REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
الجمهورية الجزائرية الديمقراطية الشعبية

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

ECOLE SUPERIEURE EN SCIENCES APPLIQUEES
--T L E M C E N--

وزارة التعليم العالي والبحث العلمي

المدرسة العليا في العلوم التطبيقية
-تلمسان-

Département second cycle

Polycopié de Travaux Pratiques

Systèmes Asservis Linéaires Continus

Présenté par : Dr. ARICHI FAYSSAL

Dr. MOKHTARI MOHAMED REDA

Liste des travaux pratiques :

TP 1 : Introduction à Matlab

TP 2 : Analyse temporelle des systèmes du premier et deuxième ordre.

TP3: Analyse fréquentielle des systèmes du premier et deuxième ordre.

TP 4 : Correction des systèmes dynamiques.

TP 5 : Commande des systèmes par retour d'état dans l'espace d'état.

TP 1 : Introduction à Matlab

1- Objectif :

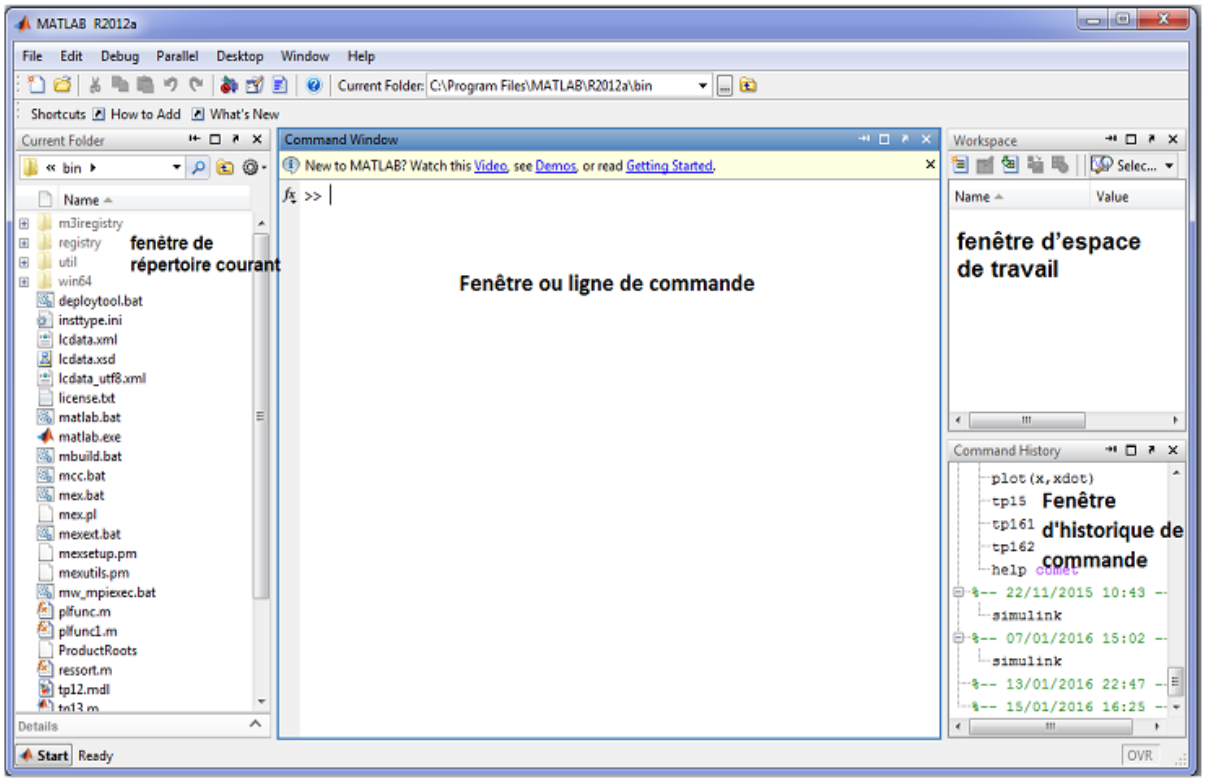
Il s'agit de familiariser les étudiants avec l'usage de Matlab et découvrir les fonctions et outils de Matlab relatifs à l'étude des systèmes asservis.

2- Introduction :

Matlab est un logiciel de calcul mathématique basé sur la manipulation de variables matricielles. Il propose la résolution numérique d'un grand nombre de problèmes mathématiques classiques, du plus grand intérêt pour les ingénieurs et les chercheurs. Matlab est d'ailleurs un raccourci pour « Matrix Laboratory ». Il possède une boite à outils dédiée à l'étude des systèmes de commande « Control system Toolbox », ainsi qu'un environnement graphique de simulation numérique des systèmes dynamiques : Simulink.

3- Démarrage de Matlab :

On démarre MATLAB en double cliquant sur l'icône MATLAB  présente sur le bureau de votre ordinateur. Une fois cette commande est exécutée il y'aura l'affichage de la fenêtre active suivante :



Cette fenêtre est divisée en plusieurs parties :

- **COMMAND WINDOW**: invite de commande permettant de taper des instructions, d'appeler des scripts, et d'exécuter des fonctions Matlab. Notez que le prompt Matlab (>>) indique que Matlab attend des instructions. Une fois l'instruction tapée correctement, on peut l'exécuter en appuyant sur la touche entrée.
- **COMMAND HISTORY**: Matlab garde en mémoire les dernières commandes effectuées. Elles sont visibles dans l'ongle Command History. On peut également y accéder directement dans la Command Window au moyen des touches [↑] et [↓]. Ceci est particulièrement utile pour répéter la dernière commande.
- **WORKSPACE**: permet d'afficher les variables utilisées et également de parcourir graphiquement le contenu des variables.
- **CURRENT DIRECTORY** : un navigateur de fichier intégré à Matlab pour visualiser le répertoire de travail courant et y effectuer les opérations classiques tel que renommer ou supprimer un fichier.

4- Les variables dans Matlab :

Le nom d'une variable doit toujours commencer par une lettre. Matlab fait la différence entre les MAJUSCULES et les minuscules. Sauf cas d'extrême pénurie de lettres, évitez de panacher majuscules et minuscules. Précisons que les commandes Matlab sont toujours tapées en minuscules.

Les variables sont créées lors de la première affectation, sous Matlab, l'affectation se fait à l'aide de l'opérateur « = » :

```
>> A=2        % A ←2  on affecte la valeur 2 à la variable A.
A =

    2

>> B=10     % B ←10 on affecte la valeur 10 à la variable B.
B =

   10
```

On peut alors utiliser une variable dans un calcul ou pour affecter son contenu à une autre variable.

```
>> A=B        % A←B on affecte la valeur contenue dans B à la variable A,
                        A vaut à présent 10.
A =
   10

>> A=A+1     % A←A+1 on affecte la valeur contenue dans A incrémentée de 1
                                à la variable A
A =

   11
```

Matlab dispose de plusieurs variables dont les noms sont, en principe, réservés et les valeurs préaffectées :

- **ans** : à défaut d'une variable crée par l'utilisateur pour recueillir un résultat, Matlab l'affecte à la variable **ans** (answer). Bien sûr le contenu de **ans** change à chaque affectation.
- **pi** : contient le nombre π, **eps** (comme epsilon) contient le nombre $\mathbf{2^{-52}}$, **realmin** et **realmax** sont respectivement le réel le plus petit et le plus grand disponibles dans Matlab.

Si, volontairement ou par erreur, on affecte une valeur à l'une de ces variables **(pi=2.51**), c'est cette nouvelle valeur qui sera véhiculée par **pi**. Pour retrouver la bonne valeur de **pi**, il faut supprimer **pi** du workspace en tapant **clear pi**. On utilise la commande **clear all** pour effacer toutes les variables de l'espace de travail.

5- L'aide dans Matlab :

La commande help est l'une des plus utiles pour commencer l'apprentissage de Matlab. Il y a deux types d'aide sur Matlab.

La première correspond à une aide générale très intéressante pour obtenir un renseignement sur une simple fonction. Afin d'obtenir l'information, il suffit de taper **help topic** où **topic** est le nom d'une fonction Matlab.

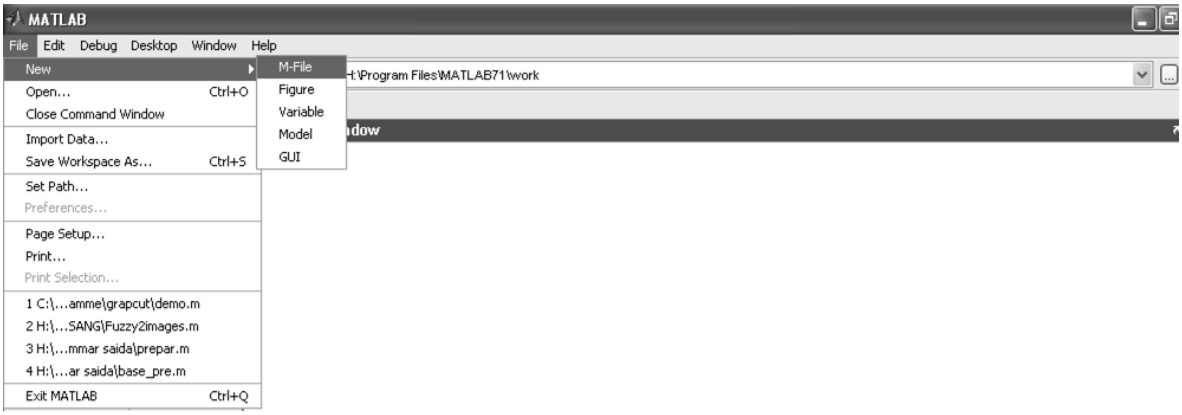
La deuxième est une aide en ligne qui est très précieuse si l'on veut éviter de consulter en permanence l'aide générale.

6- Création d'un fichier script :

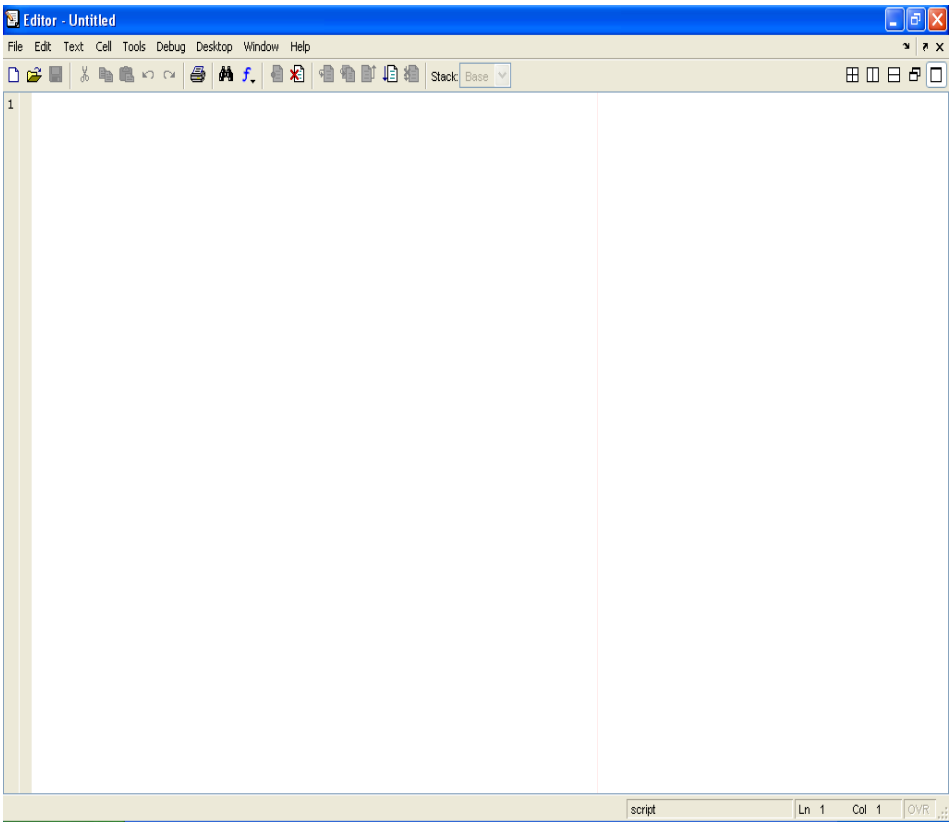
6.1- Ouvrir un script :

Un script est une suite de commande Matlab. Autrement dit, un script est un ensemble des instructions (commandes) Matlab qui joue le rôle de programme principal.

