



UNIVERSITE DE SOUSSE

Institut Supérieure des Sciences Appliquées et de Technologie de
Sousse

Département de Génie Electronique

Fasicule de Travaux Pratiques

Atelier d'Automatique

(2^{ème} année LA EEA)

Elaboré par :

M^r. Kolsi Sami

e-mail : sami_kolsi@yahoo.fr

ISSAT SOUSSE

Cité Ettafela Ibn Khaldoun, 4003, Sousse-Tunisie

Sommaire

1- TP1 : Initiation à Matlab

2- TP2 : Analyse temporelle et fréquentielle des
systèmes linéaires du premier ordre

3- TP3 : Analyse temporelle et fréquentielle des
systèmes linéaires du deuxième ordre

4- TP4 : Performance des systèmes asservis : Etude de
la précision et correction des systèmes asservis
linéaire

5- TP5 : Etude de Systèmes échantillonnés

TP1 : Initiation à Matlab

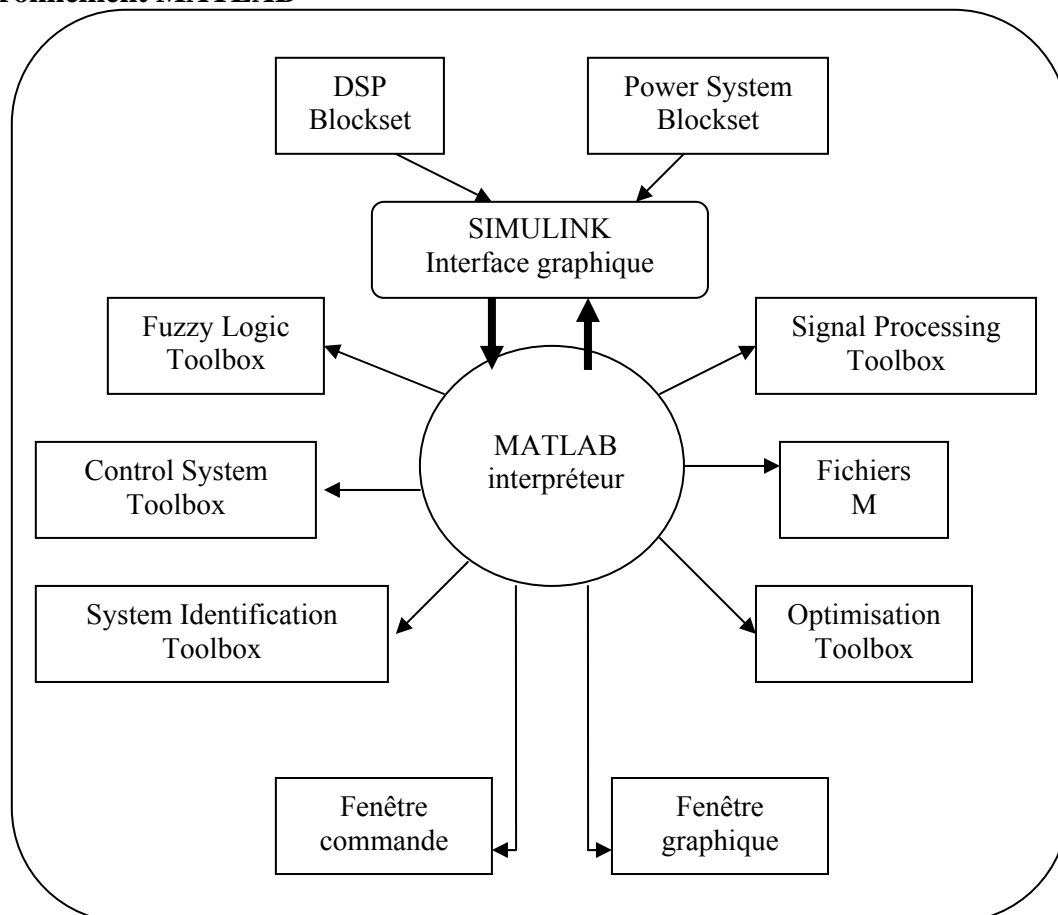
Introduction

MATLAB (MATrix LABoratory) est un logiciel de calcul numérique produit par MathWorks. Il consiste en un langage interprété qui s'exécute dans une fenêtre dite d'*exécution*.

L'intérêt de **MATLAB** tient, d'une part, à sa simplicité d'utilisation : pas de compilation, déclaration implicite des variables utilisées et, d'autre part, à sa richesse fonctionnelle : arithmétique matricielle et nombreuses fonctions de haut niveau dans divers domaines (analyse numérique, statistique, commande optimale, représentation graphique, ...). Il est à noter que toutes les commandes sont en anglais et l'aide en ligne également.

