Tutoriel Matlab Simulink

1 Introduction

Le logiciel Matlab et la toolbox Simulink, sont des outis de simulation multi-domaine de systèmes dynamiques. Ils permettent de prototyper rapidement un contrôleur et de le tester afin d'en évaluer les performances. Ce logiciel sera utilisé dans le cadre des TPs d'automatique. Cependant il existe un équivalent gratuit : Scilab et la toolbox Xcos.
Ce logiciel va nous permettre de simuler des systèmes continus ou discrets. Pour cela deux méthodes sont possibles.

— Une méthode en ligne de commande via la console de Matlab,
— Une méthode graphique via la Toolbox Simulink.

2 Accès à Matlab

Matlab/Simulink est accessible dans la salle D206 mais également sous n'importe quel poste informatique de Polytech équipé de Fedora. La procédure suivante permet de se connecter au site du SCI et d'utiliser Matlab. (Pour plus d'information reportez vous à la documentation *Aide Polytech* à partir de la page 53, chapitre, se connecter à une machine distante.)

— Menu Applications
— Internet
— VMware View
— Se connecter en utilisant le login et mot de passe
— Version Matlab R2012b.

3 Systèmes continus

3.1 Simulation à partir de lignes de commande

3.1.1 Saisie d'une fonction de transfert

Dans l'environnement Matlab, une fonction de transfert continue peut être définie au moyen de l'instruction `tf` :

- l'instruction `» F=tf([3],[2 1])` crée une variable F qui représente la fonction de transfert ci-dessous :
$$F(p) = \frac{3}{2\,p+1}$$
  — le premier paramètre [3] contient les coefficients du polynôme numérateur de $F(p)$, rangés dans l'ordre des puissances décroissantes de p.
  → ici, un seul coefficient, donc le numérateur de $F(p)$ est une constante, de valeur 3.
  — le second paramètre [2 1] contient les coefficients du polynôme dénominateur de $F(p)$, rangés dans l'ordre des puissances décroissantes de p.
  → ici, deux coefficients, donc le dénominateur de $F(p)$ est un polynôme de degré 1, dont les coefficients sont 2 et 1, c'est-à-dire il s'agit du polynôme $2\,p+1$
- si le système à simuler comporte un retard pur, il peut être spécifié en surchargeant l'instruction `tf` comme suit : `» F=tf([3],[2 1],'Outputdelay',4)`. La fonction de transfert créée est alors :
$$F(p) = \frac{3}{2\,p+1}\,e^{-4\,p}$$

3.1.2 Opérations sur les fonctions de transfert

Si on souhaite calculer la fonction de transfert équivalente, il suffit d'écrire l'équation en utilisant les opérateurs classiques (*,/,+,-). Par exemple pour connaitre la réponse en boucle fermée il suffit d'utiliser l'instruction :

```
>> Fbf = F / (1+F)
```

3.1.3 Visualisation d'une réponse indicielle

Si `F` est une variable qui décrit la fonction de transfert d'un système donné, alors la réponse de ce système à une entrée échelon (d'amplitude 1,appliquée à l'instant $t = 0$ s, depuis des conditions initiales nulles) peut-être calculée au moyen de l'instruction `step`, puis tracée au moyen de l'instruction `plot` :

```
>> [y,t]=step(F);
>> figure(1), plot(t,y), grid on
```

`t` est un vecteur colonne où sont rangés tous les instants où la sortie du système est évaluée et `y` est un vecteur colonne où l'on retrouve la valeur de la sortie à chacun de ces instants *(**t** et **y** ont donc forcément la même dimension).*

Si on souhaite superposer la réponse indicielle de 2 systèmes décrits par les fonctions de transfert `F` et `G`, il suffit d'exécuter la séquence d'instructions :

```
>> [yF,tF]=step(F); [yG,tG]=step(G);
>> figure(1), plot(tF,yF,'b',tG,yG,'r'), grid on
```

3.1.4 Visualisation d'une réponse à une entrée quelconque

Si on souhaite maintenant visualiser la réponse d'un système modélisé par une fonction de transfert `F` à une entrée quelconque, alors il convient :

- de définir le signal d'entrée. Pour cela :
  - on crée un vecteur `t` où l'on range tous les instants où le signal d'entrée va être défini,
  - puis on crée un vecteur `u`, de même dimension que `t`, où l'on range les valeurs de l'entrée à chacun de ces instants
- puis on évalue la sortie du système à ce signal d'entrée au moyen de l'instruction `lsim` : `y=lsim(F,u,t);`
- et il ne reste plus qu'à exécuter le tracé graphique.