Pour lancer un script aller à l'onglet « File », « new » et puis « M-file ».



Une fois, vous cliquer sur « M-file », la fenêtre suivante apparait :



C'est dans cette fenêtre qu'on peut écrire l'ensemble des instructions Matlab. Il est en effet beaucoup plus simple de modifier des instructions dans un script que de retaper un ensemble d'instructions Matlab dans la fenêtre de commandes.

Exercice 1 :

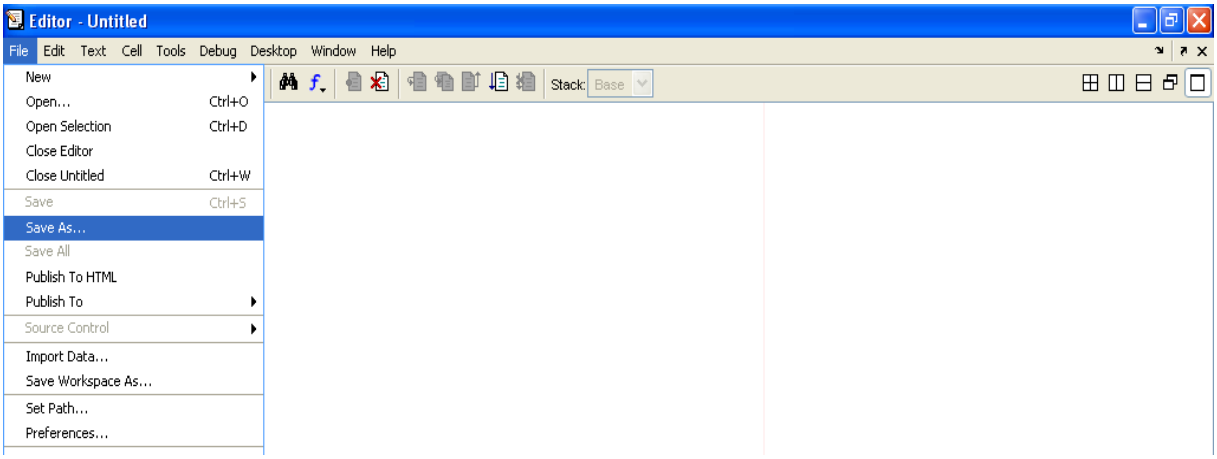
Ouvrir un nouveau script et taper le programme suivant :

```
a=2 ;
b=5 ;
c=(a+b)/2
```

6.2- Nommer et sauvegarder un script :

Une fois, vous écrivez le programme vous devez le sauvegarder et lui donner un nom (dans notre exemple, on va le nommé : programme).

- Cliquer sur « file » puis « save as »

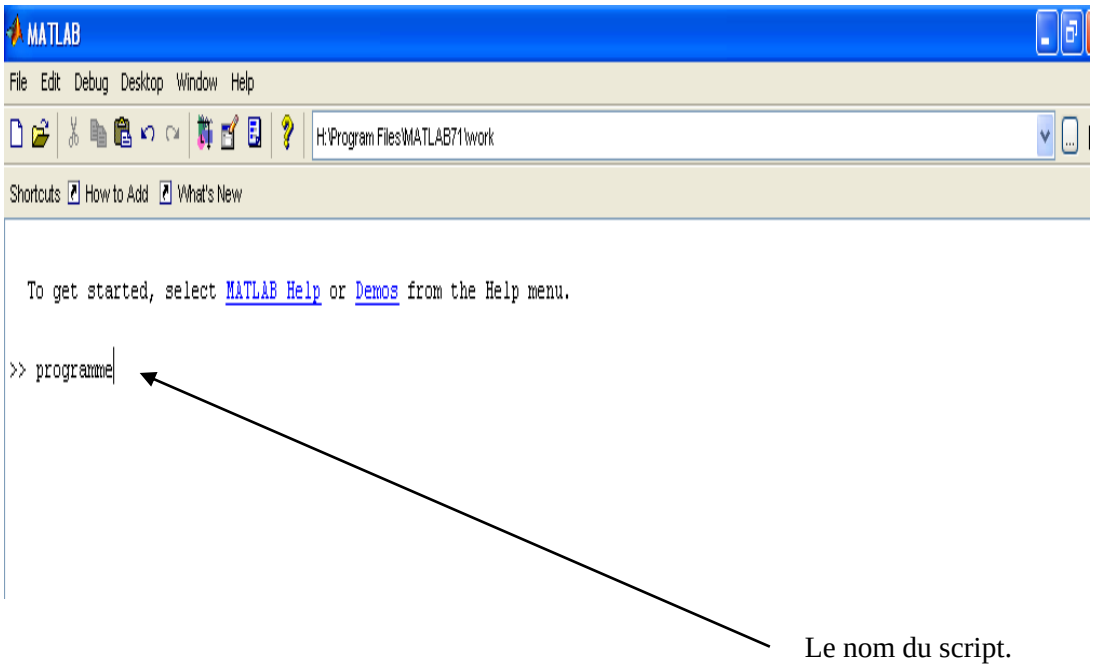


Remarques importantes :

- Le script doit avoir une extension de la forme « .m ».
- Dans l'appellation des scripts, **Il faut éviter** :
  - D'utiliser les caractères désignant un opérateur spécifique à Matlab tel que « - », « ; », « * »,…etc.
  - D'utiliser le nom d'une fonction Matlab telle que : sqrt, sin, cos,…etc.
  - D'utiliser deux mots séparées (par exemple : program st, program 1).

6.3- Exécution d'un script :

Pour exécuter un script, on écrit tout simplement son nom dans la fenêtre de commandes puis on clique sur entrer.



Si le script contient une (des) erreur(s), la ligne contenant l'erreur ainsi le type d'erreurs sont affichées dans la fenêtre des commande.

7- Travaux élémentaires sur les matrices :

Tous les objets manipulés par Matlab peuvent être considérés comme des matrices. La matrice :

$$A = \begin{bmatrix} valeur1 & valeur2 & valeur3 \\ valeur4 & valeur5 & valeur6 \end{bmatrix}$$

est définie sous Matlab comme suit: **A= [valeur1 valeur 2 valeur3 ; valeur4 valeur5 valeur6].** Un espace ou une virgule sépare deux éléments d'une même ligne et le point-virgule indique la séparation de ligne.

```
>> A=[ 2 0  -1 ; 1  2   2]
A =
   2    0    -1
   1    2     2
```

Matlab dispose également des moyens très simples pour créer des listes. Par exemple :

```
>> x=[ 1 :0.5 : 3]     %   générer un vecteur de 1 à3 avec un pas de  0.5.
x=
   1    1.5    2     2.5     3
 >>  y=[-2 : 2]            % [-2 : 2] est un raccourci pour [-2 :1 : 2].

    -2   -1     0     1      2
```

7.1- Les opérations sur les matrices :

Commande	Description
A+B, A-B	**Addition et soustraction terme à terme.**
A*B, B*A	**Produit matriciel standard.**
A.*B	**Multiplication terme à terme ; A et B doivent avoir le même format.**
A^n	**A^n élévation à la puissance au sens habituel.**
A.^n	**élévation à la puissance élément par élément.**
B/A	**Le résultat est une matrice X tel que XA=B** **Si A est inversible, alors $X=BA^{-1}$;** **Nb.col de A doit être le même que nb.lign de B.**

A\B	**Le résultat est une matrice X tel que AX=B,** **Si A est inversible, alors $X=A^{-1}B$;** **Nb.col de A doit être le même que nb.lign de B.**
A'	**Transposition et conjugaison.**
det(A)	**Renvoie le déterminant de A.**
rank(A)	**Renvoie le rang de A.**
eig(A)	**Renvoie les valeurs propres.**
poly(A)	**Renvoie le polynôme caractéristique de A.**

7.2- Matrices particulières :

Commande	Description
eye(n)	**Matrice identité carrée n par n**
eye(m,n)	**Matrice identité rectangulaire m par n.**
zeros(n)	**Matrice nulle carrée n par n.**
zeros(m,n)	**Matrice nulle rectangulaire m par n.**
ones(n)	**Matrice de 1 carrée n par n.**
ones(m,n)	**Matrice de 1 rectangulaire m par n.**

7.3- Manipulation sur les matrices :

Commande	Description
A(2, :)	**Sélection de la ligne 2**
A(:, 4)	**Sélection de la colonne 4.**
A(:, 1 :3)	**Sélection des colonnes 1 à 3.**
A(1 :3,1 :3)	**Sélection des lignes 1 à 3 et des colonnes 1 à 3.**
[A,B]	**Concaténation horizontale de A et B.**
[A ;B]	**Concaténation verticale de A et B.**

Exercice 2:

Saisir la matrice $A = \begin{bmatrix} 4 & 5 & 1 \\ 8 & 6 & 2 \\ 1 & 9 & 5 \end{bmatrix}$ et effectuer la série de calculs suivants :

1. B = A + A
2. C = A^2 ainsi que D = A. $*$ A puis comparer le résultat de C à celui de D.
3. Inverse d'une matrice.
4. Transposée de la matrice A.
5. Calculer le déterminant de A.
6. Déterminer les valeurs propres et les vecteurs propres de A.
7. Trouver le rang de la matrice A.
8. Saisir la matrice I_3.
9. Concaténation horizontale de A et B.
10. Concaténation verticale de A et B.
11. Extraction du terme de la première ligne, troisième colonne de A.
12. Extraction de la troisième ligne de A.
13. Extraire la deuxième colonne de A.

Polynômes

Un polynôme de degré m : $a_m p^m + a_{m-1} p^{m-1} + \cdots + a_1 p + a_0$ est défini sous Matlab comme un vecteur ligne :

$$P = \begin{bmatrix} a_m & \cdots & a_2 & a_1 & a_0 \end{bmatrix}$$

Les fonctions Matlab **roots** et **poly** permettent de trouver les racines d'un polynôme et les coefficients du polynôme connaissant ses racines. Pour déterminer les racines du polynôme $P(s) = s^3 + 3s^2 + 4$ on utilise les instructions suivantes:

```
» p=[1 3 0 4];
 » r=roots(p)   % ceci pour déterminer les racines de p
r =
  -3.3553
   0.1777 + 1.0773i
   0.1777 - 1.0773i
```

Pour déterminer les coefficients du polynôme connaissant ses racines par exemple :
Si les racines d'un polynôme sont -1, -2, -3, alors on fait :

```
» P=poly([-1  -2  -3 ]);
 » P=
 1      6     11      6
```

Le polynôme obtenu est : $x^3 + 6x^2 + 11x + 6$.
La multiplication de polynômes se fait à l'aide de la fonction **conv**.

```
» p=[3 2 1]; q=[1 4];
 »n=conv(p,q)
 n =
     3   14    9    4
```

Exercice 3 :

Soit le polynôme suivant : $5x^3 - 3x^2 + 4x - 6$

- Saisir ce polynôme sous MATLAB,
- Déterminer les racines de ce polynôme.
- Rétablir le polynôme P à partir des racines trouvées précédemment.