On peut utiliser **MATLAB**, soit en mode *en ligne*, c'est-à-dire saisir des commandes dans la fenêtre d'exécution au fur et à mesure, soit en mode *programmation*, en écrivant dans des fichiers séparés (*.m) l'enchaînement des commandes. Ces fichiers s'appellent des *scripts* et on les construit à l'aide de n'importe quel éditeur de texte. Le mode en ligne permet d'obtenir des résultats simples qui ne sont pas sauvegardés. Le mode programmation, quant à lui, permet de développer des applications très complexes.

Environnement MATLAB



Fenêtre Commande: Dans cette fenêtre, l'utilisateur donne les instructions et MATLAB retourne les résultats.

Fenêtres Graphique: MATLAB trace les graphiques dans ces fenêtres.

Fichiers M: Ce sont des programmes en langage MATLAB (écrits par l'utilisateur).

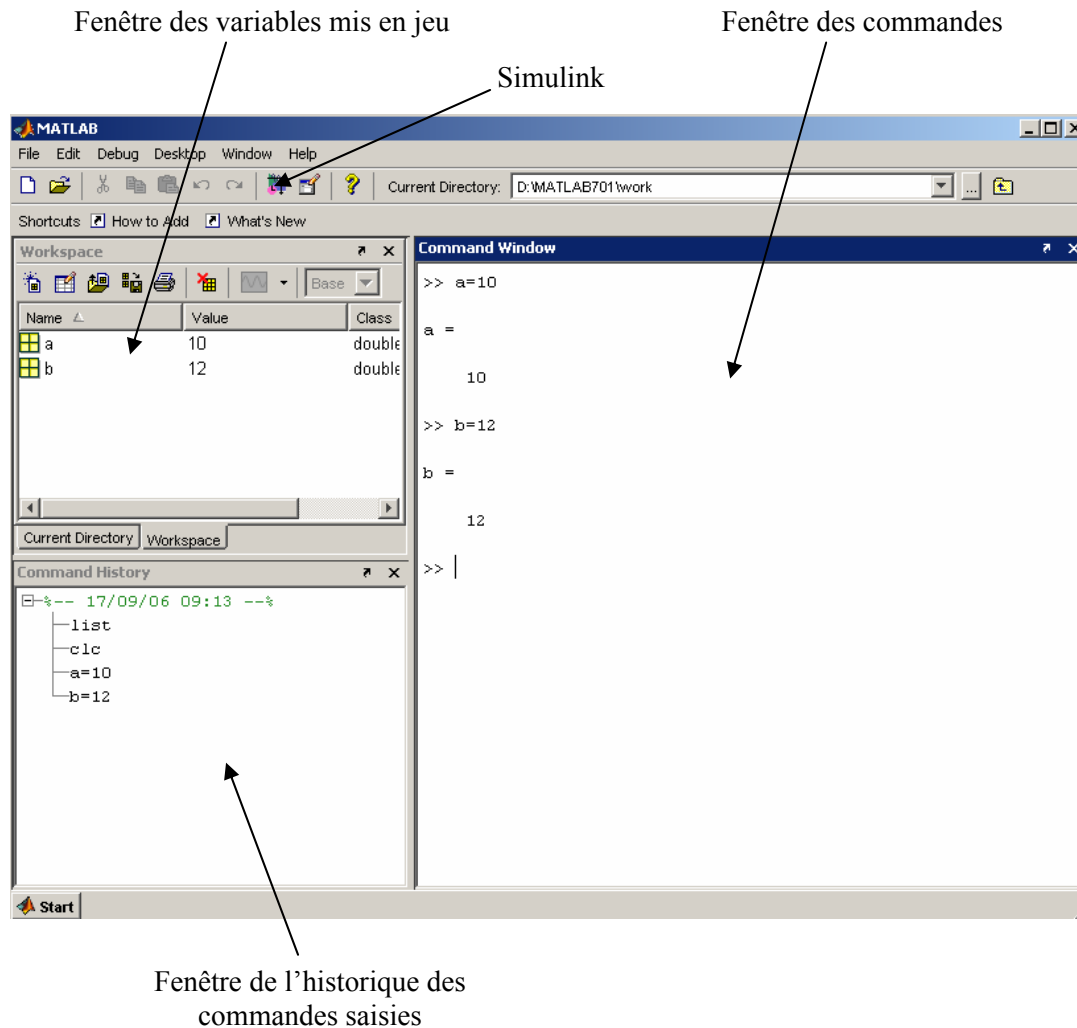
Toolboxes: Ce sont des collections de fichiers M développés pour des domaines d'application spécifiques (Signal Processing Toolbox, System Identification Toolbox, Control System Toolbox, u-

Synthesis and Analysis Toolbox, Robust Control Toolbox, Optimization Toolbox, Neural Network Toolbox, Spline Toolbox, Chemometrics Toolbox, Fuzzy Logic Toolbox, etc.)

