

MET2

# **Entrainement avec servomoteur DC**

Régulation de courant

---

**Douglas Houmard, Rahma Rejeb**

18.11.2013

Professeur : M.Girardin

Assistant : C.Simon

# Table des matières

<b>1 INTRODUCTION .....</b>	<b>4</b>
<b>2 ETUDE THÉORIQUE DU SYSTÈME À RÉGLER .....</b>	<b>4</b>
2.1 CALCUL DE L'INERTIE DE LA CHARGE MÉCANIQUE .....	4
2.2 FONCTION DE TRANSFERT THÉORIQUE .....	6
2.2.1 Schéma fonctionnel régulation de courant.....	6
2.2.2 Fonction du transfert du moteur et de la charge.....	6
2.2.3 Fonction du transfert du variateur .....	6
2.2.4 Fonction de transfert du capteur de courant.....	7
2.2.5 Fonction de transfert théorique du système à régler .....	7
<b>3 VALIDATION DU MODEL PAR IDENTIFICATION NON PARAMÉTRIQUE .....</b>	<b>8</b>
3.1 BUT.....	8
3.2 PROGRAMME POUR FAIRE CET ESSAI .....	8
3.3 DÉROULEMENT DE L'ESSAI .....	8
3.4 IDENTIFICATION .....	8
3.5 COMPARAISON ENTRE LE SYSTÈME À RÉGLER THÉORIQUE ET CELUI OBTENU PAR SYSQUAKE .....	11
<b>4 VALIDATION DU MODÈLE PAR IDENTIFICATION PARAMÉTRIQUE.....</b>	<b>12</b>
4.1 BUT.....	12
4.2 PROGRAMME POUR FAIRE CET ESSAI .....	12
4.3 DÉROULEMENT DE L'ESSAI .....	12
4.4 IDENTIFICATION .....	12
4.4.1 Utilisation de IDENT.....	12
4.4.2 Matlab.....	15
4.4.3 Traitement de la fonction de transfert .....	15
4.5 COMPARAISON DU SYSTÈME À RÉGLER THÉORIQUE, DEPUIS SYSQUAKE ET DEPUIS IDENT.....	17
<b>5 SYNTHÈSE DU RÉGULATEUR PI .....</b>	<b>17</b>
5.1 RAISONS DU CHOIX DU RÉGULATEUR PI .....	17
5.2 FONCTION DE TRANSFERT DU RÉGULATEUR PI.....	17
5.3 MÉTHODE DE LA COMPENSATION DU PÔLE DOMINANT .....	18
5.4 MÉTHODE DES PETITES CONSTANTES DE TEMPS.....	18
5.5 RÉGULATEUR 1 : MARGE DE PHASE DE 60° ET COMPENSATION DU PÔLE DOMINANT.....	18
5.6 RÉGULATEUR 2 : MARGE DE PHASE DE 60° ET COMPENSATION DES PETITES CONSTANTES DE TEMPS .....	19
5.7 RÉGULATEUR 3 : MARGE DE PHASE DE 45° ET COMPENSATION DU PÔLE DOMINANT.....	20
5.8 RÉGULATEUR 4 : MARGE DE PHASE DE 45° ET COMPENSATION DES PETITES CONSTANTES DE TEMPS .....	21
<b>6 IMPLÉMENTATION DU RÉGULATEUR PI EN C.....</b>	<b>22</b>
6.1 LOI DE COMMANDE DU RÉGULATEUR .....	22
6.2 IMPLÉMENTATION EN C.....	23
6.3 ANTI WINDUP.....	23
6.4 PROGRAMME COMPLET.....	23
<b>7 TEST DU RÉGULATEUR DE COURANT .....</b>	<b>23</b>
7.1 PROCÉDURE .....	23
7.2 RÉGULATEUR 1 .....	25
7.3 RÉGULATEUR 2 .....	25
7.4 RÉGULATEUR 3 .....	26
7.5 RÉGULATEUR 4 .....	26
7.6 ANTI WINDUP.....	27
7.7 TEST À ROTOR BLOQUÉ.....	28
<b>8 CONCLUSION .....</b>	<b>28</b>
<b>9 ANNEXES .....</b>	<b>30</b>

9.1 CODE MATLAB .....	30
9.2 IMPLÉMENTATION DU RÉGULATEUR PI .....	32

## Liste des figures

FIGURE 1 : CHARGE ENTRAÎNÉE PAR LE MOTEUR À ÉTUDIER .....	4
FIGURE 2 : SCHÉMA FONCTIONNEL DU VARIATEUR DE COURANT .....	6
FIGURE 3 : DIAGRAMME DE BODE DU SYSTÈME À RÉGLER .....	7
FIGURE 4 : DIAGRAMME DE BODE DU MOTEUR MESURE ET THÉORIE .....	9
FIGURE 5 : DIAGRAMME DE BODE APRÈS L'AJUSTEMENT .....	10
FIGURE 6: COMPARAISON DES BODE ENTRE LA THÉORIE ET SYSQUAKE .....	11
FIGURE 7: FENÊTRE DE COMMANDE DE RTP WATCH .....	12
FIGURE 8: CHARGEMENT DES DONNÉES AVEC IDENT.....	13
FIGURE 9: FENÊTRE PRINCIPALE DE IDENT.....	13
FIGURE 10: CONFIGURATION POUR L'ESTIMATION DES COURBES .....	14
FIGURE 11: CONFIGURATION DU NOMBRE DE PÔLE ET DE ZÉROS .....	14
FIGURE 12: RÉSULTATS DES AJUSTEMENTS.....	15
FIGURE 13: POURCENTAGE DES AJUSTEMENTS PAR RAPPORT À LA COURBE MESURÉE.....	15
FIGURE 14: COMPARAISON DES TROIS SYSTÈMES À RÉGLER .....	17
FIGURE 15: DIAGRAMME DE BODE AVEC LE RÉGULATEUR 1 AVEC $K_{PI}=1$ .....	18
FIGURE 16: DIAGRAMME DE BODE AVEC LE RÉGULATEUR 1 AJUSTÉ.....	19
FIGURE 17: DIAGRAMME DE BODE AVEC LE RÉGULATEUR 2 AJUSTÉ.....	20
FIGURE 18: DIAGRAMME DE BODE AVEC LE RÉGULATEUR 3 AVEC $K_{PI}=1$ .....	20
FIGURE 19: DIAGRAMME DE BODE AVEC LE RÉGULATEUR 3 AJUSTÉ.....	21
FIGURE 20: DIAGRAMME DE BODE AVEC LE RÉGULATEUR 2 AJUSTÉ.....	22
FIGURE 21: FENÊTRE DE COMMANDE POUR LE TEST DES RÉGULATEURS.....	24
FIGURE 22: LISTE DES VARIABLES DANS RTP WATCH .....	24
FIGURE 23 : COMPARAISON DE LA RÉPONSE INDICIELLE MESURÉE ET SIMULÉE POUR UNE MARGE DE $60^\circ$ (COMPENSATION DU PÔLE DOMINANT) .....	25
FIGURE 24 : COMPARAISON DE LA RÉPONSE INDICIELLE MESURÉE ET SIMULÉE POUR UNE MARGE DE $60^\circ$ (LA COMPENSATION DES PETITES CONSTANTES DE TEMPS).....	25
FIGURE 25 : COMPARAISON DE LA RÉPONSE INDICIELLE MESURÉE ET SIMULÉE POUR UNE MARGE DE $45^\circ$ (COMPENSATION DU PÔLE DOMINANT) .....	26
FIGURE 26 : COMPARAISON DE LA RÉPONSE INDICIELLE MESURÉE ET SIMULÉE POUR UNE MARGE DE $45^\circ$ (LA COMPENSATION DES PETITES CONSTANTES DE TEMPS).....	27
FIGURE 27: ANTI WINDUP .....	27
FIGURE 28: TEST À ROTOR BLOQUÉ .....	28

## **1 Introduction**

La stratégie de commande du servomoteur DC à aimants permanents a été étudiée dans le cadre du cours. L'objet de ce travail est de la tester en pratique, à la manière d'un projet. Dans un premier temps, une modélisation avec MATLAB permettant la simulation de la commande devra être établie sur la base d'une identification du système à régler. Quelques tests types devront être définis et exécutés en simulation.

Dans un second temps, la commande devra être programmée en langage C afin de pouvoir être mise en application et testée sur un convertisseur et un servomoteur DC de type industriel. Les essais effectués en simulation pourront alors être répétés sur le système réel, et les résultats pourront être comparés et commentés.