8- Graphisme :

Pour tracer un graphe dans Matlab, on fait appel à la commande « **plot »**. La fonction **plot(x,y)** trace le vecteur **y** en fonction du vecteur **x**. Par exemple, pour tracer, la fonction **y** définie par : $y(t) = \sin(0.3\pi x)$:

- Il faut tout d'abord déclarer le vecteur **x** (on suppose que **x** varie entre [0, 2π]).
- Taper en suite la formule de **y** [c.à.d.: $y = \sin(0.3\pi x)$].
- Finalement tracer le graphe de y(t) en utilisant la fonction « plot ».

```
clear all ;clc ;

x=[0 :pi/100 :2*pi] ;

y=sin(0.3*pi*x);

plot(x,y)
```

¨Pour tracer plusieurs courbes dans la même fenêtre graphique, on fait appel à la commande **plot(x1,y1,x2,y2,...).** Lorsqu'on trace deux graphes dans une même fenêtre, il est conseillé d'utiliser deux couleurs différentes ou bien deux types de trait différents.

```
clear all ;clc ;

x=[0 :pi/100 :2*pi] ;

y=sin(0.3*pi*x);

x1=x ;

y1=cos(x1) ;

plot(x,y,'+r',x1,y1,'ob')
```

Matlab possède plusieurs fonctions pour améliorer la qualité d'un graphe, les plus importantes sont :

Commande	Description
grid	**superpose (ou enlève) une grille sur la figure.**
title('texte'	**place la chaîne de caractère 'texte' en haut du graphe.**
xlabel('texte')	**place la chaîne de caractère 'texte' sur l'axe des X.**

ylabel('texte')	**place la chaîne de caractère 'texte' sur l'axe des Y.**
hold on	**gèle les caractéristiques graphiques de la fenêtre active.**
hold off	**revient au mode normal.**
subplot(m,n,p)	**division d'une fenêtre graphique en m*n sous-fenêtres,** **l'entier p indique le numéro de la cellule où se fait le tracé.**
legend('titre1','titre2',...)	**légende pour chaque courbe du graphique.**

Exercice 4 :

Générer un vecteur x variant de 1 à 10 avec un pas de 0.1. y1 est le carré de x et y2 est son sinus.

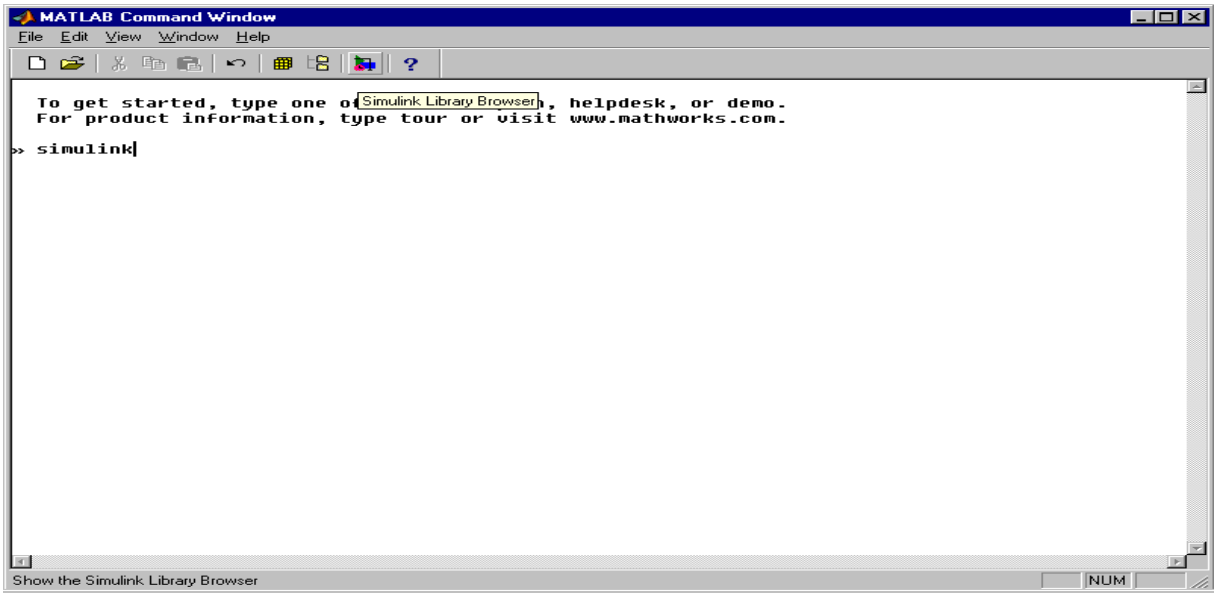
Tracer sur un même graphe la courbe y1 en fonction de x et y2 en fonction de x avec différentes couleurs et des légendes.

9- Simulink

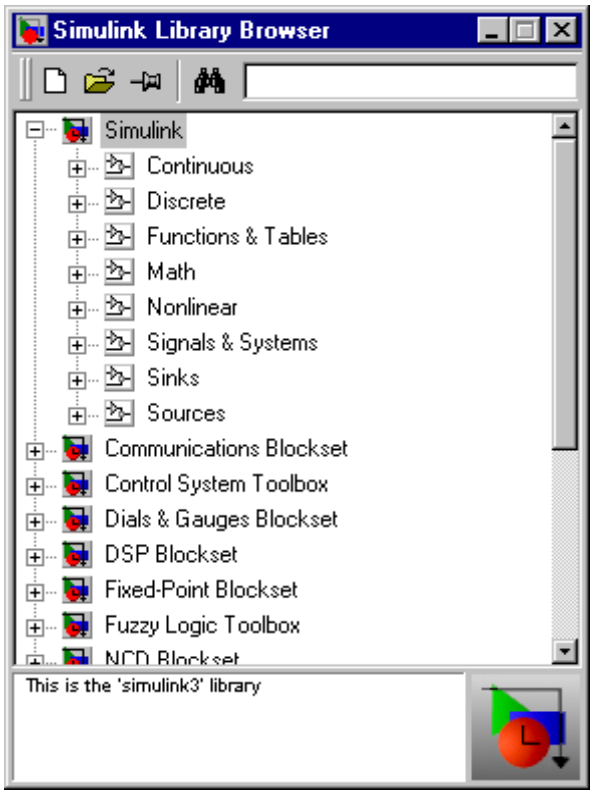
Simulink est un programme pour la simulation des systèmes dynamiques. Il utilise le principe des représentations en schémas blocs. Simulink possède deux phases d'utilisation:

- La définition du modèle
- L'analyse du modèle

On lance Simulink en cliquant sur l'icône  dans la barre d'outils de Matlab, ou bien par la commande >>simulink à l'invite, dans la fenêtre de commande comme indiquer dans la figure suivante :

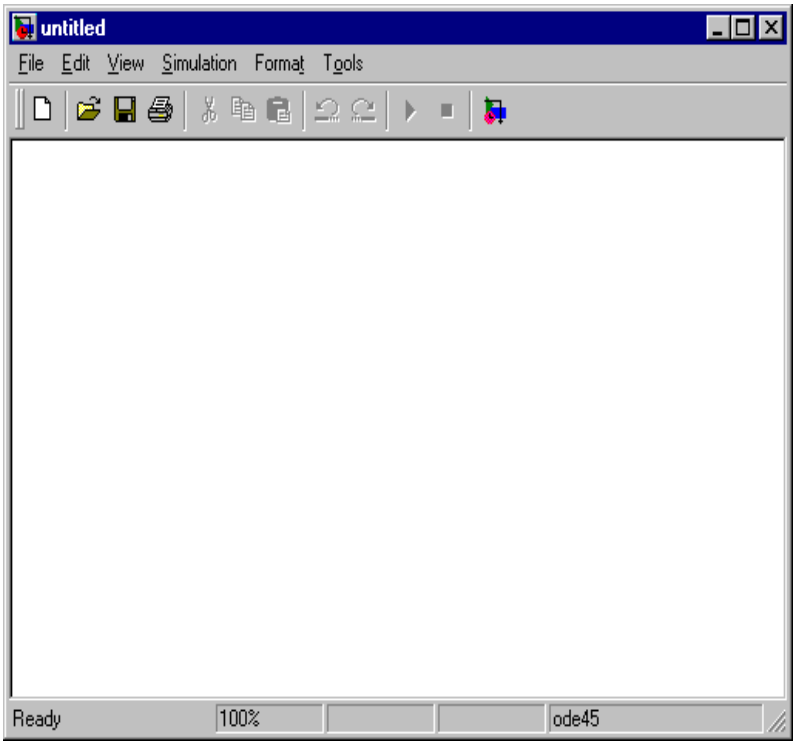


Cette commande affiche une nouvelle fenêtre contenant les icônes des blocs des sous-systèmes constituant la librairie standard de Simulink.



La librairie standard de Simulink est organisée en plusieurs sous-systèmes groupés en **blocs** (ou boîtes) selon leur comportement.

Après avoir obtenu la fenêtre principale, sélectionnez **New**...**Model** dans le menu **File**. Une fenêtre s'ouvre dans laquelle vous allez pouvoir saisir vos schémas-blocs. La nouvelle fenêtre est nommée "untitled" ; on peut lui changer de nom lors de la sauvegarde.



Pour construire votre schéma-bloc, il suffit d'aller chercher les blocs désirés à partir de la fenêtre principale de Simulink présentant les différentes bibliothèques disponibles. Ces blocs peuvent être insérés dans votre fenêtre de travail soit en utilisant la méthode du « copier-coller » soit en faisant glisser le bloc d'une fenêtre à une autre à l'aide de la souris.

Comment lancer la simulation ?

Avant de lancer la simulation, il faut impérativement définir les paramètres de votre simulation : instant de début de simulation, instant de fin de simulation, période de simulation, méthode d'intégration numérique utilisée... Pour cela, il faut aller dans le menu « Simulation » puis sur «parameters » et spécifier les paramètres adaptés à votre simulation.

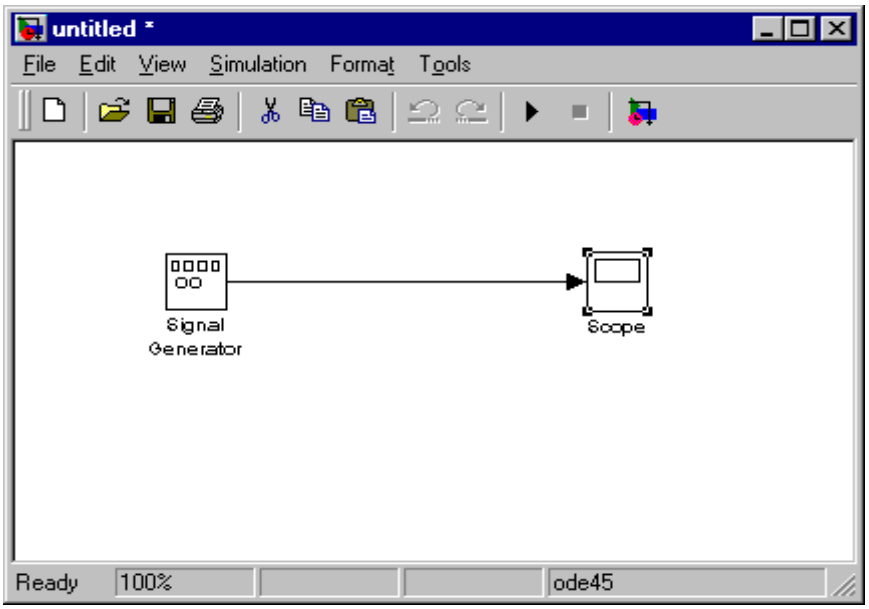
Pour lancer ensuite votre simulation il suffit d'aller dans le menu « Simulation » et de sélectionner « start ». Si des blocs de visualisation (graph, scope, …) apparaissent dans votre schéma-bloc, veillez à paramétrer ceux-ci en fonction des paramètres de votre simulation.

Comment arrêter votre simulation ?