Simulink: C'est l'extension graphique de MATLAB permettant la modélisation et la simulation de systèmes dynamiques en utilisant une représentation de type graphique (schéma bloc).

Blocksets: Ce sont des collections de blocs Simulink développés pour des domaines d'application spécifiques (DSP Blockset, Power System Blockset, etc.).

Lancement de MATLAB



Il existe 2 principales façons d'utiliser Matlab :

– En ligne de commande : l'utilisateur tape des commandes dans la fenêtre de commande ("command Window" pour **Matlab 6.5** sous Windows). Celles-ci sont directement exécutées par Matlab. Une commande se termine par un retour chariot ("entrée"). Il est possible d'exécuter plusieurs commandes successivement en les écrivant sur la même ligne et en les séparant par le caractère ";" ; ". Lorsqu'une commande se termine par le caractère ";" ; " elle est muette : le résultat n'est pas affiché à l'écran. Le caractère % indique des commentaires. Tout ce qui se trouve après ce caractère sur la même ligne n'est pas interprété par Matlab.

Exemples :

```
>> a = 2 % Affectation de 2 à la variable a  
a =
```

```
2
>> b=1; % Affectation de 1 à la variable b sans affichage
>> a+b
ans =
3
>> c = a + b; d = c + 1;
>> d % permet d'afficher le contenu de la variable d
d =
4
```

Les variables créées sont stockées dans l'espace de travail (workspace) et peuvent être réutilisées à tout moment.

En ligne de commande, les commandes tapées précédemment peuvent être rappelées en utilisant les flèches vers le haut et vers le bas.

– Par l'intermédiaire de scripts ou de fonctions. Les commandes sont écrites dans un fichier texte de la même façon qu'en ligne de commande. Le fichier doit être sauvegardé avec une extension ".m" pour signifier qu'il s'agit d'un script Matlab. Pour exécuter ce script il suffit de taper son nom (sans l'extension ".m") dans la fenêtre de commande. Toutes les commandes du script sont alors exécutées successivement. Attention, pour que les fichiers ".m" puissent être lancés depuis Matlab vous devez vous placer dans le répertoire où se trouve le fichier.

Exemples :

```
Fichier exemple.m *****
a = 2 % Affectation de 2 à la variable a
b=1; % Affectation de 1 à la variable b sans affichage
a+b
c = a + b; d = c + 1;
d % permet d'afficher le contenu de la variable d
*****
>> exemple
a =
2
ans =
3
d =
4
```

Les variables créées dans un script sont stockées dans le workspace lors de l'exécution du script. Elles peuvent donc ensuite être utilisées en ligne de commande. Matlab dispose d'une aide en ligne permettant d'obtenir les détails d'utilisation d'une commande : paramètres d'entrée, arguments de sortie, etc. Par exemple, pour obtenir des informations sur la commande d'inversion de matrice "inv", il suffit de taper

```
>> help inv
```

MATLAB conserve l'historique des commandes saisies de façon interactive lors d'une session. Il est donc possible de récupérer des instructions déjà saisies et de les modifier dans le but de les réutiliser.

Des touches utiles sont utilisées pour cet objectif :

- ↑ passer à l'instruction précédente.
- ↓ passer à l'instruction suivante.
- ← aller vers la gauche sur la ligne de commande.
- aller vers la droite sur la ligne de commande.

N.B : MATLAB fait la distinction entre minuscule et majuscule !!.

Quelques commandes utiles :

clc efface toutes les commandes affichées dans la fenêtre des commandes

clear <nom_variable> efface la variable portant le nom « nom_variable » du workspace

clear all efface tous les objets de la mémoire

help general donne l'aide sur les commandes de MATLAB à usage général.

help elfun cette commande affine l'aide sur les fonction mathématiques élémentaires

help <nom_fonction> donne l'aide sur la fonction « nom_fonction »

Remarques :

>> commande

résultat affichage du résultat

>> commande ; le point virgule provoque l'absence de l'affichage du résultat.