## **2 Etude théorique du système à régler**

Paramètres du moteur et du convertisseur disponibles

- Moteur : Servomoteur DC MT30U4 Technical Data
- Charge : Bague en laiton
- Variateur : fréquence d'échantillonnage  $f_e = 12 \text{ kHz}$
- Mesure du courant : Filtre passe-bas d'ordre 1 (la fréquence de coupure n'est pas connue)

$$R_a = 0.45 \text{ } [\Omega]$$

$$L_a = 3.3 \cdot 10^{-3} \text{ [H]}$$

$$K_T = 0.34 \text{ [Nm/A]}$$

$$K_E = K_T = 0.34 \text{ [V.s/rad]}$$

### **2.1 Calcul de l'inertie de la charge mécanique**

La charge entraînée par le moteur à étudier est une pièce mécanique trouée en laiton, voir figure ci-dessous :

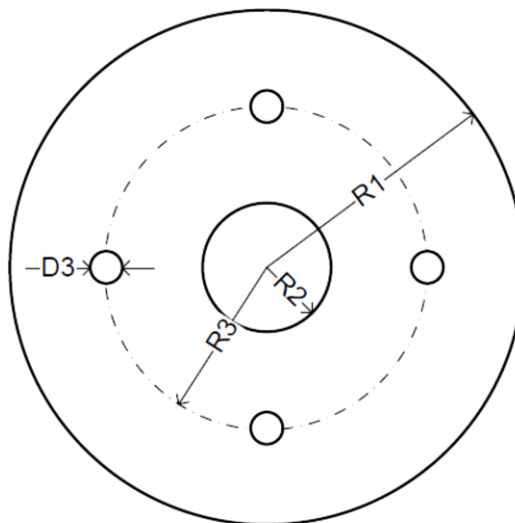


Figure 1 : Charge entraînée par le moteur à étudier

La longueur de cette charge cylindrique est de  $l = 20 \text{ mm}$ .

La densité du laiton est  $\rho = 8500 \text{ kg/m}^3$ .

$$R_1 = 25 \text{ mm}$$

$$R_2 = 8 \text{ mm}$$

$$R_3 = 6 \text{ mm}$$

$$R_4 = 17.5 \text{ mm}$$

L'inertie de la charge est calculée par étapes :

✚ Calcul de l'inertie du grand cylindre

$$J_{\text{cylindre}} = \frac{1}{2} \cdot (m_1 \cdot R_1^2) \text{ et } m_1 = \rho \cdot \pi \cdot R_1^2$$

$$\Rightarrow J_{\text{cylindre}} = \frac{1}{2} \cdot (\rho \cdot \pi \cdot R_1^4) = \frac{1}{2} \cdot 8500 \cdot \pi \cdot (25 \cdot 10^{-3})^2 = 1.0431 \cdot 10^{-4} \text{ kgm}^2$$

✚ Calcul de l'inertie des quatre petits trous

$$J_{\text{trous}} = \frac{1}{2} \cdot (m_2 \cdot R_3^2) + m_2 \cdot R_4^2 \text{ et } m_2 = \rho \cdot \pi \cdot R_3^2$$

$$\Rightarrow J_{\text{trous}} = 2 \cdot (\rho \cdot \pi \cdot R_3^4) + 4 \cdot (\rho \cdot \pi \cdot R_3^2) \cdot R_4^2 = 6.6668 \cdot 10^{-6} \text{ kgm}^2$$

✚ Calcul de l'inertie du trou central

$$J_{\text{troucentrde}} = \frac{1}{2} \cdot (m_2 \cdot R_2^2) \text{ et } m_2 = \rho \cdot \pi \cdot R_2^2$$

$$\Rightarrow J_{\text{troucentrde}} = \frac{1}{2} \cdot (\rho \cdot \pi \cdot R_2^4) = \frac{1}{2} \cdot 8500 \cdot \pi \cdot (4 \cdot 10^{-3})^2 = 1.0938 \cdot 10^{-6} \text{ kgm}^2$$

L'inertie totale de la charge est l'inertie du cylindre complet moins les inerties des 4 petits trous et l'inertie du trou central.

$$J_{\text{ch}} = J_{\text{cylindre}} - J_{\text{troucentrde}} - J_{\text{trous}}$$

$$\Rightarrow J_{\text{ch}} = 9.655 \cdot 10^{-5} \text{ kgm}^2$$

➤ Nous pouvons constater que l'inertie des 4 trous vaut seulement 6.4% de l'inertie de la charge et l'inertie du trou central vaut seulement 1.13%.

On peut maintenant déduire facilement l'inertie totale :

$$J_{\text{tot}} = J_{\text{mot}} + J_{\text{ch}}$$

$$\Rightarrow J_{\text{tot}} = 0.0024 \text{ kgm}^2$$

➤ Nous pouvons constater que l'inertie de la charge vaut 4.022 % de l'inertie totale.

## 2.2 Fonction de transfert théorique

### 2.2.1 Schéma fonctionnel régulation de courant

Voici le schéma fonctionnel régulateur de courant :

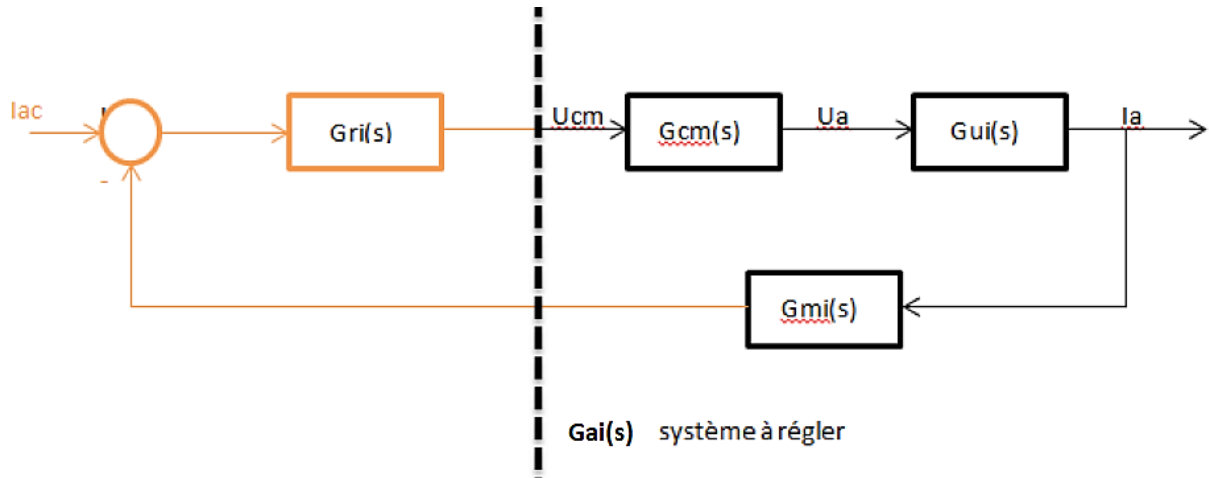


Figure 2 : Schéma fonctionnel du variateur de courant

### 2.2.2 Fonction du transfert du moteur et de la charge

La fonction de transfert du moteur et de la charge est la suivante

$$G_{ui} = \frac{J_{tot}}{K_T \cdot K_E} \cdot \frac{s}{1 + s\tau_m + s^2\tau_m \cdot \tau_e}$$

Calcul des constantes de temps :

$$\tau_e = \frac{L_a}{R_a} = \frac{3.3 \cdot 10^{-3}}{4.5} = 7,3[ms]$$

$$\tau_m = \frac{J_{tot} \cdot R_a}{K_T \cdot K_E} = 9,3[ms]$$

### 2.2.3 Fonction du transfert du variateur

Le variateur est modélisé à l'aide d'une fonction de transfert d'un filtre basse bas d'ordre 1.

$$G_{cm} = \frac{K_{cm}}{1 + s \cdot \tau_{cm}}$$

Avec :

$K_{cm} = 1$  : Gain unitaire

Puisque c'est un variateur numérique la constante de temps est égale à la période d'échantillonnage.

$$\tau_{cm} = \frac{1}{f_e} = 83,33\mu s : \text{Constante de temps du variateur}$$

### 2.2.4 Fonction de transfert du capteur de courant

Le capteur de courant est aussi modélisé à l'aide d'une fonction de transfert d'un filtre basse bande d'ordre 1.

$$G_{mi} = \frac{K_{mi}}{1 + s \cdot \tau_{mi}}$$

Avec :

$$K_{mi} = 1 \text{ et } \tau_{mi} = 0[s]$$

La constante de temps du filtre est inconnue pour le moment donc pour simplifier elle est mise à une valeur nulle.