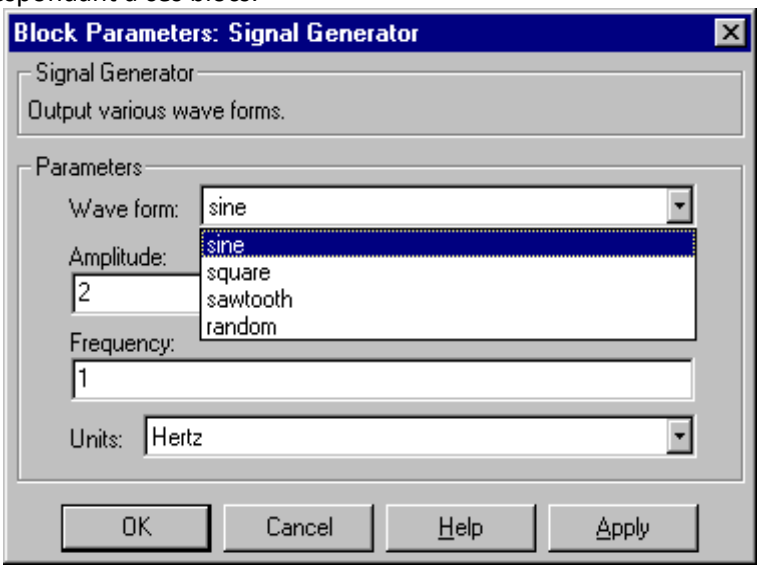
Si vous avez spécifié un instant de fin de simulation, la simulation prendra fin lorsque cet instant sera atteint. Vous pouvez néanmoins arrêter à tout moment une simulation en cours en allant dans le menu « Simulation » et en cliquant sur « stop » ou en cliquant sur le bouton stop.

Exemple: Simulation d'un signal sinusoïdal :

1- Pour saisir votre schéma, ouvrez la bibliothèque **Sources**, sélectionnez l'icône «Signal generator» en «cliquant» une fois dessus, et faîtes glisser celle-ci dans votre fenêtre de travail.
2- ouvrez **Sinks** et sélectionnez le bloc **scope** et faîtes glisser celui-ci dans votre fenêtre de travail.
3- A l'aide de la souris, reliezla sortie du bloc générateur de signal à l'entrée de l'oscilloscope. L'oscilloscope permet de visualiser une partie du signal à l'écran.



4- A l'aide d'un double clic, ouvrir un ou plusieurs blocs et changer certains des paramètres internes correspondant à ces blocs.



5- Lancer la simulation en sélectionnant "start" du menu **Simulation**.
6- Pour visualiser le signal sur l'oscilloscope, il suffit de cliquer deux fois sur le bloc Scope. Scope peut s'ouvrir même si la simulation a déjà commencé.

TP 2 : Analyse temporelle des systèmes du premier et deuxième ordre.

1- Objectif :

L'objectif de ce TP est de pouvoir déterminer les réponses temporelles des systèmes dynamiques du 1er et du 2eme ordre (réponse indicielle, réponse impulsionnelle, …) ainsi que la détermination des caractéristiques temporelles d'un système (temps de réponse, constante du temps, erreur, …).

2- Fonction de transfert :

Une fonction de transfert est un modèle mathématique de la relation entrée-sortie d'un système linéaire. Elle est représentée sous la forme suivante :

$$G(s)=\frac{a_m s^m + a_{m-1}s^{m-1} + \cdots + a_1 s + a_0}{s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0}$$

Où, $m \leq n$. Les racines du numérateur sont les zéros du système. Les zéros du dénominateur sont les pôles du système. La réponse transitoire du système est liée aux pôles et zéros du système. L'équation caractéristique est donnée par : $s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0 = 0$.

2.1- Représentation d'une fonction de transfert :

Pour définir une fonction de transfert sous Matlab, on utilise la commande « **tf** » en spécifiant le polynôme de numérateur et le polynôme de dénominateur. Par exemple pour afficher $H(s)=\frac{s-1}{s^2+s+2}$, on utilise les instructions suivantes:

```
» num=[1 -1];    % charger le polynôme s-1
» den=[1 1 2];    % charger le polynôme s^2+s+2
» h= tf(num,den)
Transfer function:
        s - 1
    -------------
    s^2 +  s + 2
```

2.2- Représentations poles/zéros/gain :

Pour calculer les pôles et les zéros d'une fonction de transfert, on utilise tout simplement les commandes **pole(h)** et **zero(h)** respectivement. Par exemple :

```
» pole(h)
  ans =
-0.5000 + 1.3229i
-0.5000 - 1.3229i

» zero(h)
 ans =
 1
```

La fonction **pzmap** permet de représenter la position des pôles et des zéros dans le plan complexe. La syntaxe est la suivante:

```
» [poles,zeros]=pzmap(num,den)  %ou [poles,zeros]=pzmap(h)

  poles

-0.5000 + 1.3229i
-0.5000 - 1.3229i

    zeros =

       1
```

Si on invoque uniquement **pzmap(num,den)** ou **pzmap(h)** , une figure sur laquelle est indiqué la position des pôles et des zéros du système s'ouvrira.

La fonction **zpkdata** permet de calculer les zéros, les pôles et le gain statique k.

» [z,p,k]=zpkdata(h)

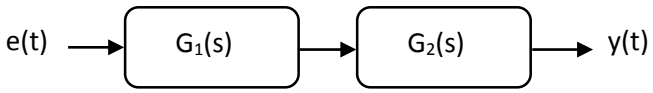
Inversement, toutes représentation d'un système linéaire à temps invariant à partir de la liste de ses pôles et zéros se fait à partir de la fonction **zpk(zer,pol,gain)**. Elle consiste à factoriser les racines des polynômes numérateur et dénominateur. Par exemple :

```
» G=zpk([1 -2],[1 2 -3],5)

Zero/pole/gain:

  5 (s-1) (s+2)
 ----------------
(s-1) (s-2) (s+3)
```

3- Opération sur les fonctions de transfert :

Il est possible d'assembler 2 fonctions de transfert entre elles :

- **En série:**

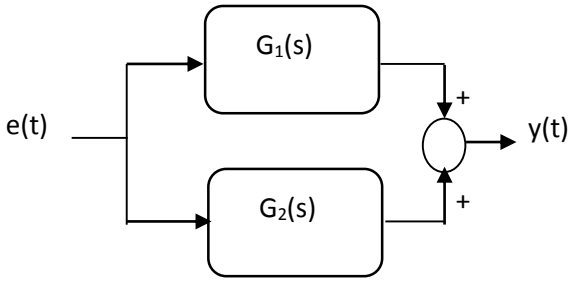


Soient $G_1(s) = \frac{num1}{den1}$; $G_2(s) = \frac{num2}{den2}$; $G(s) = \frac{Y(s)}{U(s)} = \frac{num}{den}$

Le système équivalent s'obtient par l'une des instructions suivantes :

>> [num,den]=series(num1,den1,num2,den2)
>> G=series(G1,G2)
>> G=G1*G2;

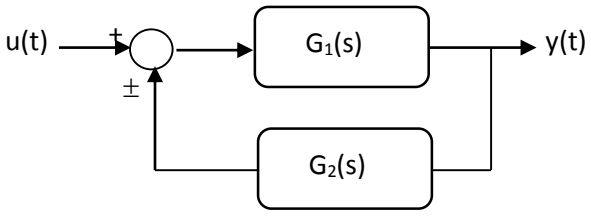
- **En parallèle**:



De même le système équivalent s'obtient par l'une des instructions suivantes :

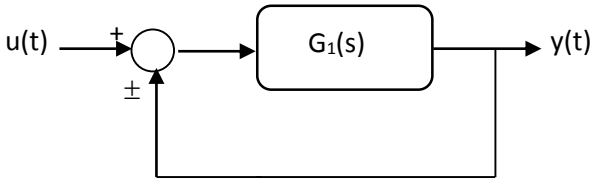
>> [num,den]=parallel(num1,den1,num2,den2)
>> G=parallel(G1,G2)
>> G=G1+G2;

- **En boucle fermé à retour non unitaire:**



>>[num,den]=feedback(num,den,num1,den1, ±1)
>> G= feedback(G1,G2, ±1)

En boucle fermé à retour unitaire



>>[num,den]=feedback(num,den, ±1)
>> G= feedback(G1, ±1)

Remarque: ±1 signifie un feedback positif ou négatif (par défaut égal à -1).

4- Réponses temporelles des systèmes de premier ordre :

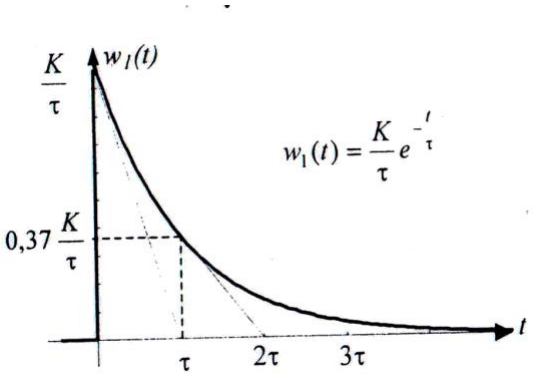
On appelle réponse temporelle d'un système l'évolution de sa sortie en fonction du temps, suite à l'application d'une entrée en partant de l'état de repos.
Un système du premier ordre est donné sous la forme suivante *:*

$$G(s) = \frac{K}{1 + \tau s}$$

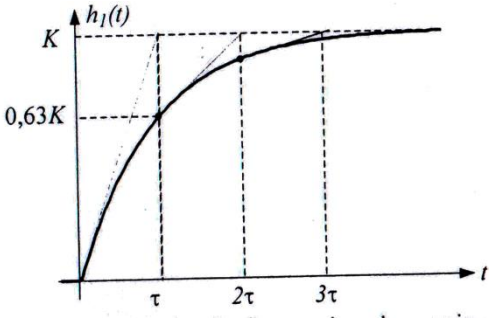
avec :

K : Gain statique.

τ : Constante de temps du système.



Réponse impulsionnelle $S(t) = \frac{K}{\tau} e^{-t/\tau}$



réponse indicielle $S(t) = K(1 - e^{-t/\tau})$

Matlab met à la disposition de l'utilisateur plusieurs commandes dont le rôle est d'injecter un certain signal à un système et de fournir la réponse temporelle à ce signal. Parmi ces commandes, distinguons celles qui nous semblent les plus utiles.

Commande	**Descriptif**	**Syntaxe**
step	Permet de donner la réponse indicielle du système	step(G)
impulse	Permet de donner la réponse impulsionnelle du système (c.à.d. la réponse à une impulsion de Dirac).	impulse(G)
lsim	Cette commande permet de donner la réponse d'un système à une entrée quelconque.	lsim(g,u,t) avec: **u** l'entrée du système

5- Réponses temporelles des systèmes de deuxième ordre :

On considère comme système de 2nd ordre tout système décrit par une équation différentielle du second ordre dont la fonction de transfert peut se mettre sous la forme :

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

Où:

K : Le gain statique du système

ξ : Le coefficient d'amortissement du système

ω_n : La pulsation propre du système.

Le comportement dynamique d'un tel système dépend de la valeur des deux constantes ω_n et surtout de ξ. Si on cherche les pôles de la fonction de transfert (racines du dénominateur), on distingue trois cas possibles :

- Si $\xi > 1$: Le système admet deux pôles réelles de la forme :

$$p_{1,2} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$$

Sa réponse indicielle s'écrit sous forme d'une somme de deux exponentielles, correspondant à la mise en cascade de deux systèmes du 1er ordre de constantes de temps $\tau 1$ et $\tau 2$.