Les variables sous MATLAB

MATLAB gère les nombres entiers, réels, complexes, les chaînes de caractères ainsi que les tableaux de nombres de façon transparente. Il n'est pas utile de déclarer le type de la variable que l'on manipule, y compris les tableaux. Par ailleurs, toutes les variables utilisées restent présentes en mémoire et peuvent être rappelées. Ainsi les instructions suivantes, déclarent les variables lors de leur affectation :

```
>> a=1
```

```
a =
```

```
1
```

```
>> b=1.01
```

```
b =
```

```
1.0100
```

```
>> X=1.0e+05
```

```
X =
```

```
100000
```

```
>> nom=' mon nom'
```

```
nom =
```

```
mon nom
```

```
>> c=1+2i
```

```
c =
```

```
1.0000 + 2.0000i
```

On déclare un vecteur colonne de la façon suivante :

```
>> u=[1;3;-1]
```

```
u =
```

```
1
```

```
3
```

```
-1
```

Un vecteur ligne de la façon suivante :

```
>> v=[1,3,-1]
```

```
v =
```

```
1 3 -1
```

et une matrice d'ordre 3x2 :

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

```
A =
```

```
1 2
```

```
-1 3
```

```
4 0
```

La ',' sert à séparer les éléments d'une ligne et ';' les éléments colonnes. En fait, on peut remplacer la ',' par un espace pour améliorer la lisibilité.

Un polynôme P, exemple : $P=3*x^3+4*x-7$ de la façon suivante :

```
>> P=[3 0 4 -7]
```

```
P=
```

```
3 4 0 -7
```

Les constantes prédéfinies :

pi= 3.14159265358979

eps= 2.2204e-016 (la distance entre 1.0 et le flottant le plus proche. Cette valeur peut être modifiée.

Inf (Infinite) : nombre infini.

NaN (Not a Number) : n'est pas un nombre, exprime parfois une indétermination.

Variable prédéfinie :

ans : variable contenant la dernière réponse.

5. Fonctions de la "Control toolbox"

Pour utiliser ces fonctions, il suffit de les taper soit dans la fenêtre de commande, soit dans un script.

Voici la plupart des fonctions qui vous seront utiles en TP.

– créer une fonction de transfert : ex : $G(s) = s+2 / s^2+3s$

```
>> G = tf([1 2], [1 3 0])
```

– créer une fonction de transfert numérique : ex : $Gz(z) = z+0.5 / z-0.2$

```
>> Gz = tf([1 0.5], [1 -0.2], Te)
```

où T_e est la valeur de la période d'échantillonnage.

– créer une fonction de transfert par les pôles, zéros et gain : ex : $G(s) = 5(s+2) / s^2+3s = 5(s+2) / (s+3)s$

```
>> G = zpk([-2], [0 -3], 5)
```

– mettre une fonction de transfert G sous forme de pôles et zéros

```
>> zpk(G)
```

– Transposer une fonction de transfert continue en numérique

```
>> Gz = c2d(G, Te, 'method')
```

où T_e est la période d'échantillonnage et method est la méthode de transposition choisie (zoh, tustin, prewarp ou matched)

– Tracer la réponse indicielle d'un système donne par sa fonction de transfert G

```
>> step(G)
```

– Obtenir la carte des pôles et des zéros d'un système

```
>> pzmap(G)
```

– Diagramme de Bode :

```
>> bode(G) %trace les diagrammes de Bode de G
```

```
>> bode(G, {w_min, w_max}) % trace les diagrammes entre les pulsations w_min et w_max  
%par exemple
```

```
>> bode(G, {0.1, 100})
```

%trace les diagrammes de Bode entre 0.1 et 100 rad/s

```
>> [gain, phase, w] = bode(G) % Ne trace pas les courbes, mais rend trois tableaux :
```

%le gain sans dimension (pas en décibels) et la phase (en degrés) déterminés aux pulsations w

–Diagramme de Nyquist :

```
>> Nyquist(G)
```

Trace le diagramme de Nyquist. Attention, les éventuels cercles à l'infini ne sont pas représentés. Il faut donc analyser la fonction de transfert pour pouvoir déterminer avec certitude la stabilité d'une boucle fermée.

– Lieu des racines :

```
>> rltool(G)
```

Trace le lieu des racines de G, avec retour unitaire. Les pôles et zéros du système G sont représentés par des croix et cercles bleus. La position des pôles de la boucle fermée est représentée par des carrés rouges.