### 2.2.5 Fonction de transfert théorique du système à régler

La fonction de transfert du système à régler est déterminée à partir des équations précédentes :

$$G_{ai} = G_{cm} \cdot G_{mi} \cdot G_{ui}$$

$$\Rightarrow G_{ai} = K_{ai} \cdot \frac{1}{1 + s \cdot \tau_{cm}} \cdot \frac{s}{1 + s \cdot \tau_m + s^2 \cdot \tau_m \cdot \tau_e} \cdot \frac{1}{1 + s \cdot \tau_{mi}}$$

$$\text{Avec : } K_{ai} = \frac{K_{cm} \cdot K_{mi} \cdot J_{tot}}{K_T \cdot K_E} = 20,73 \cdot 10^{-3}$$

**Diagramme de Bode du système à régler :**

Le diagramme de Bode de notre système à régler est le suivant :

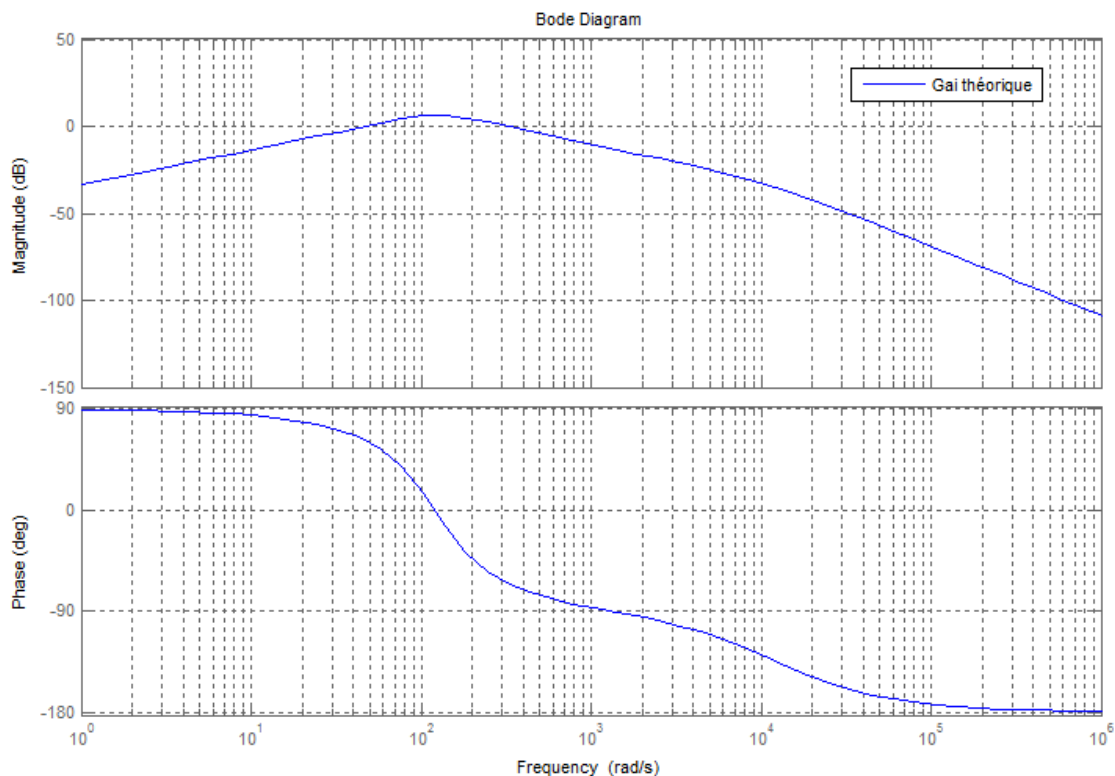


Figure 3 : Diagramme de Bode du système à régler

## **3 Validation du model par identification non paramétrique**

### **3.1 But**

Cette identification est faite avec Sysquake après avoir relevé la caractéristique du moteur avec un bruit blanc. Et pour mesurer la caractéristique il faut d'abord créer un programme adapté.

### **3.2 Programme pour faire cet essai**

La seule chose à faire dans ce programme pour pouvoir faire cet essai est de faire le bout de code suivant :

```
case RTPWATCH_MODE_IDENT_BCL_INTERNE:
{
    tension = consigne;
}
break;
```

Il faut seulement prendre le code C donné et remplacer la partie correspondante dans le code par le bout de code ci-dessus.

### **3.3 Déroulement de l'essai**

Une fois le code C prêt il faut démarrer RTP Watch et faire ce qui est indiqué dans la marche à suivre pour démarrer le variateur et pouvoir faire fonctionner le moteur.

Une fois le variateur prêt il faut lui appliquer un bruit blanc.

### **3.4 Identification**

Une fois les mesures faites il faut utiliser Sysquake pour réaliser l'identification. Il faut commencer par charger la mesure réalisée et mettre entré dans les champs la fonction de transfert théorique du moteur (Gui). La courbe suivante a été obtenue :



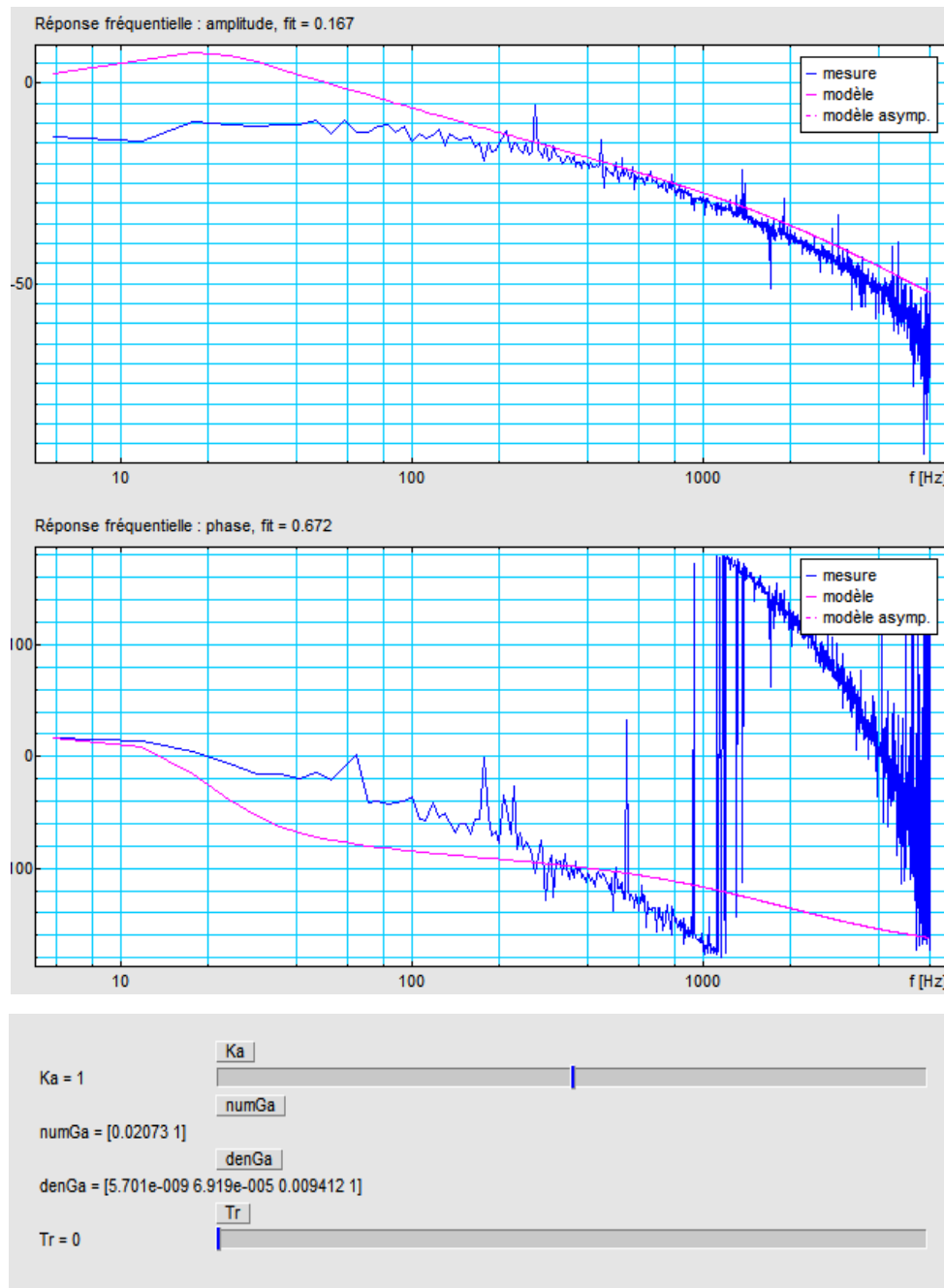


Figure 4 : Diagramme de Bode du moteur mesure et théorie

- Sur chaque graphique, existe 2 courbes, celle en rose correspond à nos calculs théoriques faits précédemment, et celle en bleu correspond à la courbe réelle du moteur.
- Le graphique en haut est le gain du diagramme de bode en boucle ouverte et celui en bas est la phase du diagramme de bode en boucle ouverte.
- On observe un petit décalage entre les deux courbes théorique et mesurée, qu'il faut le minimiser pour être plus précis avec nos calculs.