Par exemple, si on souhaite visualiser la réponse à une rampe de pente 2 appliquée depuis $t = 0$ s pendant 10 s, il convient d'exécuter les instructions :

```
>> t=0:0.1:10; u=2*t;
>> y=lsim(F,u,t);
>> figure(1), plot(t,u,'b',t,y,'r'), grid on
```

Si on souhaite maintenant visualiser la réponse de ce même système à une entrée sinusoïdale de fréquence 0.5 Hz et d'amplitude 2, il suffit de modifier le signal u

```
>> t=0:0.1:10; u=2*sin(2*pi*0.5*t);
```

3.1.5 Tracé d'un diagramme fréquentiel

La visualisation des diagrammes de Bode, de Nyquist ou de Black-Nichols d'un système modélisé par une fonction de transfert `F` s'obtient immédiatement au moyen de l'instruction :

```
>> figure(1), bode(F), grid on
```

Enfin, si à des fins de comparaison, on souhaite superposer les diagrammes de Bode de 2 systèmes modélisés par les fonctions de transfert F et G, il suffit de surcharger l'instruction `bode` :

```
>> figure(1), bode(F,G), grid on
```

On utilisera l'instruction `nyquist` ou `nichols` pour les diagrammes de Nyquist ou Black-Nichols.

3.2 Simulation à partir de schémas-blocs

3.2.1 Simulation sans interface physique réelle

Une alternative pour simuler la réponse d'un système linéaire consiste à dessiner, au moyen de Simulink, le schéma-bloc de la simulation. Pour cela, il convient :

- d'ouvrir Simulink, soit en tapant `simulink` dans la fenetre de commande ou de cliquer sur l'icône correspondant.
- d'ouvrir une palette graphique *(depuis la barre de menu : File > New > Model)*
- puis d'ouvrir depuis cette fenêtre la librairie de composants *(View > Library Browser)*
- on amène ensuite, par glisser-déplacer, les différents composants sur la palette graphique vierge :
  - le bloc *Transfer Fcn* se trouve dans le répertoire ***Continuous*** de la librairie. Si le système comporte un retard pur, il faut faire glisser également un bloc *Transport Delay*.
  - le signal d'entrée se sélectionne dans le répertoire ***Source***. On peut sélectionner un échelon *Step*, une rampe *Ramp* ou un signal sinusoïdal *Sine Wave*.
  - pour visualiser le signal de sortie, on pioche dans le répertoire ***Sinks*** :
    - soit un oscilloscope virtuel *Scope*, pour une visualisation immédiate,
    - soit une sortie vers l'espace de travail Matlab *To Workspace*, pour donner un nom au signal de sortie et le visualiser a posteriori depuis Matlab via une instruction `plot` (utile lorsque l'on veut faire des superpositions).
- on relie ensuite les différents blocs (i.e. on tire un fil (au moyen de la souris) depuis le bloc d'entrée et l'entrée du bloc fonction de transfert, puis un fil entre la sortie de ce bloc et le bloc de visualisation)
- il faut ensuite configurer ces blocs. Pour cela, on ouvre la boîte de dialogue en double-cliquant sur le bloc. On peut ainsi préciser les caractéristiques du signal d'entrée, spécifier la fonction de transfert et nommer la variable qui récupérera le signal de sortie (si on utilise un bloc *To Workspace*). Pour ce dernier bloc, à l'item *Save format*, on sélectionnera *Array*.
- depuis la barre de menu de la palette graphique, on sélectionne ensuite *Simulation > Configuration parameters*

  - sur la première page de la fenêtre qui s'ouvre, on indique la durée de simulation dans *Stop time*,
  - puis en naviguant sur l'item *Data Import/Export*, on peut renommer la variable qui regroupe les instants où la sortie est évaluée (par défaut, elle se nomme `tout`). Cette variable sera ensuite à utiliser au sein de l'instruction `plot` pour exécuter les tracés graphiques.