En cliquant sur ces carrés, il est possible de les déplacer le long du lieu d'Evans. Le gain affiche en haut de la fenêtre est alors modifié simultanément.

rltool est un utilitaire permettant de régler un correcteur par la méthode du lieu des racines. Une fois le lieu des racines de G tracé, l'utilisateur a de nombreux outils.

Il est ainsi possible d'ajouter des pôles et des zéros sur la carte des pôles et zéros. Ces zéros et pôles sont affectés au correcteur. Ils peuvent être déplacés par des "clique - glisse".

Pour éditer le correcteur, cliquer sur le bloc C rouge (ou K cyan selon la version) dans le schéma bloc représenté en haut à droite.

– file → import : permet d'affecter des fonctions de transfert aux différents éléments du modèle. Ces fonctions de transfert doivent avoir été définies dans le workspace. Attention, les blocs de gain doivent être entrés sous la forme de fonction de transfert (par exemple, pour $H = 5$, il faut entrer $H = \text{tf}([5], [1])$)

– file → export : permet d'exporter un bloc vers le workspace (par exemple le correcteur)

– Analysis → response to step command (ou cliquer sur le bouton "step" en bas à gauche) : trace la réponse indicielle du système en boucle fermée pour le gain donné.

– Clic droit dans la fenêtre rltool (ou tools ! Add grid / boundary) : permet de définir des contraintes pour le système boucle : dépassement, temps d'établissement, etc. Attention : ces contraintes ne sont valables que si le système en BF est équivalent à un système du 2^{ème} ordre.

L'utilisation de Simulink

Simulink est un utilitaire de simulation permettant de représenter les systèmes à partir de schémas bloc.

– Taper la commande Simulink à partir de la fenêtre « Matlab command Windows ».

Une fenêtre est alors affichée sur la partie supérieure de l'écran. Elle contient les différentes **familles de blocs** disponibles dans la bibliothèque Simulink.

L'utilisation de Simulink est assez intuitive. Elle consiste à sélectionner des blocs représentant des fonctions de transfert, des gains, etc. et à les glisser sur le schéma de simulation. Les blocs sont reliés entre eux par des traits orientés tracés à l'aide de la souris.

– file → new → model : crée un nouveau modèle Simulink (*.mdl)

– Les fonctions de transfert continues se trouvent dans le menu "Continuous" ou "Linear"

– Les fonctions de transfert numériques se trouvent dans le menu "Discrete" ou "Discontinuous"

– Les convertisseurs numériques analogiques (BOZ) ou "Zero-Order Hold" sont dans le menu "Discrete" ou "Discontinuous"

– Les gains sont dans le menu "Math Operations" ou "Linear"

– Les comparateurs sont dans le menu "Math operations" ou "Linear"

– Les outils de mesure (scope) sont dans le menu "Sinks"

– Les sources (échelons (step), rampes, etc.) sont dans le menu "Sources"

– Les retards à utiliser sont appelés "Transport Delay" et se trouvent dans le menu "non linear" ou "Continuous"

– Les saturations sont dans le menu "Discontinuities" ou "non linear".

– Les quantificateurs "Quantizer" sont dans le menu "Discontinuities" ou "non linear".

Remarques

– Simulink détermine la nature des signaux en fonction de la nature des blocs dans lesquels ils entrent. Il est nécessaire pour les blocs de nature "numérique" de spécifier la période d'échantillonnage en double cliquant dessus. Attention, Matlab n'est pas dérangé par des périodes d'échantillonnage différentes selon les blocs et ne vous avertira pas en cas d'erreur.

– Par défaut, les échelons démarrent à l'instant $t = 1$ s. Il est parfois utile de modifier en $t = 0$.

Une fois les blocs placés sur le schéma, vous pouvez les modifier en double cliquant dessus. Par exemple, en cliquant sur un bloc fonction de transfert vous pouvez ajouter des pôles et des zéros, modifier les gains, etc.

Il est possible d'utiliser dans ces blocs des variables définies dans le workspace. Par exemple, si vous avez défini :

```
>> K = 5
```

alors en mettant K dans un bloc Simulink, sa valeur sera 5

La simulation est ensuite lancée par simulation → Start. Les résultats de la simulation peuvent être obtenus en double cliquant sur les scopes. Une fenêtre s'ouvre alors avec les traces des signaux mesures.

Les paramètres de la simulation peuvent être modifiés en faisant Simulation → Simulation parameters. On peut notamment modifier la durée de la simulation. Lorsque les courbes des "Scopes" sont tronquées, il suffit d'ouvrir le bloc "Scope" et d'augmenter le paramètre "Data history" de l'onglet "Settings".