Comme le modèle ne correspond pas en basse fréquence et que la limite en haute fréquence est de 6 kHz à cause de la fréquence d'échantillonnage du variateur qui est de 12 kHz, et donc pour ne pas avoir de repliement spectral il ne faut pas aller plus haut que  $f_e/2$ . Il ne faut donc utiliser que la partie du modèle en moyenne fréquence.

Ensuite il faut déterminer la constante de temps du filtre passe bas de la mesure de courant en regardant la partie en plus haute fréquence et quand la courbe rose est décalée de trois dB par rapport à la courbe bleu. La pulsation trouvée vaut :

$$\omega_{mi} \cong 1000 \frac{\text{rad}}{\text{s}} \Rightarrow \tau_{mi} = \frac{1}{2 \cdot \pi \cdot \omega_{mi}} = 159.15 \mu\text{s}$$

Puis ensuite il faut rajouter la fonction de transfert du filtre basse bas dans Sysquake. Une fois cette étape faite il faut encore ajuster la courbe de phase pour trouver le temps de retard pur du variateur Tr et ajuster le gain pour que la courbe d'amplitude s'aligne le mieux possible. Voici la courbe obtenu après l'ajustement :

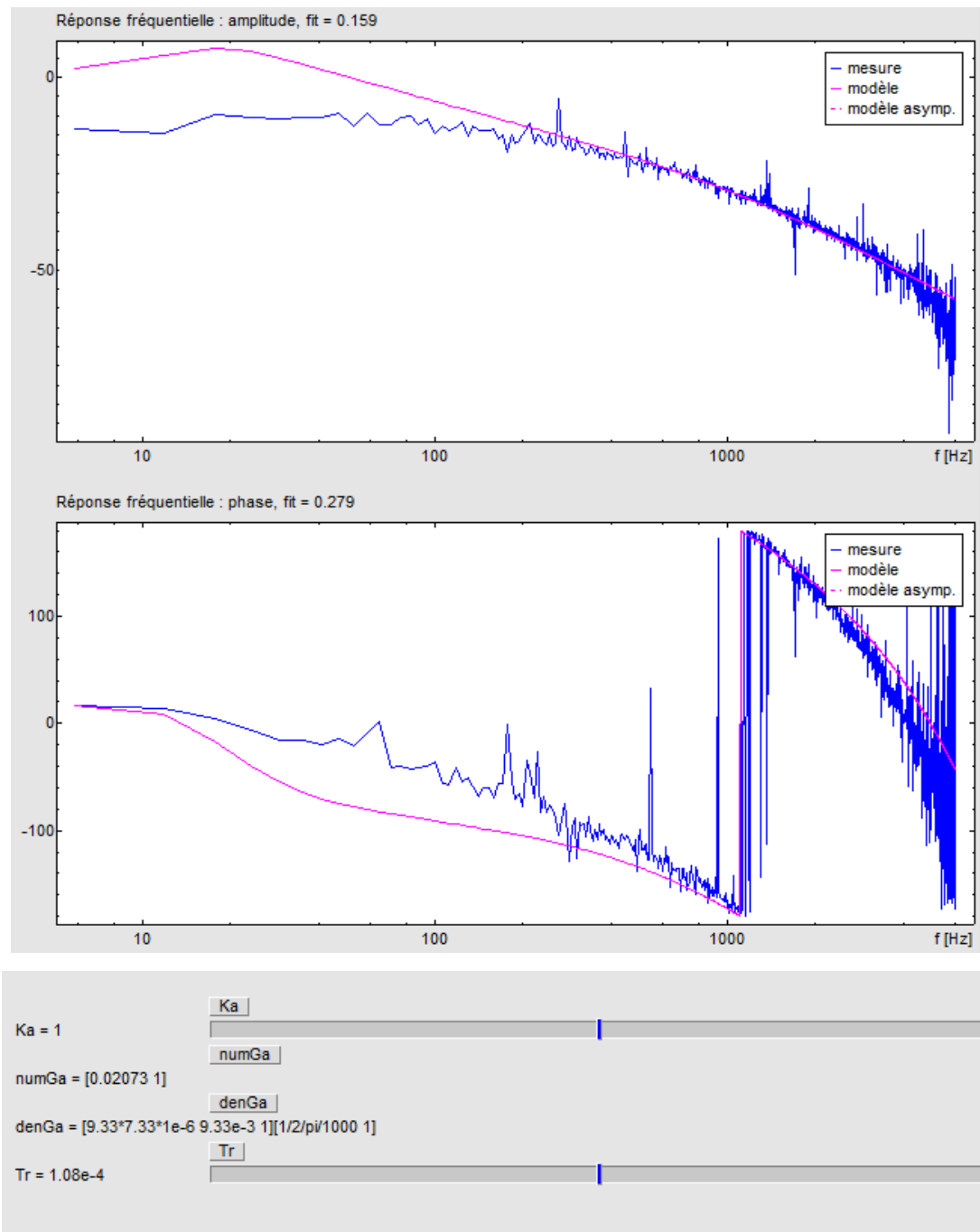


Figure 5 : Diagramme de Bode après l'ajustement

- La courbe théorique est maintenant plus proche de celle mesurée. Nous pouvons donc calculer les nouvelles valeurs.
- ✓ Les valeurs obtenues après l'ajustement sont :  
 $K_{ai} = 1$ , le gain n'a pas changé.  
 $T_r = 108\mu s$

Une fois la valeur du temps de retard trouvé il faut créer la nouvelle fonction du système à régler Gai. Pour ce faire il faut additionner le temps de retard  $T_r$  et la constante de temps du filtre de la mesure du courant et créer un passe bas d'ordre 1, qu'il faut mettre en série avec la fonction de transfert du moteur Gui. La fonction de transfert Gui ne change pas car le gain  $K_{ai}$  trouvé avec Sysquake n'a pas changé.

Donc la constante de temps du filtre passe bas équivalent vaut :

$$\tau_{PB} = \tau_{mi} + T_r = 108 \cdot 10^{-6} + \frac{1}{2 \cdot \pi \cdot 1000} = 267\mu s \text{ et } K_{PB} = 1$$

La fonction de transfert du système à régler sera :

$$G_{ai} = \frac{J_{tot}}{K_T \cdot K_E} \cdot \frac{s}{1 + s\tau_m + s^2\tau_m \cdot \tau_e} \cdot \frac{K_{PB}}{1 + s\tau_{PB}}$$

### 3.5 Comparaison entre le système à régler théorique et celui obtenu par Sysquake

Les deux systèmes à régler sont représentés sur la courbe suivante :

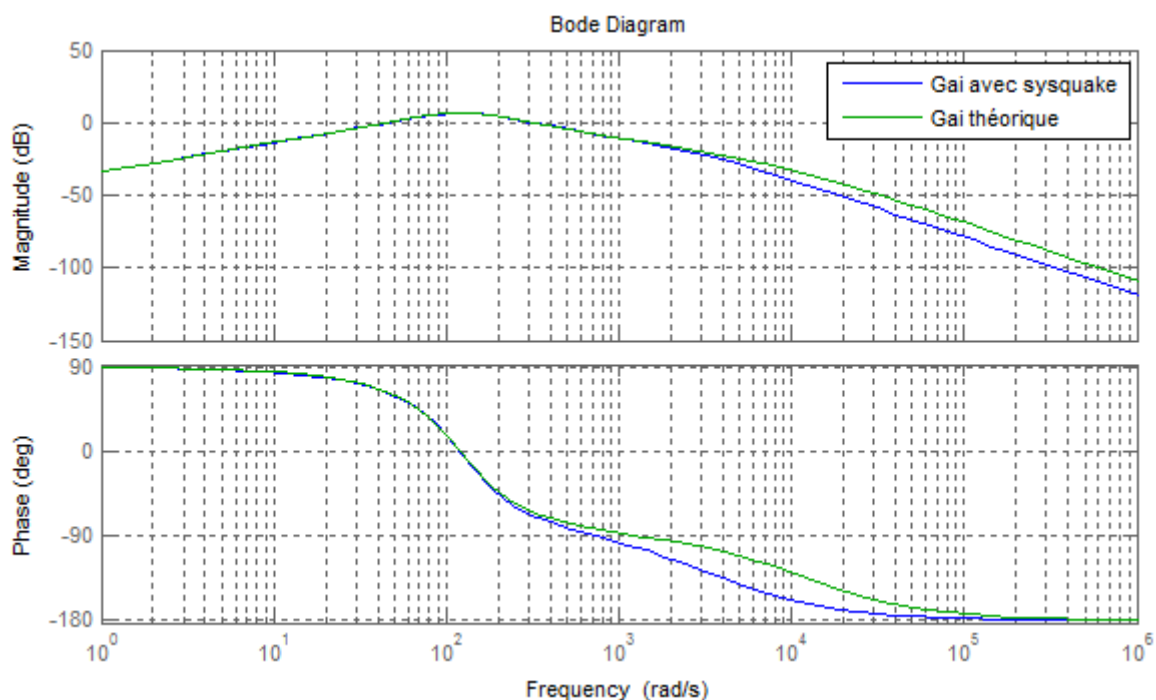


Figure 6: Comparaison des Bode entre la théorie et Sysquake

Il est visible que les deux systèmes à régler commencent à diverger vers 1000 rad/s, mais qu'ils restent assez semblables. Donc dans ce cas la théorie n'est pas beaucoup différente des mesures.