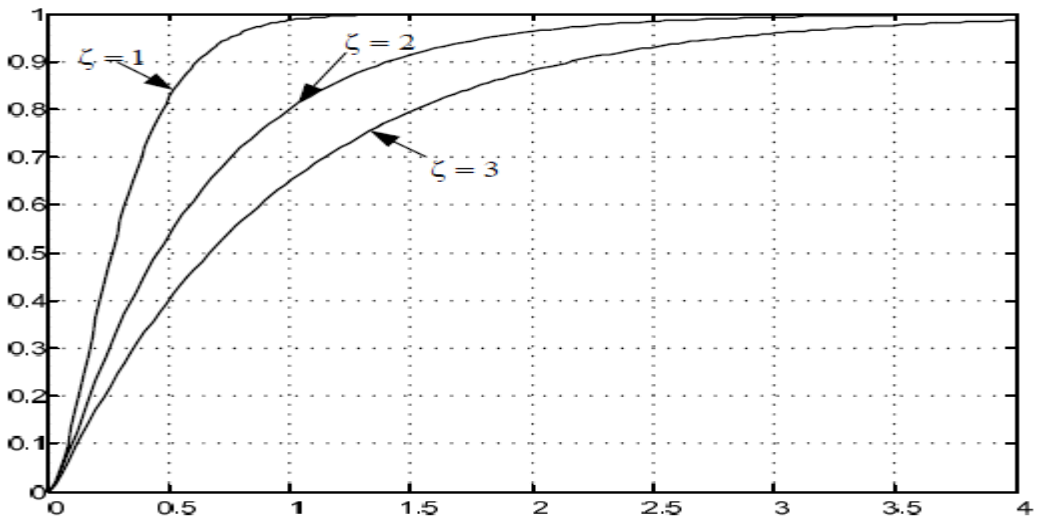
$$s(t) = KE_0\left(1 - \frac{\tau_1}{\tau_1 - \tau_2}e^{-\frac{t}{\tau_1}} + \frac{\tau_2}{\tau_1 - \tau_2}e^{-\frac{t}{\tau_2}}\right)u(t)$$

- Si $\xi = 1$: Le système admet un pôle réelles double : $p_{1,2} = -\omega_n$

Sa réponse indicielle s'écrit :

$$s(t) = KE_0\left(1 - (1 + \omega_n t)e^{-\omega_n t}\right)$$

Dans les deux cas précédents, le comportement est non oscillant et non amorti comme indiquer dans la figure suivante :



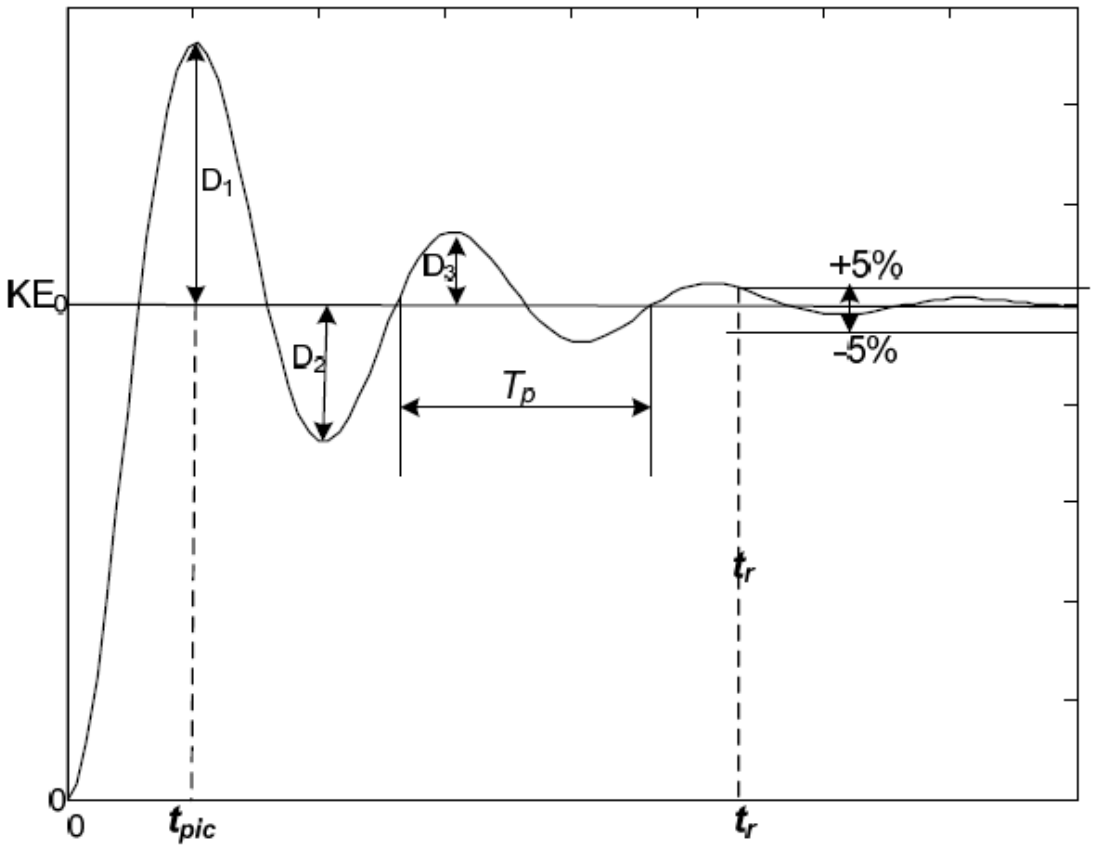
- Si $\xi < 1$: Le système admet deux pôles complexes conjugués de la forme:

$$p_{1,2} = -\xi\omega_n \pm j\omega_n\sqrt{\xi^2 - 1}$$

Sa réponse indicielle s'écrit :

$$s(t) = KE_0\left(1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}}\sin(\omega_n\sqrt{1-\xi^2} + \varphi)\right)$$

Le comportement dans ce cas est oscillant et amorti comme indiquer dans la figure suivante :

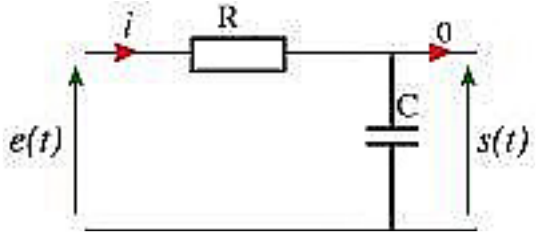


6- Travail demandé :

6.1- Etude théorique :

Partie I:

Soit le circuit RC donné par la figure suivante



- Etablir l'équation différentielle qui régit le fonctionnement de ce circuit si les grandeurs sont quelconques.
- Donner la fonction de transfert de ce système.
- Donner l'expression de la constante de temps et le gain statique correspondant au système en fonction de R et C.

On considère que R = 10 kW et C = 100 nF.

- Calculer les réponses indicielles, impulsionnelle et à une rampe.
- Donner commentaires et conclusions.

Partie II:

- Donner un exemple de système physique du 2ème ordre (schéma du système)
- Donner l'équation différentielle et la fonction de transfert de ce système.
- Calculer la réponse indicielle, en mettant en valeur les cas suivants : $\xi = 1$, $\xi < 0$, $\xi > 0$
- donner l'allure de la réponse indicielle dans la même figure pour différentes valeurs de ξ.
- Donner vos observations, commentaires et conclusions.

6.2- Etude pratique (étude sous Matlab) :

Partie I:

Soit le système sys1 représenté par une fonction de transfert du premier ordre de gain égale à 5 et de constante de temps égale à 3.

1- Dans un fichier script, écrire le programme qui permet de :
   a- Donner la fonction de transfert.
   b- Tracer la réponse impulsionnelle.
   c- Tracer la réponse indicielle.
   d- Tracer la réponse à une rampe.

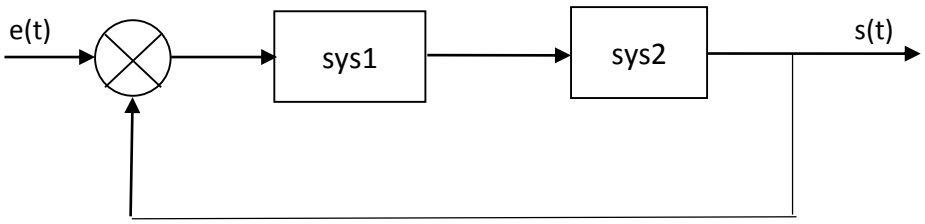
2- Déterminer à partir de la réponse indicielle :
   a- Temps de réponse à 5%.

b- Valeur de la sortie en régime permanent.

3- pour **τ=1s** et ***K*** **=0.5** et **1.5** enregistrer la réponse indicielle du système en boucle ouverte puis en boucle fermée et mesurer le temps de réponse à 5%. Quel est l'effet de la valeur de ***K*** sur le temps de réponse $\mathbf{t_r}$. Conclure.

4- Pour ***K*** **=1** et **τ=0.5s et 1s** enregistrer la réponse indicielle du système en boucle ouverte puis en boucle fermée et mesurer le temps de réponse à 5%. Quel est l'effet de la valeur de **τ** sur le temps de réponse $\mathbf{t_r}$. Conclure.

5- Soit un système **sys2** est placé en série avec le système **sys3**, comme indiquer dans la figure suivante, avec $sys2 = \frac{5}{5s+1}$ et $sys3 = \frac{3}{4s+1}$



a- Dans le même fichier script :
- Déterminer la fonction de transfert en boucle fermé.
- Tracer les réponses impulsionnelle et indicielle du système bouclé.

b- Dans un fichier simulink, visualiser les réponses impulsionnelle et indicielle du système bouclé dans le même oscilloscope.

Partie II:

1- Soit un système du second ordre de la forme suivante :

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

En prenant les valeurs suivantes : **K=1**, $\boldsymbol{\omega_n}$**=10rd/s**,

a- Tracer, sur la même figure, les réponses indicielles à un échelon unitaire pour chaque valeur de ξ avec $\xi = 0.1, 0.7, 1, 1.2.$

b- Retrouver pour chaque valeur de ξ:
- Le temps de réponse à 5% (T_r).
- Le temps du pic (T_{pic}).
- Le dépassement D en %.
- Que peut-on en conclure ?

c- Quel est les l'effet, pour la réponse indicielle du système en boucle ouverte puis en boucle fermée d'une variation de la valeur k (**ζ** et **ω_n** sont constantes) sur les paramètres de la question b. Conclure.

d- Quel est les l'effet, pour la réponse indicielle du système en boucle ouverte puis en boucle fermée d'une variation de la valeur **ω_n** (**ζ** et k sont constantes) sur les paramètres de la question b. Conclure.

2- Considérons un système asservi représenté par la fonction de transfert en boucle fermée :

$$F(s) = \frac{3.06}{(s^2 + s + 1)(s^2 + 0.5s + 3.06)}$$

1/ A l'aide de Simulink, visualiser en même temps sur un oscilloscope l'entrée en échelon et la sortie indicielle de F(s).

2/ On veut approximer F(s) par un système $F_1(s) = \frac{3.06}{(s^2 + 0.5s + 3.06)}$

Pour cela, visualiser sur un oscilloscope la réponse indicielle de $F_1(s)$.

3/ Pour pouvoir comparer les deux réponses tracer les en même temps sur un même oscilloscope.

4/ Que peut-on en conclure ?

TP 3 : Analyse fréquentielle des systèmes du premier et deuxième ordre.

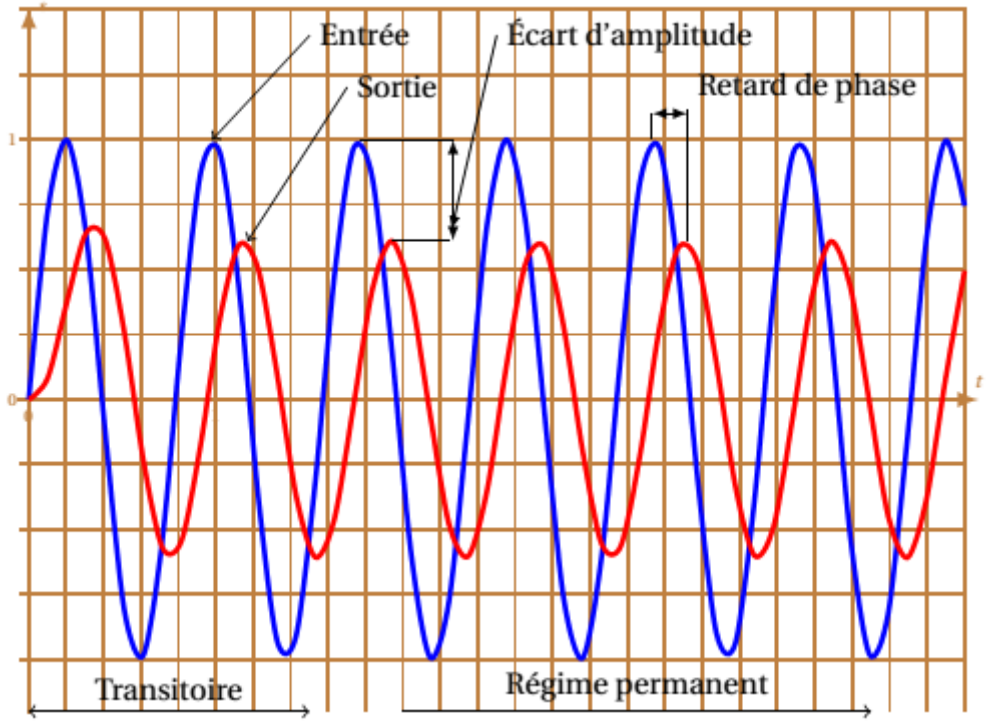
1- Objectif :

Le but de ce TP est d'étudier les systèmes dynamiques dans le domaine fréquentiel en introduisant les commandes permettant de tracer les réponses harmoniques (fréquentielle), et de tirer les caractéristiques fréquentielles d'un système.

