisant la fonction Matlab **step**, (En cliquant sur le tracé avec le bouton droit, il est possible d'obtenir des renseignements sur le temps de réponse, le temps de montée, le gain statique ainsi que le dépassement),

13) Analyser les performances fréquentielles du système(fonction de transfert et représentation d'état) en utilisant la fonction Matlab **bode(** En cliquant sur le tracé avec le bouton droit, il est possible d'obtenir des renseignements sur le gain statique et les marges de gain et de phase)**,**
14) Lancer Simulink, construire le modèle du système. Pour cela, on recherchera dans les menus Simulink les différents blocs nécessaires (fonction de transfert ou représentation d'état continue, gain, sommateur, oscilloscope pour visualiser…etc), que l'on fera glisser dans le modèle. Une fois les éléments essentiels présents, il suffit de les lier entre eux avec la souris c'est-à-dire tracer des connections entre la sortie du premier bloc et l'entrée du deuxième bloc. Ne pas oublier de sauvegarder le schéma dans votre répertoire de travail avant de l'exécuter,
15) Simuler les réponses indicielles et impulsionnelles du système en boucle ouverte,
16) Faire une étude comparative entre les deux résultats obtenus par le programme (Mfile) et par le modèle Simulink,
17) Interpréter les résultats,
18) Tracez l'allure de la réponse indicielle unitaire en boucle fermée du système en utilisant les fonctions Matlab **step** et **feedback**,
19) Simuler dans Simulink, la réponse indicielle du système en boucle fermée.

Solution :

1) Représentation d'état continue du système,

```
A = [0 1 0
     0 -b/J K/J
     0 -K/L -R/L];
B = [0 ; 0 ; 1/L];
C = [1 0 0];
D = [0];
```

motor_ss = ss(A,B,C,D)

2) Dimensions matrice d'état

   size(A) ;size(B) ;size(C) ;size(D)

3) Passage fonction transfert vers espace d'etat

ft= C*(inv(s*eye(2,2)-A))*B, ou [Num,Den]=ss2tf(A,B,C,D) ;g1=tf(motor_ss);

4) Passage espace d'etat vers fonction transfert

   [A,B,C,D]=tf2ss(Num,Den);

5) Forme canonique modale

csys = canon(motor_ss,'modal')

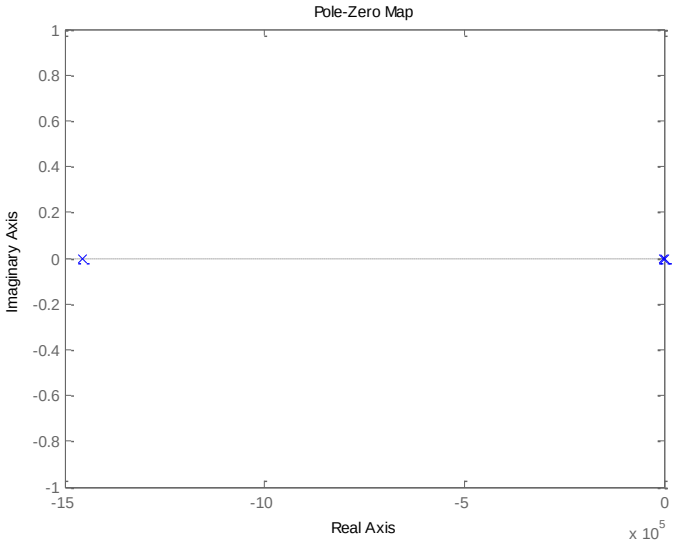
6) Forme observable

   csys = canon(motor_ss,'companion')

7) Vérification :

   On met les trois representations de meme entres on remarque que obtient les memes sortie par simulink

8) D'après la figure ci-dessus, un système linéaire invariant dans le temps en boucle ouverte est stable si : En temps continu, tous les pôles du plan s complexe doivent se trouver dans le demi-plan gauche (région bleue) pour assurer la stabilité. Le système est marginalement stable si des pôles distincts se trouvent sur l'axe imaginaire, c'est-à-dire que les parties réelles des pôles sont nulles


Pole-Zero Map
Imaginary Axis
Real Axis
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-15
-10
-5
0
x 10^5