## 4 Validation du modèle par identification paramétrique

### 4.1 But

Le but de cette identification est de retrouver les constantes de temps mécanique et électrique du moteur. Pour ce faire il faut utiliser la fonction IDENT de Matlab.

### 4.2 Programme pour faire cet essai

Le programme pour faire cet essai est le même qu'au point 3.2 à la page 8

### 4.3 Déroulement de l'essai

Une fois le code C prêt il faut démarrer RTP Watch et faire ce qui est indiqué dans la marche à suivre pour démarrer le variateur et pouvoir faire fonctionner le moteur.

Une fois le variateur prêt il faut lui appliquer un double saut de tension comme sur l'image suivant et il faut mettre l'amplitude de la tension de consigne la plus grande possible mais sans qu'il y ait une surcharge de courant. Il faut configurer l'essai comme sur l'image suivante :

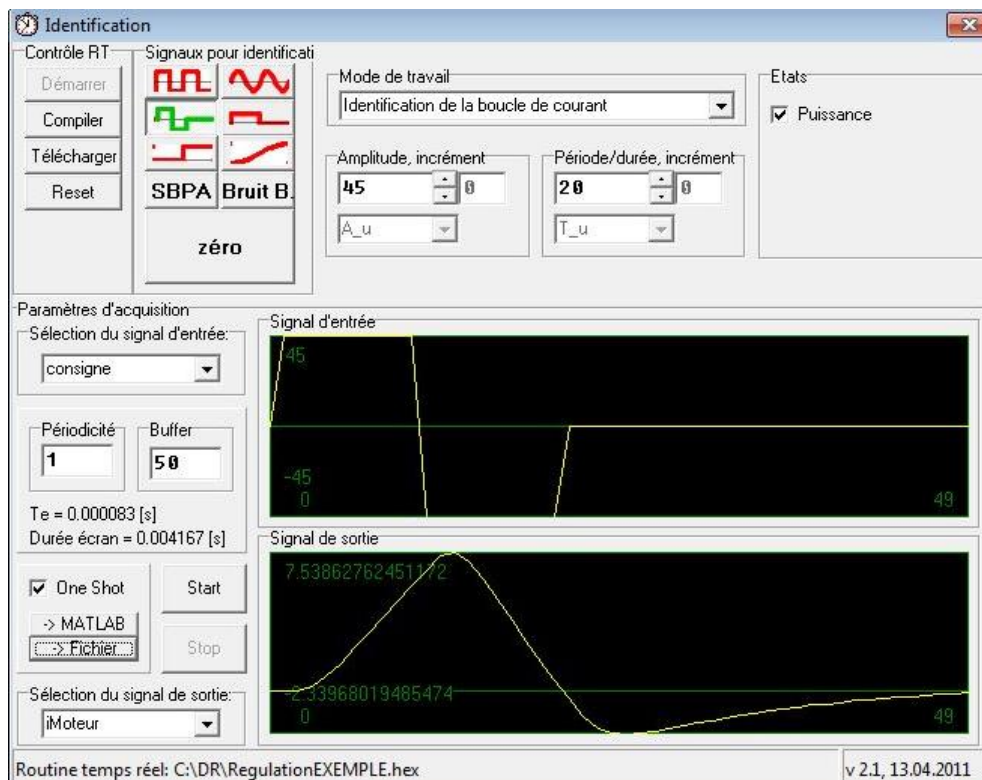


Figure 7: Fenêtre de commande de RTP Watch

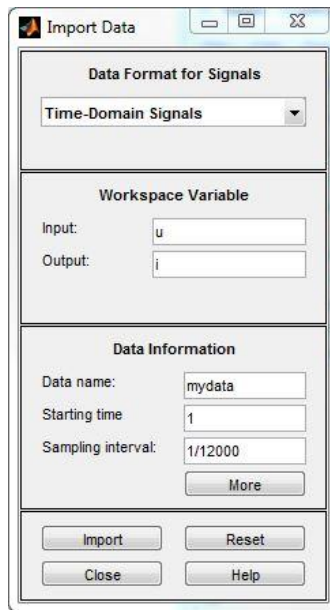
Il faut ensuite enregistrer la mesure dans un fichier et démarrer Matlab ainsi que la fonction IDENT

### 4.4 Indentification

#### 4.4.1 Utilisation de IDENT

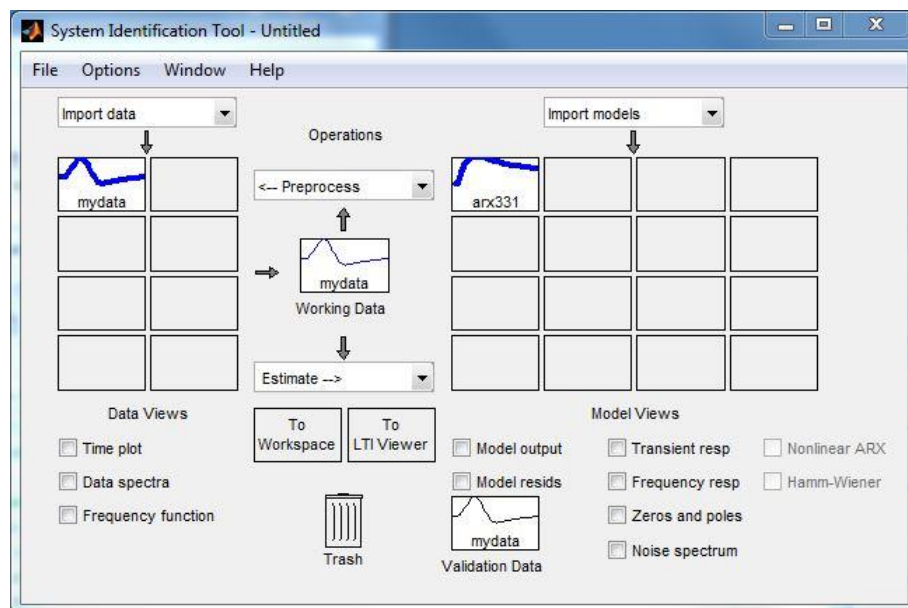
L'identification se déroule de la manière suivante :

Il faut commencer par charger les données enregistrées précédemment comme sur l'image ci-dessous :



**Figure 8: Chargement des données avec IDENT**

Et ensuite il faut configurer IDENT comme suis :



**Figure 9: Fenêtre principale de IDENT**

Puis il faut ouvrir la fenêtre suivante et configurer le nombre de pôles, de zéros et de retards pur et configurer la partie centrale comme dans l'encadrer rouge.

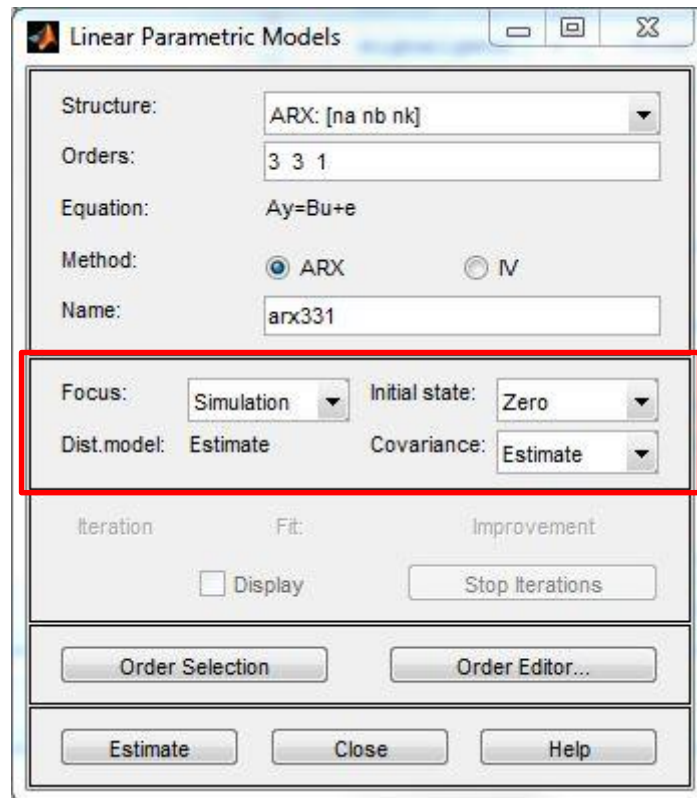


Figure 10: Configuration pour l'estimation des courbes

La configuration du nombre de pôles, de zéros et de retard pur se fait avec la fenêtre suivante :