2- Réponses fréquentielles:

L'analyse fréquentielle permet de connaître la réponse du système à une excitation sinusoïdale, à différentes fréquences. La réponse en fréquence du système est l'étude du régime permanent.

La sortie d'un système linéaire sollicité par un entrée sinusoïdale est de forme sinusoïdale de même pulsation que le signal d'entré mais d'amplitude différente et déphasé par rapport au signal d'entrée.

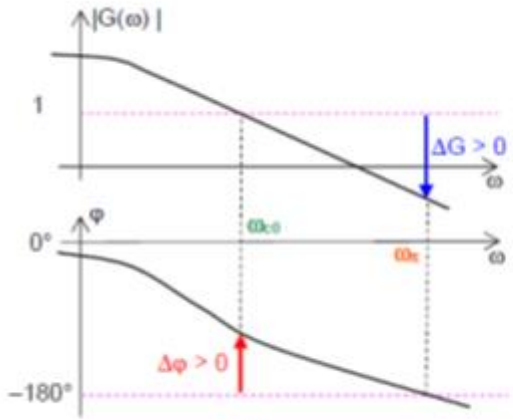


L'amplitude est modifiée par le gain du système à la pulsation ω, |G(ω)|. Il est déphasé de Arg [G(ω)] par rapport à l'entrée.

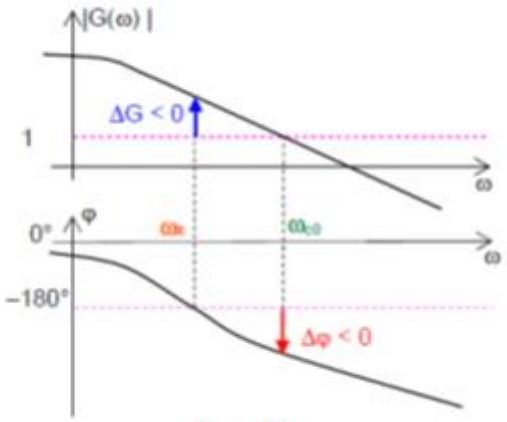
Pour **ω** fixée, **G(jω)** est un nombre complexe caractérisé par son amplitude(le gain du système) et son argument (la phase du système) qui seront donc des fonctions de la pulsation de la sinusoïde d'entrée. Quand **ω** varie l'ensemble des gains et des arguments constitue la réponse fréquentielle qui peut être alors représenté graphiquement dans différents types de plans.

Comme **G(jω)** est une fonction de la variable complexe, on a le choix entre plusieurs types de représentation.

- On peut conserver l'information de fréquence (pulsation) : on doit donc tracer le module et la phase en fonction de ω sur deux graphes séparés, on parle alors du plan de **Bode**.

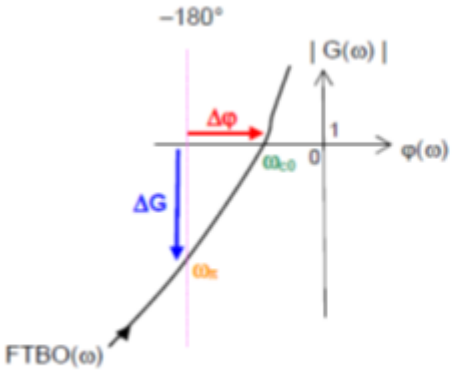


Stable

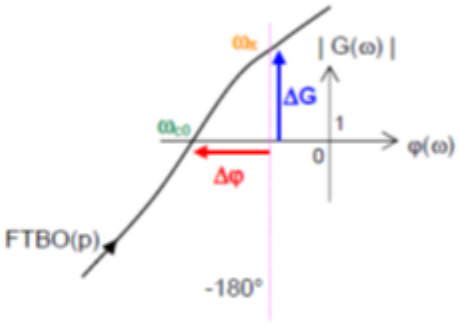


Instable

- On peut tracer le module en fonction de la phase sur un seul graphe. La courbe obtenue sera alors paramétrée en ω, mais la pulsation n'intervient plus directement : on parle alors du plan de **Black-Nichols**.

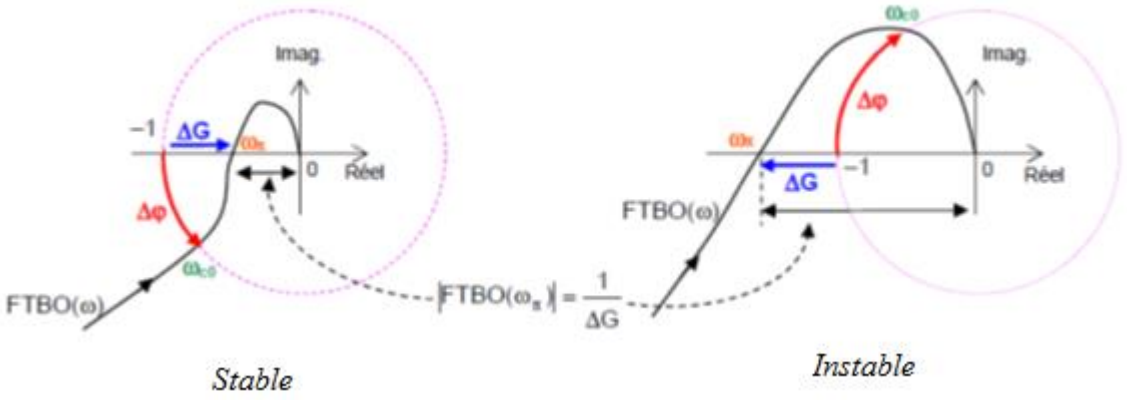


Stable



Instable

- Il est possible aussi de passer d'une représentation module/phase à une représentation de type partie réelle/partie imaginaire : **H(jω) = Re(ω) + jIm(ω)**, on trace la partie imaginaire en fonction de la partie réelle, le graphe obtenu est paramétré en ω : on parle alors du plan de **Nyquist**.



3- Bande passante :

On appelle bande passante, l'intervalle de pulsations ou le module de la FTBO est supérieur à un ; ce qui donne une expression approximative de la fonction de transfert en boucle fermée :
|FTBF| = 1, dans le cas d'un retour unitaire.

1- Aux basses fréquences **:** $|FTBF| \to 1$
2- Aux basses fréquences : $|FTBF| \to 0$

Cela veut dire qu'un système asservi (processus physiques) est, en général, un filtre passe-bas dont la sortie suit l'entrée.
Fixer une bande passante impose un temps de réponse (plus elle est grande, plus le temps de réponse est court).

4- Représentation de réponses fréquentielles sous Matlab :

L'étude fréquentielle est souvent envisagée pour le réglage en boucle fermée d'un système. Cette approche permet d'établir la notion de marge de gain et marge de phase d'un système. Afin de représenter la réponse harmonique d'un système, Matlab met à la disposition de l'utilisateur plusieurs commandes. Parmi ces commandes, distinguons celles qui nous semblent les plus utiles.

- **Tracé de Bode :**

Soit un système de fonction de transfert $G(s)=\frac{Num}{Den}$. Les différentes commandes pour analyser la réponse fréquentielle à l'aide de tracé de bode sont arrangées dans le tableau suivant :

Commande	**Descriptif**
bode(G)	Permet d'afficher le tracer de Bode sur un intervalle fréquentiel choisi de façon automatique.
bode(G,w)	Permet d'afficher le plan de Bode sur l'intervalle de pulsations défini par le vecteur w.
w=logspace(a,b,n)	Permet de génèrer un vecteur de **n** points entre 10^a rad/sec et 10^b rad/sec.
bode(G1,G2,…,w)	Est identique à **bode(G)** mais appliquée aux différents systèmes Gi. Les i tracés se font sur la même figure.
[gain,phase,w]= bode(G)	Permet de récupérer sous forme de tableaux le module (gain) et la phase (phase) de la réponse fréquentielle.
Margin(G)	Permet de tracer le lieu de Bode et d'indiquer dessus les marges de gain et de phase.
[MG,MP,wcg,wcp]= margin(G)	Permet d'indiquer sous forme de valeurs numériques la marge de gain **MG**, de phase **MP** et les fréquences associées **wcg, wcp.**
db(x)	Permet de convertir **x** en décibels.

- **Tracé de Black**

Le tracé dans le plan de Black est obtenu avec la fonction **nichols.** L'utilisation de cette fonction est semblable à celle de la fonction **bode** aussi nous ne donnerons que la syntaxe :

nichols(G)

nichols(G,w)

nichols(G1,G2,…)

ngrid : permet de superposer au tracé obtenu avec la fonction nichols l'abaque de nichols.

[gain,phase,w] = nichols(G,…) le lieu de Black n' est pas tracé.

- **Tracé de Nyquist**

Le tracé de Nyquist s'obtient à l'aide de l'instruction **nyquist (G).**

L'utilisation de cette fonction, pour la plupart des cas de figures, est semblable à celle de la fonction **bode.** La syntaxe d'utilisation est la suivante :

nyquist (G,w)

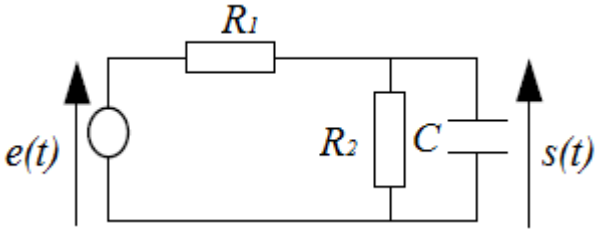
nyquist (G1,G2,…)

[re,im,w]= nyquist (G) renvoie la partie réelle et la partie imaginaire de la réponse fréquentielle dans les matrices **re** et **im** calculées pour les fréquences **w**.

5- Travail demandé :

Etude théorique :

1- Soit le circuit RC donné par la figure suivante :



- Déterminer la fonction de transfert *H(s)* du circuit et montrer qu'il s'agit d'un système du premier ordre.
- Déterminer le gain statique **K** et la constant de temps **τ** en fonction de ***R₁***, ***R₂*** et ***C***. On prend K=10 et **τ**=1 :
- Donner l'expression du gain complexe et en décibel de *H(s).*
- Donner l'expression de la phase et déduire la pulsation de coupure du système ω_c.

2- Soit un système du second ordre de la forme suivante :

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

- Donner l'expression du gain complexe et en décibel de *G(s)* ainsi que l'expression de la phase.

On prend ω_n=10rd/s, et$\xi = 0.7$.

- Pour quelle valeur de ***K*** ce système peut-il être stable en boucle ouverte et en boucle fermée (Critère de Routh).

Etude pratique (étude sous Matlab) :

A- On considère le système du premier ordre suivant *:*

$$G(P) = \frac{K}{1 + Ts}$$

On prend $T = 0.1\ et\ K = 1$

1. Tracer les allures des courbes dans le tracé de Bode, Nyquist et black.

2. Calculer les marges de stabilité.

3. Quel est l'effet d'une variation de la valeur du gain K ($T = cte$) sur la stabilité, la rapidité et la précision à partir de tracé de Bode.