– Si on utilise le « **To workspace** », on doit le nommer, une fois la simulation est terminée, on tape la commande **plot(name)** à partir de la fenêtre « Matlab command Windows ».

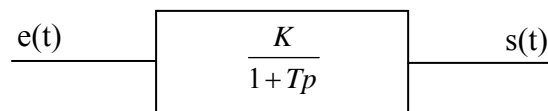
name : le nom du « **To workspace** ».

TP2 : Analyse temporelle et fréquentielle des systèmes linéaires du premier ordre

Objectifs :

- Se familiariser avec les techniques d'identification pratiques des performances d'un système dynamique par l'étude de sa réponse temporelle et fréquentielle.
- Observer le comportement temporel et fréquentiel d'un système dynamique du premier ordre.

On considère le système du premier ordre suivant :



A. Réponse indicielle

- 1) a- Donner l'expression de la réponse indicielle $s(t)$ face à un échelon unitaire.
b- Que représente K et T pour le système.
c- Tracer l'allure de cette courbe (K et T sont quelconques).
d- Donner l'expression du temps de réponse à 5% (T_r).
- 2) a- Pour $T=1s$ et $K=0,5$ et $1,5$ enregistrer la réponse indicielle et mesurer le temps de réponse à 5%. Quel est l'effet de la valeur de K sur le temps de réponse T_r . Conclure.
b- Pour $K=1$ et $T=0,5s$ et $1s$ enregistrer la réponse indicielle et mesurer le temps de réponse à 5%. Quel est l'effet de la valeur de T sur le temps de réponse T_r . Conclure.
- 3) Que peut-on dire de la stabilité du système du premier ordre? Argumenter votre réponse.

B. Réponse à une rampe

- 1) Pour $T=1s$ et $K=0,5$ et $1,5$, enregistrer la réponse à une rampe unitaire. A partir de la réponse déduire la valeur de la constante du temps T .
- 2) Pour quelle valeur de K a-t-on une réponse parallèle à la rampe.

C. Réponse fréquentielle (harmonique)

- 1) Donner l'expression du gain complexe $|G(j\omega)|$ ainsi que celle de $G_{db}(\omega)$. En déduire la pulsation de coupure du système ω_c .
- 2) Donner l'expression de la phase $\phi(j\omega)$.

On prend $T=0,2$ et $K=1$.

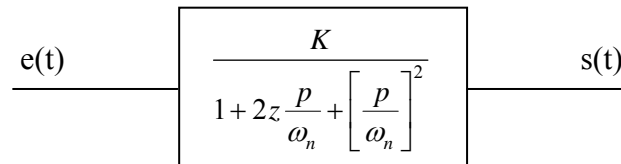
- 3) Tracer alors les allures des courbes dans le diagramme de Bode, Nyquist et Black. Retrouver les valeurs de T et K.
- 4) Tracer avec Matlab le lieu des racines de G.
- 5) Quel est l'effet d'une variation de la valeur du gain K (T=cte) sur :
 - Le gain G.
 - La phase ϕ .
 - La pulsation de coupure.
- 6) Quel est l'effet d'une variation de la valeur de T (K=cte) sur :
 - Le gain G.
 - La phase ϕ .
 - La pulsation de coupure.

TP3 : Analyse temporelle et fréquentielle des systèmes linéaires du deuxième ordre

Objectifs :

- Se familiariser avec les techniques d'identification pratiques des performances d'un système dynamique par l'étude de sa réponse temporelle et fréquentielle.
- Observer le comportement temporel et fréquentiel d'un système dynamique du deuxième ordre.

On considère le système du deuxième ordre suivant :



K : gain statique

ω_n : pulsation propre non amortie

A. Réponse indicielle

- 1) Donner l'expression de la réponse indicielle $s(t)$ face à un échelon unitaire. Discuter suivant les valeurs de z .
- 2) Pour $z < 0,7$, donner l'expression du :
 - a- Temps de montée (T_m).
 - b- Le temps de réponse à 5% (T_r).
 - c- Le temps du pic (T_{pic}).
 - d- Le dépassement D en %.
 - e- La pseudo-pulsation ω_p .

On prend $K=1$ et $\omega_n=10\text{rd/s}$.

- 3) Visualiser sur une même figure la réponse indicielle de ce système pour $z=0,3$; $0,5$; $0,7$; 1 ; $1,3$. Retrouver les paramètres de la question 2 pour chaque cas. Conclure.
- 4) Quel est l'effet, pour la réponse indicielle d'une variation de la valeur de K (z et ω_n sont constantes) sur les paramètres de la question 2. Conclure.
- 5) Quel est l'effet, pour la réponse indicielle d'une variation de la valeur de ω_n (z et K sont constantes) sur les paramètres de la question 2. Conclure.