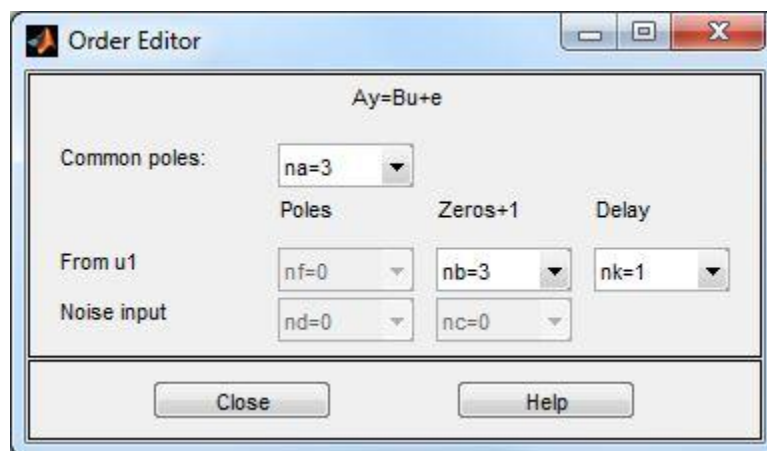


Figure 11: Configuration du nombre de pôle et de zéros

Une fois la configuration faite il faut cliquer sur Close et ensuite cliquer sur Estimate de la Figure 10. Une fois les différentes estimations faites il faut les afficher en cliquant sur Model Output de la Figure 9. Le résultat obtenu est le suivant :

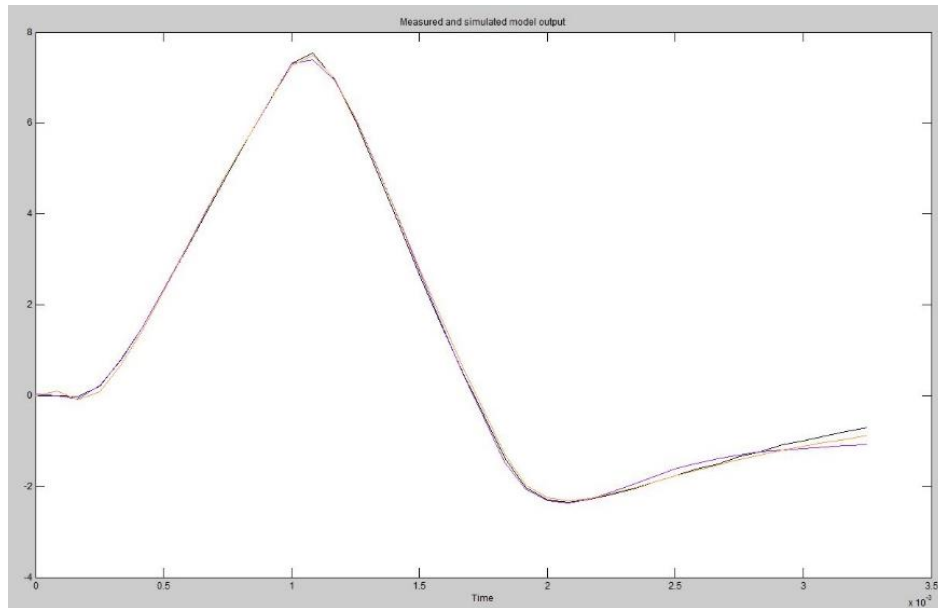


Figure 12: Résultats des ajustements



Figure 13: Pourcentage des ajustements par rapport à la courbe mesurée

Il faut faire varier le nombre de pôles, de zéros et de retard pur pour que la courbe créée soit la plus proche de la courbe mesurée. Mais il ne faut pas mettre trop de pôle, de zéros ou de retard pur. Pour ce cas le nombre maximal de pôle et de zéros sont de quatre chacun et le nombre de retard pur est de trois. Car si un nombre trop grand est choisi les résultats des différents pôles et zéros n'auront plus aucune signification physique. Il ne faut donc pas trop s'éloigner du modèle théorique. Et pour être encore plus proche de la courbe il faut enlever la fin de la courbe car à la fin de la courbe l'influence du frottement sec devient non négligeable et il faut donc l'enlever car les frottements secs n'ont pas été pris en compte dans notre modèle.

Dans notre cas la courbe la plus proche est pour trois pôles, trois zéros et deux retards purs.

Une fois la bonne courbe trouvée il faut l'exporter sous Matlab simplement en glissant le meilleur résultat sur l'icône To Worspace de la Figure 9.

#### 4.4.2 Matlab

Dans Matlab il faut utiliser les trois lignes de code suivantes pour pouvoir passer du mode digital (en Z) au mode continu (Laplace).

```
Gai= d2c(arx332);
[num, den] = tfdata(Gai, 'v')
Gai2 = tf(num, den)
```

Cette dernière opération permet d'avoir la fonction de transfert du système à régler depuis IDENT.

#### 4.4.3 Traitement de la fonction de transfert

Après avoir obtenu la fonction de transfert il faut identifier les différents paramètres. Dans ce cas de figure il faut trouver les trois pôles. Deux des trois pôles sont les pôles du moteur. Et le dernier pôle (en haute fréquence) représente le filtre passe bas combiné entre le variateur et la mesure du courant. Les pôles du moteurs sont les pôles dominant et ils peuvent être complexe conjugués.

Dans notre cas les pôles valent :

$$-7375 \text{ rad/s}$$

$$-919.1 \text{ rad/s}$$

$$-141.2 \text{ rad/s}$$

Ce qui donne les constantes de temps suivantes:

$$136.66 \mu\text{s}$$

$$1.09 \text{ ms}$$

$$7.08 \text{ ms}$$

Ensuite il faut utiliser les deux constantes de temps les plus grandes et en faisant une fonction de transfert du second ordre avec, et faire une identification avec la fonction de transfert Gui pour trouver les constantes de temps mécanique et électrique.

La fonction de transfert d'un système du deuxième ordre est la suivante :

$$G(j\omega) = \frac{1}{1 + \frac{j\omega}{Q_0\omega_0} + \frac{(j\omega)^2}{\omega_0^2}} = \frac{1}{1 + s\tau_m + s^2\tau_m \cdot \tau_e}$$

En créant une fonction de transfert depuis les deux pôles en basse fréquence il vient :

$$\frac{1}{7.706e-006 s^2 + 0.008171 s + 1}$$

Donc il est facile de trouver la constante de temps mécanique qui vaut :

$$\tau_m = 8.171 \text{ ms}$$

Et la constante de temps électrique vaut

$$\tau_e = \frac{7.706 \cdot 10^{-6}}{\tau_m} = \frac{7.706 \cdot 10^{-6}}{8.71 \cdot 10^{-3}} = 884.73 \mu\text{s}$$

La constante de temps mécanique trouvée est assez proche de la constante de temps mécanique théorique. Mais la constante de temps électrique trouvée est beaucoup trop faible par rapport à la constante de temps électrique théorique (environ 8 fois plus faible). Comme cette différence est trop grande, cela veut dire que notre identification avec IDENT n'est pas valable et ne peut donc pas être utilisée.



## 4.5 Comparaison du système à régler théorique, depuis Sysquake et depuis Ident

En affichant les trois systèmes à régler obtenus depuis la théorie, depuis Sysquake et de puis Ident il vient le diagramme de bode suivant :

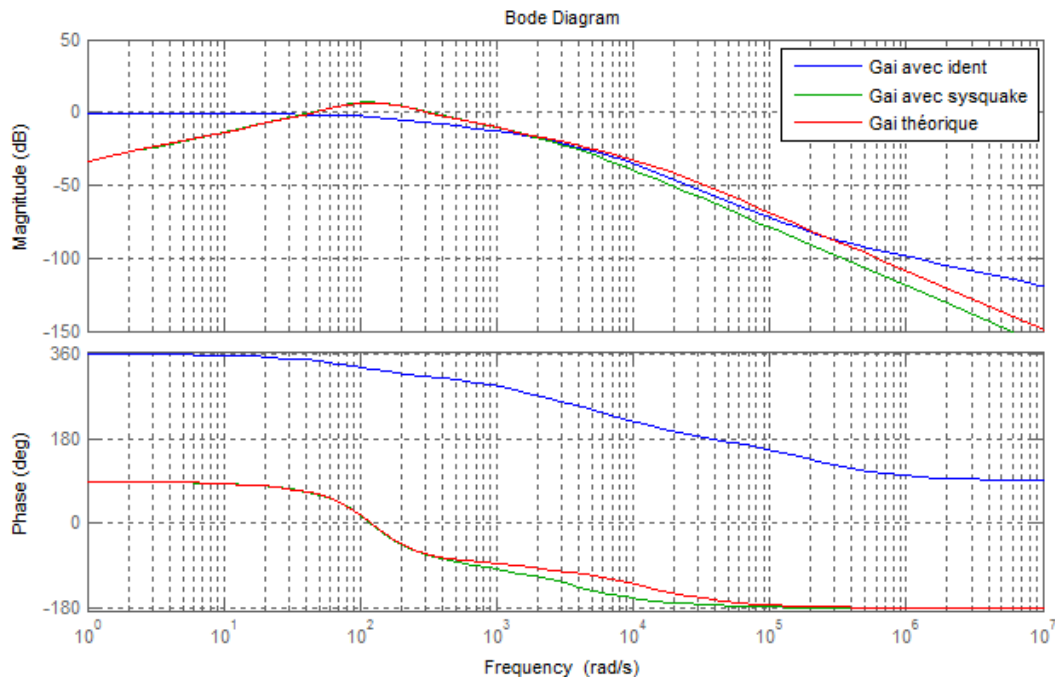


Figure 14: Comparaison des trois systèmes à régler

Il est visible que les systèmes à régler théorique et depuis Sysquake sont semblable, mais que le système à régler depuis Ident est complètement différent et ne correspond pas du tout. Il ne peut donc pas être utilisé. Donc pour le dimensionnement des régulateurs il faudra utiliser le système à régler depuis Sysquake.