4. Quel est l'effet d'une variation de la valeur de T ($\mathrm{K} = cte$) sur la stabilité, la rapidité et la précision à partir de tracé de Bode.

A- On considère le système du deuxième ordre suivant ***:***

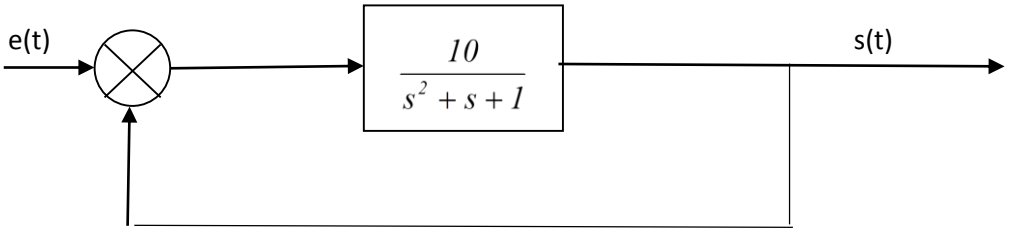
$$F(P) = \frac{k}{\frac{s^2}{w^2{}_n} + \frac{2\zeta}{w_n}s + 1}$$

On prend $\omega_n = 10rd/s\ et\ K = 1$

1- Tracer les allures des courbes dans le diagramme de Bode, Nyquist et black pour **ζ=0.3 ; 0.7 ; 1**. Conclure sur l'effet de ζ sur la stabilité, la rapidité et la précision.

2- Quel est l'effet d'une variation des valeurs K et ω_n sur la stabilité, la rapidité et la précision à partir de tracé de Bode.

B- On considère le système représenté par la fonction de transfert suivante :



- Quelle est l’ordre du système ?
- Calculer les pôles de système.
- Visualiser la réponse indicielle. conclure sur la stabilité.
- Faire le tracé de nyquist du système
- Conclure sur la stabilité.
- Calculer la fonction de transfert en boucle fermée (retour unitaire).
- Calculer les pôles de système bouclé.
- Visualiser la nouvelle réponse indicielle sous Simulink.
- Conclure.

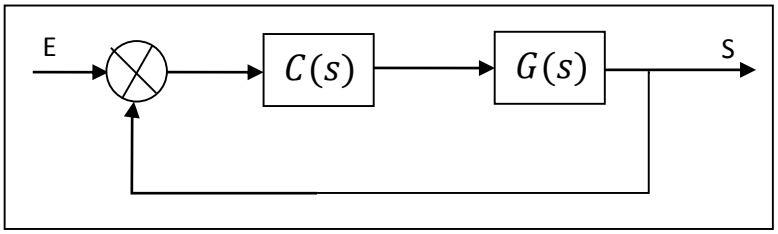
TP 4 : Correction des systèmes dynamiques linéaires.

1- Objectif :

Le but de ce TP est de fournir les outils afin d'améliorer les performances d'un système dynamique linéaire (en termes de stabilité, précision et rapidité). Ces outils sont, bien sûr, les correcteurs.

2- Introduction :

Une configuration générale de système bouclé avec un retour unitaire est la suivante:



Le correcteur C(p) est placé en série avec le système G(p) (il doit toujours être placé en amont). Le problème est de choisir le correcteur C(p) et de régler ses paramètres pour que l'asservissement fonctionne correctement c'est-à-dire faire suivre le mieux possible la sortie du système **S** vers le signal de référence **E** malgré la présence de perturbations qui agissent sur le système.
Ce réglage est effectué de manière à respecter des spécifications imposées par un cahier des charges. Ainsi, il peut être demandé d'avoir :

- une bonne précision statique (régime permanent).
- un faible temps de réponse et dépassement limité.
- un bon rejet des perturbations, problème de régulation.
- un bon suivi de trajectoire, problème de poursuite.
- de bonne marge de stabilité.
- une bonne robustesse (garantir le bon fonctionnement sur le procédé réel)
- …

3- Caractéristiques principales des correcteurs P,PI, PD, PID :

a- Correcteur Proportionnel (P): $C(p) = K_p$

Un correcteur proportionnel est un gain inséré en amont (avant) du système. (dit aussi l'action Proportionnelle), il peut :

- Améliore la précision en réduisant l'erreur statique.
- Peut améliorer la rapidité du système (pour des valeurs de K_p grandes)

MAIS

Une grande augmentation du gain K_p peut causer une instabilité du système.

b- Correcteur Proportionnel et Intégrale (PI) : $C(s) = K_p\left(1 + \frac{1}{T_i s}\right)$

K_p : Le gain de l'action proportionnelle.

T_i : La constante de temps de l'action Intégrale.

Un correcteur proportionnel et Intégral (PI) peut:

- Améliore la précision (il annule l'erreur en régime permanent).

  MAIS
- Diminue la stabilité (en réduisant la marge de phase).
- Ralentit le système.

c- Correcteur Proportionnel et Dérivé (PD)/ Correcteur à avance de phase :

$C(s) = K_p(1 + T_d s)$

K_p : Le gain de l'action proportionnelle

T_d : La constante de temps de l'action dérivée.

Le gain de ce correcteur est infini pour les hautes fréquences. Ceci est donc physiquement irréalisable, on l'approxime par un correcteur à avance de phase dont la fonction de transfert est : $C(s) = K_p\left(\frac{1+aT_d s}{1+T_d}\right)$ avec a>1.

Un correcteur à avance de phase peut :

- Améliore la stabilité du système en introduisant un déphasage positif d'où le nom de correcteur à avance de phase.
- Augmente la rapidité du système.

  MAIS
- Il n'améliore pas la précision.

d- Correcteur Proportionnel Intégral et Dérivé (PID) : $C(s) = K_p\left(1 + \frac{1}{T_i s} + T_d s\right)$

K_p : Le gain de l'action proportionnelle

T_i : La constante de temps de l'action Intégrale.

T_d : La constante de temps de l'action Dérivée.

Le correcteur PID permet de bénéficier des avantages du correcteur PD et ceux du PI. Il peut :

- Améliore la précision (il annule l'erreur)
- Améliore la stabilité du système.
- Améliore la rapidité du système.

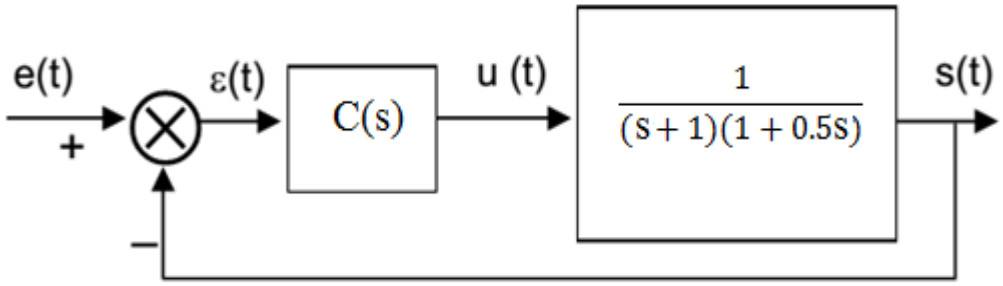
4- Travail demandé :

On considère un système linéaire donné par la fonction de transfert suivante *:*

$$G(s) = \frac{1}{(s+1)(1+0.5s)}$$

1- Tracer la réponse indicielle, le tracé de Bode et le diagrame de Niqyist de G(S). Conclure sur la stabilité de G(S).

Si on désire d'améliorer les caractéristiques de stabilité, précision, et de rapidité, sans modifier **G(S)** il est nécessaire d'introduire dans la boucle de commande un correcteur **C(S)** comme indiquer dans la figure suivante :



2- Correcteur Proportionnel (P): C(s)=Kp, On prend Kp=0.1,0.5, 1, 3, 9, 19.

- Quel est l'effet d'une variation de Kp sur la stabilité, la rapidité et la précision du système à partir de :
  a. La réponse indicielle en boucle fermée (en terme de D %, tr, t_{pic}, …).
  b. Le tracé de Bode.
- Conclure

3- Correcteur Proportionnel et Intégral (PI) : $C(s) = K_p + \frac{K_i}{s}$.

Nous fixons **Kp = 5**, et nous varions la valeur de **Ki** (Ki=0, 2.5, 5, 10 et 15).

- Quel est l'effet d'une variation de Ki sur la stabilité, la rapidité et la précision à partir de :
  a. La réponse indicielle (en terme de D %, tr, t_{pic}, …).
  b. Le tracé de Bode.
- Conclure.

4- Correcteur à avance de phase : $C(s) = K_p(\frac{K_p + aK_d s}{K_p + K_d s})$

Nous fixons Kp = 5 et nous varions la valeur de $\mathbf{K_d}$ (K_d=0.1, 1, 2 et 5).

- Quel est l'effet d'une variation de K_d sur la stabilité, la rapidité et la précision à partir de :
  a. La réponse indicielle (en termes de D %, tr, t_{pic}, …).
  b. Le tracé de Bode.
- Conclure.

5- Correcteur Proportionnel, Intégral et Dérivé (PID) : $C(s) = K_p + \frac{K_i}{s} + K_d s$

1. Nous fixons Kd = 10, Ki= 30 et nous varions la valeur de $\mathbf{K_p}$(K_p= 20, 40 et 80).

Quel est l'effet d'une variation de Kp sur la réponse indicielle ?

a- Nous fixons Kd = 10, Kp= 20 et nous varions la valeur de $\mathbf{K_i}$(K_i= 20, 30 et 40).

Quel est l'effet d'une variation de Ki sur la réponse indicielle ?

b- Nous fixons Kp = 20, Ki= 30 et nous varions la valeur de $\mathbf{K_d}$(K_d= 1, 2 et 10).

Quel est l'effet d'une variation de Kd sur la réponse indicielle ?

c- Quel correcteur vous parait le plus adapté. Conclure.

TP 5 : Commande des systèmes par retour d'état dans l'espace d'état

1- Objectif :

L'objectif de ce TP a est de présenter Les principes et propriétés de commande par retour d'état de systèmes linéaires continus. Les concepts d'une commande par retour d'état sont étudiés en simulation à travers de l'environnement logiciel MATLAB/SIMULINK.

2- Représentation d'état d'un système continu

En générale, un système linéaire continu mono variable décrit par la représentation d'état est donné sous la forme suivante :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$$

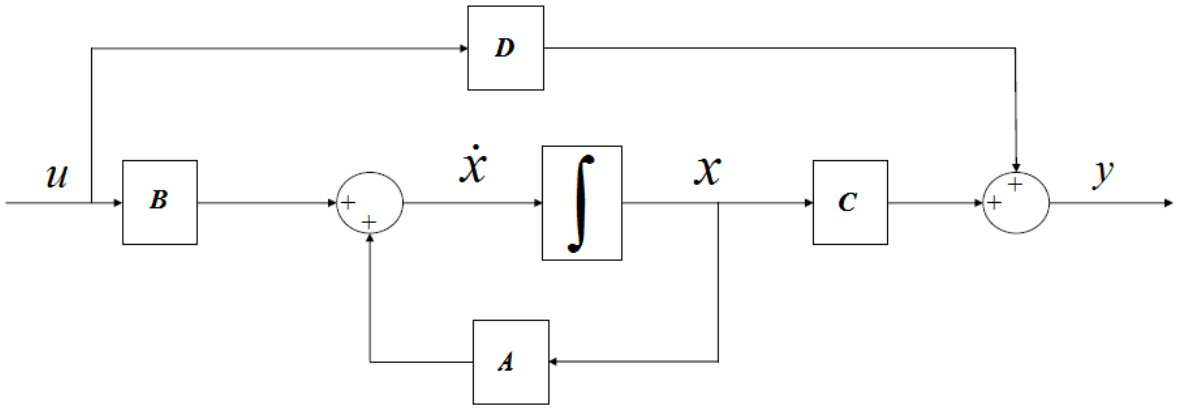
Avec $x(t)$ est le vecteur d'état, $u(t)$ est le vecteur d'entrée (ou de commande) et $y(t)$ représente le vecteur de sortie (ou d'observation). La première équation s'appelle l'équation de commande ; la seconde, équation de sortie ou d'observation.