B. Réponse à une rampe

Tracer l'allure de $s(t)$ pour différentes valeurs de z (prendre les valeurs de la question A. 3.).

C. Réponse fréquentielle

- 1) Donner l'expression du gain complexe $|G(j\omega)|$ ainsi que celle de $G_{db}(\omega)$.
- 2) Donner l'expression de la phase $\varphi(j\omega)$, de la pulsation de résonance ω_R et du facteur de qualité Q du système.

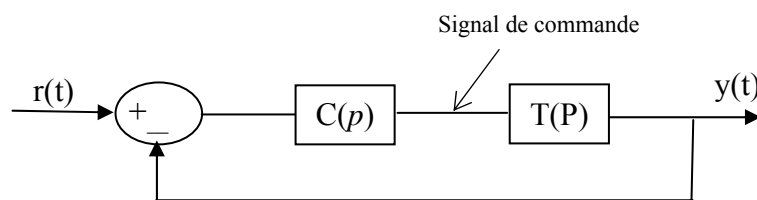
$K=1$ et $\omega_n=10\text{rd/s}$

- 3) Tracer alors les allures des courbes dans le diagramme de Bode. Prendre les valeurs de z de la question 3. Conclure.
- 4) Tracer avec Matlab le lieu des racines de G
- 5) Quel est l'effet d'une variation de la valeur du gain K sur :
 - Le gain G
 - La phase
 - La pulsation de résonance.

TP4 : Performance des systèmes asservis Etude de la précision et correction des systèmes asservis linéaires

I. But du TP

On veut déterminer le correcteur $C(p)$ qui permet d'obtenir la réponse attendue du système $y(t)$ en matière de précision et de stabilité par rapport à la consigne $r(t)$. Celui-ci est inséré dans la chaîne d'action comme suite :



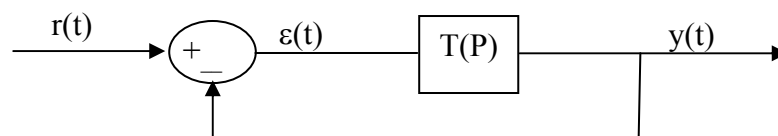
où $T(p)$ est la fonction de transfert du système à commander.

II. Simulations Numériques

Soit le système linéaire d'ordre 2 définis par sa fonction de transfert en boucle ouverte comme suite :

$$T(p) = \frac{25}{p^2 + 2p + 25}$$

- 1- Déterminer le coefficient d'amortissement ξ , la pulsation propre w_n et le gain statique K_S .
- 2- On considère le système en boucle fermée suivant :



2-1- Calculer l'erreur statique de position ($t \rightarrow \infty$) définie par

$$\lim_{t \rightarrow \infty} \varepsilon(t) = \lim_{p \rightarrow 0} p \varepsilon(p)$$

et ceci pour un signal de référence continu $r(t) = 20 u(t)$, $u(t)$ est le signal échelon unitaire, et visualiser la réponse $y(t)$ obtenue et déduire le dépassement.

2-2- Tracer en boucle ouverte les diagrammes de Bode et de Black du système $T(p)$ et donner la marge de gain et de phase. Interpréter et conclure.

A. Correcteur Proportionnel P

1- Pour un correcteur proportionnel $C(p) = K$, donner le schéma bloc équivalent en boucle fermée et étudier la stabilité du système en fonction de K par le critère algébrique de Routh.

2- Calculer K pour obtenir une erreur statique de position égale à 10%.

3- Pour cette valeur de K , visualiser la réponse $y(t)$ pour un signal de référence échelon $r(t) = 20 u(t)$ et déterminer le dépassement.

4- En boucle ouverte, tracer les diagrammes de Bode et de Black et donner la marge de gain et de phase. Interpréter et conclure.

B. Correcteur à avance de phase

Le correcteur PD est généralement substitué par un correcteur à avance de phase de fonction de transfert :

$$C(p) = K \frac{1 + a \tau p}{1 + \tau p}, \quad a > 1$$

Sachant que ce correcteur est caractérisé par :

- une phase maximale φ_{\max} définie par : $\varphi_{\max} = \text{artg} \left(\frac{a-1}{2\sqrt{a}} \right)$
- la pulsation w_m qui correspond à cette phase maximale est égale à $w_m = \frac{1}{\tau\sqrt{a}}$

1- Déterminer a et τ pour φ_{\max} et w_m choisis. Par suite, calculer l'erreur statique de position, visualiser la réponse $y(t)$ pour la consigne $r(t) = 20 u(t)$ et déduire le dépassement.