## 5 Synthèse du régulateur PI

### 5.1 Raisons du choix du régulateur PI

Les raisons pour lesquelles nous avons choisi d'utiliser un régulateur PI sont :

- ✚ Compenser le comportement dérivateur à basse fréquences.
- ✚ Compenser les couples perturbateurs à moyennes fréquences.
- ✚ Annuler la composante I en hautes fréquences.
- ✚ Ne pas dégrader la bande passante.

### 5.2 Fonction de transfert du régulateur PI

La fonction de transfert d'un régulateur PI est la suivante :

$$G_{pi} = K_{pi} \cdot \frac{1 + s \cdot \tau_{ii}}{s \cdot \tau_{ii}}$$

Pour le dimensionnement de la constante de temps de notre régulateur PI, on a deux méthodes :

- ✚ La méthode de la compensation du pôle dominant.
- ✚ La méthode des petites constantes de temps.

### 5.3 Méthode de la compensation du pôle dominant

Cette méthode consiste à compenser un zéros avec le pôle dominant :

$$\tau_{ii} = \tau_{\max} = \frac{2 \cdot \tau_m \cdot \tau_e}{\tau_m - \sqrt{\tau_m^2 - 4 \cdot \tau_m \cdot \tau_e}} = 8,271[ms]$$

### 5.4 Méthode des petites constantes de temps

Cette méthode consiste à compenser les petites constantes de temps du système à régler. La constante de temps du régulateur est définie par la relation suivante :

$$\tau_{ii} = N \cdot \sum \tau_{pct} = N \cdot \tau_{PB} = N \cdot 267 \cdot 10^{-6} s$$

Où  $N = 8 \dots 50$

Pour trouver la bonne valeur de N il faut essayer différentes valeurs de N et voir où le gain en boucle ouverte (avec le régulateur) est le plus grand. Il faut donc combiner (multiplier ou additionner selon la forme des gains (normal ou dB)) le gain statique du système à régler pour  $K_{pi} = 1$  et la valeur de  $K_{pi}$  pour avoir la marge de phase souhaitée. Et ensuite il faut choisir le N qui donne le plus grand gain en boucle ouverte.

### 5.5 Régulateur 1 : marge de phase de 60° et compensation du pôle dominant

Le diagramme de Bode en boucle ouverte pour  $K_{pi}=1$  sur la figure ci-dessous :

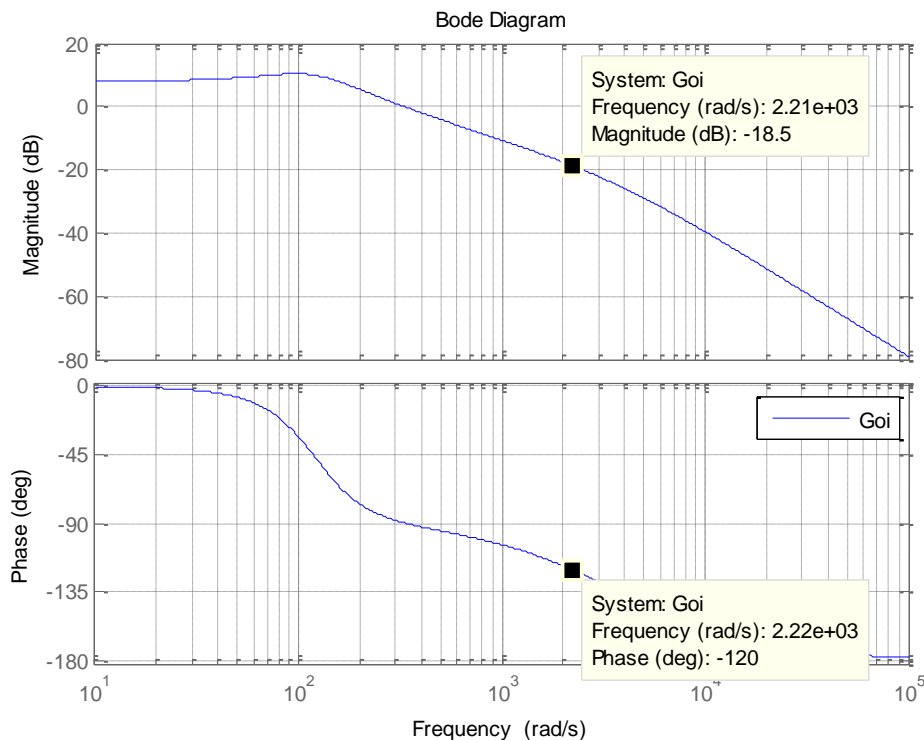


Figure 15: Diagramme de Bode avec le régulateur 1 avec  $K_{pi}=1$

Il est maintenant possible de trouver le gain à ajouter afin d'obtenir la marge de phase de 60°.

Pour avoir une marge de phase de 60° (comme c'est affiché sur le diagramme) il faut un gain  $K_{pi}$  de 18.5 dB soit 8.414.

Le nouveau diagramme de Bode avec la nouvelle valeur de  $K_{pi}$  calculée pour une marge de phase de  $60^\circ$  est sur la courbe ci-dessous.

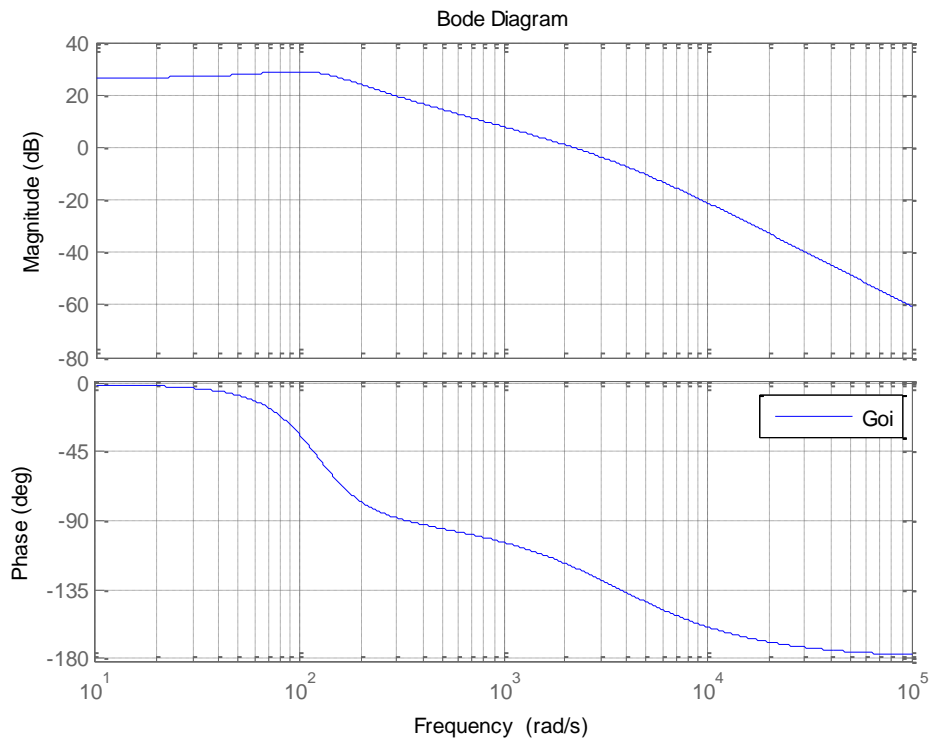


Figure 16: Diagramme de Bode avec le régulateur 1 ajusté

## 5.6 Régulateur 2 : marge de phase de $60^\circ$ et compensation des petites constantes de temps

Pour trouver la valeur de  $N$  idéale il faut trouver le gain en boucle ouverte le plus grand. La récapitulation des différents gains est dans le tableau ci-dessous.