- à ce stade, la simulation à réaliser est complètement définie. On peut l'exécuter en appuyant sur le bouton *Play* dans la barre de menu de la palette graphique *(symbolisé par un triangle noir pointant vers la droite)* et exploiter ses résultats.

- si les courbes visualisées à l'issue de la simulation se présentent comme des lignes brisées, cela traduit qu'un pas d'intégration trop long a été utilisé par Simulink pour réaliser la simulation.
  → pour contraindre Simulink à utiliser un pas plus court, il faut de nouveau, depuis la barre de menu de la palette graphique, sélectionner *Simulation > Configuration parameters*, puis remplacer la mention *auto* dans le champ *Max step size* par une valeur numérique appropriée à la simulation réalisée.

Si la version de Matlab dispose de la boîte à outils *Control System Toolbox*, la librairie de Simulink offre dans ce répertoire un bloc *LTI System* qui permet de spécifier le système à simuler directement par la variable décrivant sa fonction de transfert : si l'instruction `tf` a été utilisée pour associer une fonction de transfert à la variable `F`, alors on peut directement spécifier le système à simuler en reportant `F` dans la boîte de dialogue du bloc *LTI System*.

3.2.2 Simulation avec interface physique réelle

Lors de certaines séances de TPs, vous serez amenés à utiliser Simulink pour contrôler des processus réels. Il faudra alors utiliser les fichiers simulink (*.mdl) disponibles sur l'ENT. Il sera nécessaire de construire le fichier (appuyer sur le bouton build) puis de se connecter au matériel (bouton connect to target) pour enfin pouvoir lancer la manipulation.

4 Systèmes discrets

4.1 Simulation à partir de lignes de commande

Il est possible de définir une transmittance par la commande

- `» Fdiscret=tf([0 1],[1 2 1],0.1)` Où :
  - le premier paramètre [0 1] contient les coefficients du polynôme numérateur, rangés dans l'ordre des puissances décroissantes de z.
  - le deuxième paramètre [1 2 1] contient les coefficients du polynôme dénominateur, rangés dans l'ordre des puissances décroissantes de z.
  - le troisième paramètre [0.1] contient la période d'échantillonage.

Il est également possible de "convertir" une fonction de transfert continue en transmittance (discrète) par la commande :

- `» Fdiscret=c2d(F,0.1)` Où :
  - le premier paramètre [F] est la fonction de transfert.
  - le second paramètre [0.1] contient la période d'échantillonage.

4.2 Simulation à partir de schémas-blocs

Il est également possible d'obtenir des schémas-blocs de composants discrets dans le répertoire ***discrete***. Il faudra néanmoins veiller à configurer correctement les périodes d'échantillonage.

5 Tableau des principaux blocs Simulink

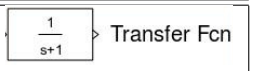
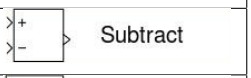
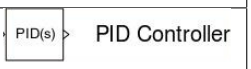
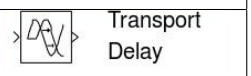
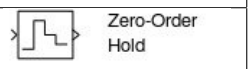
Description	Emplacement	Nom	Icone
Fonction de transfert continue	Continuous	Transfer Fcn	
Fonction de transfert discrète	Discrete	Discrete Transfer Fcn	
Échelon	Sources	Step	
Multiplexeur	Signal Routing	Mux	
Scope	Sinks	Scope	
Soustracteur	Math Operators	Subtract	
Controlleur (PI/PID)	Continuous	PID Controller	
Controlleur (PD)	Continuous	Transfer Fcn	
Correcteur (P)	Math Operators	Gain	
Additionneur	Math Operators	Add	
Intégrateur	Continuous	Integrator	
Retard pur	Continuous	Transport Delay	
Signal sinusoïdal	Sources	Sine Wave	
Bloqueur	Discrete	Zero holder	
Polynome	Math Operations	Polynomial	
Saturation	Discontinuities	Saturation	

Table 1 – Tableau des principales fonctions de Matlab Simulink