2- Pour le système corrigé obtenu, tracer les diagrammes de Bode et de Black en boucle ouverte et donner la marge de gain et de phase. Commenter les performances du système asservi.

C. Correcteur Proportionnel Intégral Dérivé PID :

On monte en cascade un correcteur PD et un correcteur PI , L'ensemble est un correcteur PID :



Le correcteur PI est défini par la fonction de transfert suivante :

$$C(p) = \frac{K_p (p + \tau_i)}{p}$$

Pour K_p et τ_i choisis, on demande de calculer l'erreur statique de position, de visualiser la réponse $y(t)$ pour $r(t) = 20 u(t)$ et d'en déduire le dépassement. En boucle ouverte, tracer les diagrammes de Bode et de Black et donner la marge de gain et de phase. Interpréter et conclure.

TP5 : Etude des Systèmes échantillonnés

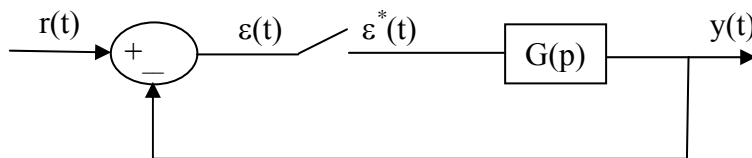
I. But de la manipulation

Etude de la réponse et de la stabilité des systèmes échantillonnés en fonction de la période d'échantillonnage.

II. Simulations Numériques

Partie 1

Soit le système asservi échantillonné donné par le schéma bloc suivant :



La transmittance $G(p)$ est :

$$G(p) = \frac{0,1}{(p + 0,1)}$$

1- Pour ce système asservi, on demande de :

1-1- placer un Bloqueur d'Ordre Zéro (BOZ) et mentionner son rôle, tout en sachant qu'il est défini par la fonction de transfert suivante :

$$\text{BOZ}(p) = \frac{1 - e^{-T}p}{p}$$

1-2- donner la fonction de transfert échantillonnée $H(z)$ en boucle fermée.

2- Pour les deux périodes d'échantillonnages $T_1 = 2s$ et $T_2 = 0,5s$ et une entrée en échelon de 15, on demande de :

2-1- tracer la réponse en boucle fermée du système échantillonné de deux manières possibles :

2-1-1- avec $G(p)$ et BOZ

2-1-2- avec $G(z)$

2-2- donner le nombre d'échantillons par seconde.

2-3- comparer les sorties obtenues par rapport aux réponses en cas continu

3- Interpréter et conclure

Partie 2

On considère un moteur caractérisé par sa fonction transfert suivante :

$$G(p) = \frac{\Omega(p)}{U(p)} = \frac{K}{(1 + \tau p)}$$

où $\Omega(p)$ est la transformée de Laplace de la vitesse de rotation, $U(p)$ est la transformée de la place de la tension du moteur, τ est la constante de temps mécanique du système (on néglige la constante de temps électrique).

$K = 30 \text{ rad/s.V}$ et que $\tau = 10 \text{ ms}$.

1- Pour une période d'échantillonnage T_e choisie dans l'intervalle $\frac{\tau}{4} < T_e < \tau$:

- 1-1- donner la transformée en Z du système obtenu en boucle ouverte avec le BOZ
- 1-2- donner la fonction de transfert $H(z)$ de ce système en boucle fermée en considérant un retour unitaire.
- 1-3- étudier la stabilité du système asservi échantillonné

2- Si l'on effectuait un retour unitaire en continu on demande de :

- 2-1- donner la fonction de transfert $H(p)$ en boucle fermée
- 2-2- quelle serait la nouvelle constante de temps obtenue τ' ?
- 2-3- la période d'échantillonnage choisie précédemment est-elle toujours adaptée ? Sinon, proposer une nouvelle période d'échantillonnage
- 2-4- calculer la nouvelle fonction de transfert $G(z)$ échantillonnée en boucle ouverte, puis $H(z)$ en boucle fermée.
- 2-5- Calculer l'erreur statique du système
- 2-6- avec le premier et le deuxième choix du pas d'échantillonnage, comparer la réponse du système échantillonné avec celle en continu pour une consigne en échelon de 10 rad/s

3- Interpréter et conclure