A est la matrice d'état,

B est la matrice d'entrée,

C est la matrice de sortie,

D est le transfert direct (ou couplage) entrée/sortie.



Représentation schématique d'une modélisation d'état.

Afin de définir les matrices ***A, B, C*** et ***D*** sous Matlab, on déclare alors que le système de nom « sys » admet cette représentation d'état avec la commande « ss » (i.e. state-space)

```
>> sys= ss([4 1 6; 0 1 2; 1 1 -2],[1;0;1],[1 0 0], 4)
a =                          c =
     x1  x2  x3                   x1  x2  x3
 x1   4   1   6               y1   1   0   0
 x2   0   1   2
 x3   1   1  -2
b =                          d=
     u1                           u1
 x1   1                       y1   4
 x2   0
```

Remarque : Il est important de noter, que la représentation d'état d'un système n'est pas Unique ; et dépend du choix des variables d'état que nous opérons. Dans ce cas, les deux modèles suivants :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \quad \text{et} \quad \begin{cases} \dot{z}(t) = \tilde{A}z(t) + \tilde{B}u(t) \\ y(t) = \tilde{C}z(t) + \tilde{D}u(t) \end{cases}$$

Sont équivalents s'il existe un changement de variable régulier entre eux, i.e. s'il existe une matrice inversible ***T*** telle que :

$$z = Tx, \quad \tilde{A} = TAT^{-1}, \quad \tilde{B} = TB, \quad \tilde{C} = CT^{-1}, \quad \tilde{D} = D.$$

3- Commandabilité et observabilité :

3.1- Commandabilité

Avant de synthétiser un système de commande, il est important de savoir si le système est commandable ou pas. Un système est commandable si et seulement s'il est possible de déterminer un signal de commande u(t) sur $[t_1, t_2]$ conduisant tout état $x(t_1) = x_1$ vers l'état voulu $x(t_2) = x_2$ en un temps fini *t1 ≤ t ≤ t2*.

Formellement, un système est complètement commandable si et seulement si la matrice de commandabilité, défini par :

$$C_{AB} = \begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix}$$

est de rang ***n***.

Pour vérifier la commandabilité sous Matlab, on calcule la matrice de commandabilité C_{AB} par la commande « ctrb » et puis le rang de cette matrice.

```
>> Cr=ctrb(A,B)
Cr =
   1   10   36
   0    2    0
   1   -1   14
>> rank(Cr)
ans =
   3
```

3.2- Observabilité :

Un système est dit observable à un instant t_1, si la connaissance du signal d'entrée et du signal de sortie sur un intervalle de temps $[t_1, t_2]$ permet de calculer l'état du système à l'instant t_1.

Un système est observable si et seulement si la matrice d'observabilité définie par :

$$O_{CA} = \begin{bmatrix} C & CA & CA^2 & \dots & CA^{n-1} \end{bmatrix}^T$$

est de rang ***n***.

Pour vérifier l'observabilité sous Matlab, on calcule la matrice d'observabilité O_{CA} par la commande « obsv » et puis le rang de cette matrice.

```
>> Ob=obsv(A,C)
Ob=
      1    0    0
      4    1    6
     22   11   14
>> rank(Ob)
ans =
   3
```

4- Passage de la représentation d'état à la fonction de transfert

L'application de la transformée de Laplace à la représentation d'état d'un système mono-variable donne :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \Rightarrow \begin{cases} sx(s) - x(0) = Ax(s) + Bu(s) \\ y(s) = Cx(s) + Du(s) \end{cases}$$

soit :

$$sx(s) - x(0) = Ax(s) + Bu(s) \Rightarrow (sI - A)x(s) = Ax(s) + Bu(s)$$

$$\Rightarrow (sI - A)x(s) = Ax(s) + Bu(s)$$

$$\Rightarrow x(s) = (sI - A)^{-1}x(0) + (sI - A)^{-1}Bu(s)$$

on a donc :

$$y(s) = Cx(s) + Du(s) = C(sI - A)^{-1}x(0) + \left[(sI - A)^{-1}B + D\right]u(s)$$

La fonction de transfert du système est donc :

$$G(s) = \frac{y(s)}{u(s)} = C(sI - A)^{-1}B + D$$

L'inverse d'une matrice carrée étant égale à sa matrice adjointe divisée par son déterminant, nous pouvons en déduire que les pôles de la fonction de transfert sont les valeurs de ***s*** qui sont solutions de l'équation :

$$det(sI - A)^{-1} = 0$$

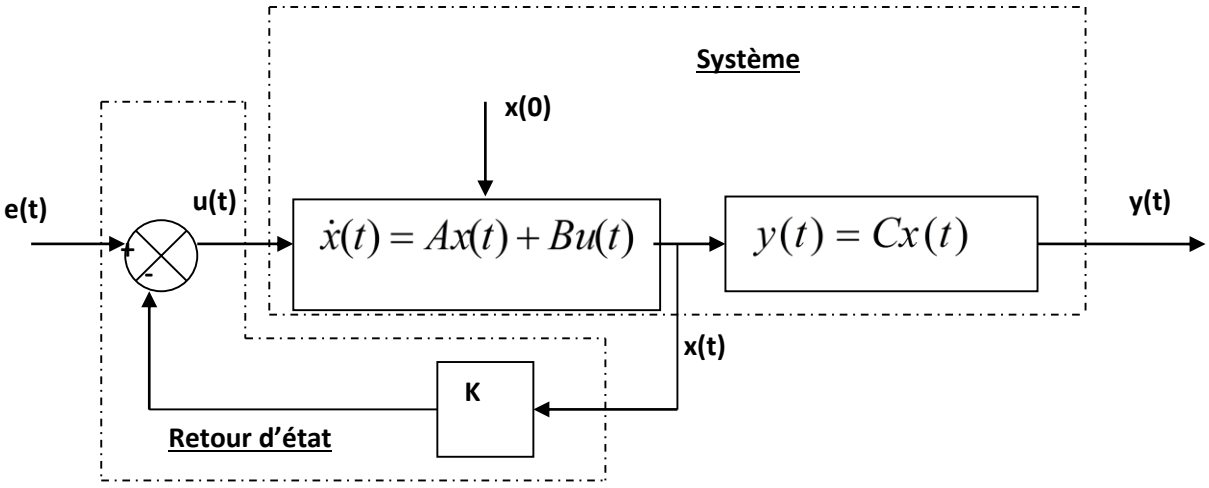
Ce sont donc les valeurs propres de la matrice ***A***.

Afin de passer d'une représentation d'état à la fonction de transfert à l'aide de Matlab, on utilise la commande « ss2tf ».

```
>> [num,den]= ss2tf(A,B,C,D)
num =
   4.0000  -11.0000  -49.0000   72.0000
den =
   1.0000   -3.0000  -14.0000   20.0000
>> G=tf(num,den)
 Transfer function:
4 s^3 - 11 s^2 - 49 s + 72
--------------------------
 s^3 - 3 s^2 - 14 s + 20
```

5- Commande par retour d'état- placement de pôles :

La commande par retour d'état consiste à utiliser le vecteur d'état en contre réaction pour améliorer le comportement propre du processus. La méthode employée est une méthode de placement de pôles c'est-à-dire que l'on doit déterminer une commande telle que les pôles du système en boucle fermée soient correctement placés dans le plan complexe et satisfassent des spécifications d'amortissement, de rapidité... Les pôles de la fonction de transfert étant les valeurs propres de la matrice d'état, le but est donc de réaliser un asservissement modifiant convenablement la matrice d'évolution du système.



On utilise l'état x(t) du système pour construire le signal de commande u(t) par un bouclage de la forme :

$$u(t) = e(t) - Kx(t) = e(t) - \begin{pmatrix} k_1 & k_2 & \dots & k_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

L'objectif de la synthèse de la commande par retour d'état consiste à déterminer le vecteur ligne K (vecteur de gain) de façon à satisfaire des spécifications qui reposent sur un placement des valeurs propres en boucle fermée (performances dynamiques) ou sur un placement de

structure (fonction de transfert en boucle fermée ou équation d'état). **Cette commande par retour d'état impose que le système (A, B) est commandable.**

6- Fonction de transfert en boucle fermée à partir de la représentation d'état:

Le système en boucle ouverte est régi par :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$$

On a alors :

$$\dot{x}(t) = Ax(t) + B\big(e(t) - Kx(t)\big) \Rightarrow \dot{x}(t) = \big[A - BK\big]x(t) + Be(t)$$

Passage aux transformées de Laplace :

$$X(s) = \big[sI - A + BK\big]^{-1} BE(s),$$

D'où la fonction de transfert en boucle fermée :

$$F(s) = \frac{Y(s)}{E(s)} = C\big[sI - A + BK\big]^{-1} B.$$

Dans le cas d'un placement de pôles pour un système commandable, on identifie le dénominateur de la fonction de transfert en boucle fermée tenant compte du retour d'état avec le dénominateur de la fonction de transfert obtenue à partir du cahier de charges.

7- Travail demandé :

Etude théorique :

Soit un système dynamique décrit par ses équations d'état :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$$

où :

$$A = \begin{bmatrix} -1 & -5 \\ 2 & -8 \end{bmatrix}, \quad B = \begin{bmatrix} 4 \\ -1 \end{bmatrix} \quad \text{et} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

- Etudier la commandabilité et l'observabilité de ce système.
- Calculer la fonction de transfert ***G(s)*** à partir de la représentation d'état et donner sa réponse indicielle.
- Vérifier que le dénominateur de la fonction de transfert ***G(s)*** correspond au polynôme caractéristique de de la matrice d'état ***A.***

- Vérifier que les pôles du système correspondent aux valeurs propres de la matrice d'état ***A.***
- Peut-on réaliser une commande par retour d'état, justifier votre réponse.

On souhaite placer ce système dans une boucle à retour d'état avec un vecteur de gain ***K***.

- Calculer le vecteur de gain ***K*** pour que le système, en boucle fermée et soumis à un échelon unitaire de consigne, soit caractérisé par une marge de phase de ***45***◦ et par un temps de montée $t_m = 0{,}4$ ***s.***

Etude pratique (étude sous Matlab) :

On considère un système dynamique décrit par sa représentation d'état:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$$

avec :

$$A = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 2 & 0 \\ -1 & 1 & 1 \end{bmatrix},\ B = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \text{ et } C = [1 \quad 0 \quad 0]$$

Dans un fichier script, écrire le programme qui permet de :

1- Ecrire le système sous la forme d'état.
2- Vérifier la commandabilité et l'observabilité du système.
3- Calculer la fonction de transfert G(s).
   - Quelle est l'ordre du système ?
   - Calculer et représenter dans le plan complexe les pôles et les zéros du système
   - Visualiser la réponse indicielle.
   - Conclure sur la stabilité.
4- Vérifier que le dénominateur de la fonction de transfert ***G(s)*** correspond au polynôme caractéristique de de la matrice d'état ***A.***
5- Vérifier que les pôles du système correspondent aux valeurs propres de la matrice d'état ***A.***

On désire placer ce système dans une boucle à retour d'état avec un vecteur de gain K de manière à imposer les pôles ***[-1 -2 -3]*** en boucle fermée.

6- Calculer le vecteur de gain ***K*** en utilisant la commande ***« place ».***
7- Calculer la fonction de transfert en boucle fermée ***H(s)*** à partir de la représentation d'état.
   - Vérifier la stabilité du système bouclé.
   - Conclure.
8- Vérifier que le dénominateur de la fonction de transfert ***H(s)*** correspond au polynôme caractéristique de de la matrice (***A–BK).***

9- Vérifier que les pôles du système correspondent aux valeurs propres de la matrice (***A-BK).***

A l'aide de Simulink, visualiser en même temps sur un oscilloscope :

- La réponse indicielle en boucle ouverte (sans la commande de retour d'état).
- La réponse indicielle en boucle fermée (avec la commande de retour d'état).