N	Gain Bo avec $K_{pi} = 1$ [dB]	$K_{pi}$ pour avoir la marge de phase [dB]	Gain tot [dB]
8	19.74	8.73	28.47
10	17.8	13.3	31.1
14	14.88	16.2	31.08
15	14.28	17	31.28
16	13.72	17.2	30.92
20	11.78	17.9	29.68
25	9.84	18.1	27.94

Le gain maximum est pour  $N = 15$  donc la constante de temps du régulateur vaut :

$$\tau_{ii} = N \cdot 267 \cdot 10^{-6} = 15 \cdot 267 \cdot 10^{-6} = 4ms \text{ et } K_{pi} = 17dB = 7.08$$

Le diagramme de Bode avec ce régulateur est le suivant :

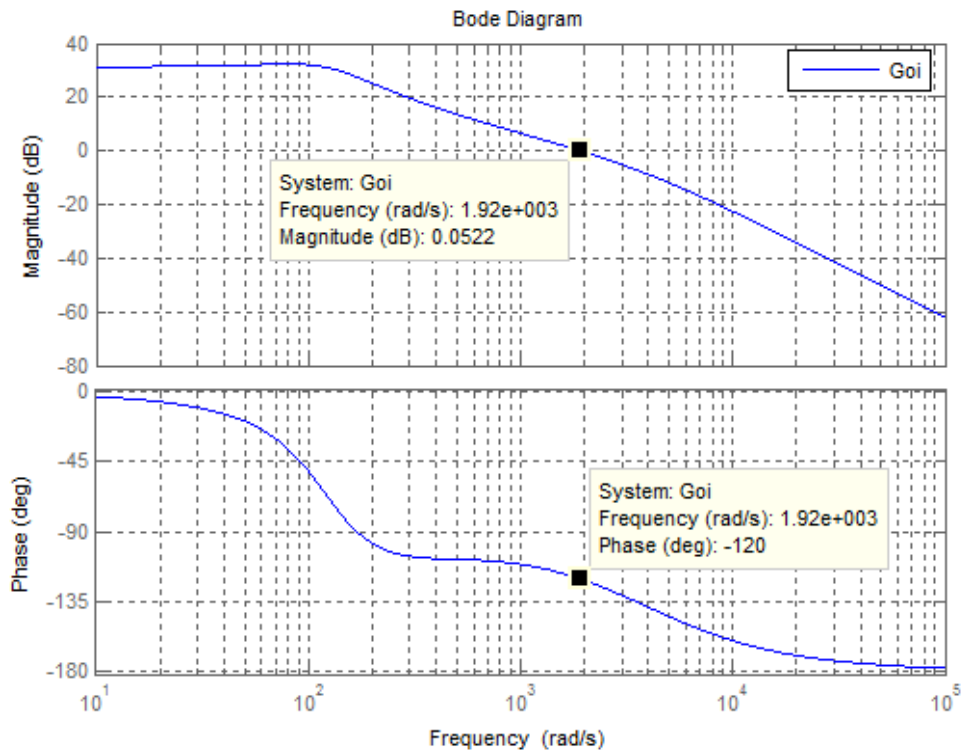


Figure 17: Diagramme de Bode avec le régulateur 2 ajusté

### 5.7 Régulateur 3 : marge de phase de 45° et compensation du pôle dominant

Le diagramme de bode en boucle ouverte pour  $K_{pi}=1$  sur la figure ci-dessous :

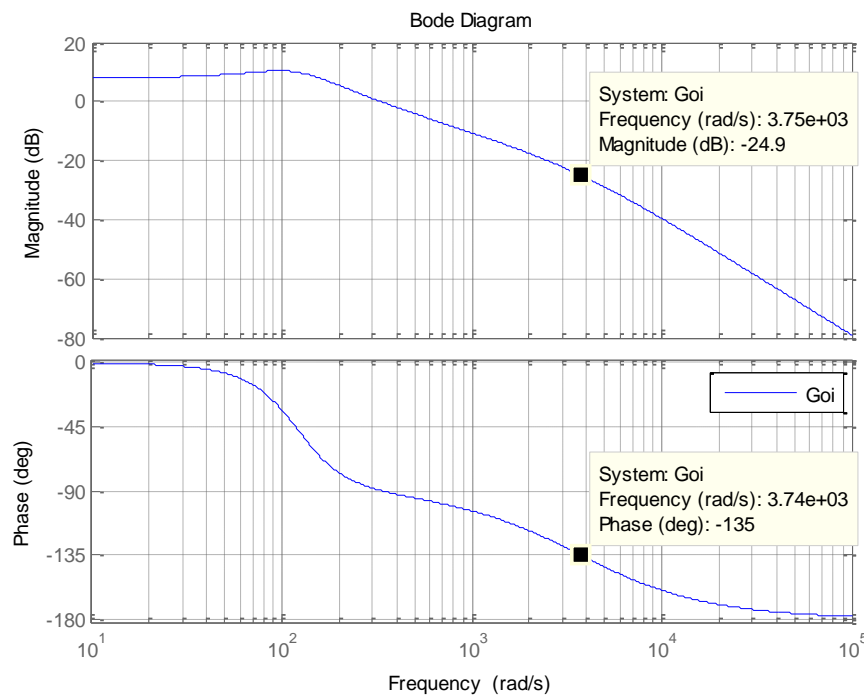


Figure 18: Diagramme de Bode avec le régulateur 3 avec  $K_{pi}=1$

Il est maintenant possible de trouver le gain à ajouter afin d'obtenir la marge de phase de 60°.

Pour avoir une marge de phase de  $45^\circ$  (comme c'est affiché sur le diagramme) il faut un gain  $K_{pi}$  de 24.9 dB soit 17.579.

Le nouveau diagramme de Bode avec la nouvelle valeur de  $K_{pi}$  calculée pour une marge de phase de  $45^\circ$  est sur la courbe ci-dessous.

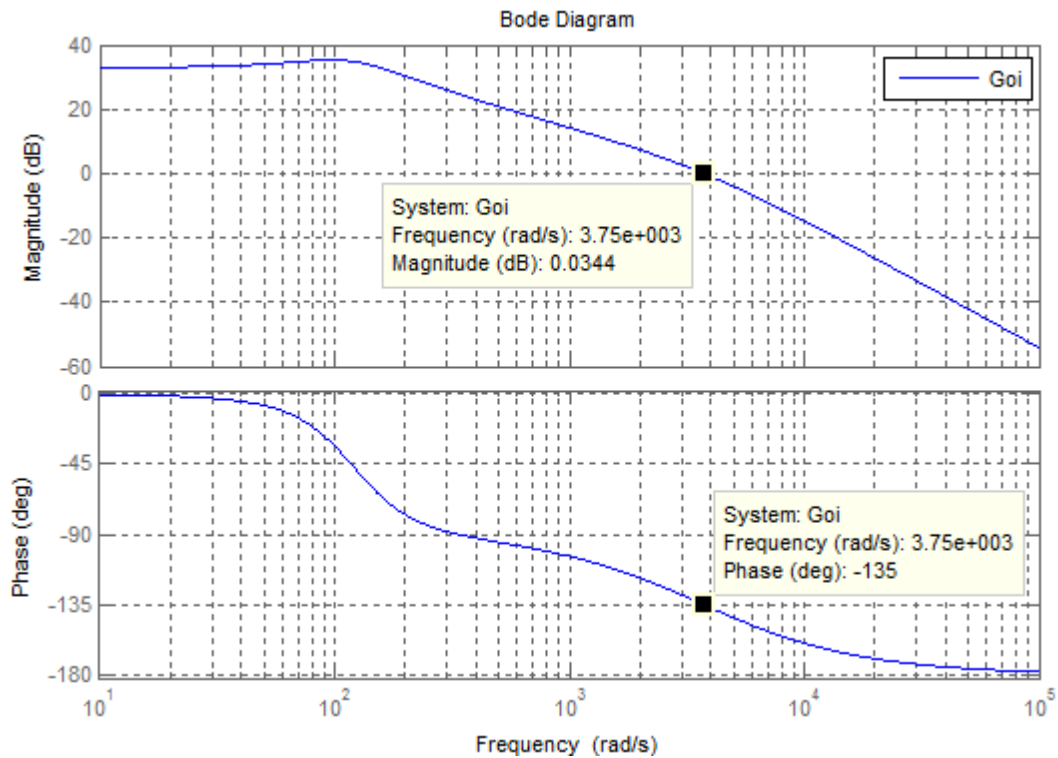


Figure 19: Diagramme de Bode avec le régulateur 3 ajusté

## 5.8 Régulateur 4 : marge de phase de $45^\circ$ et compensation des petites constantes de temps

Pour trouver la valeur de  $N$  idéale il faut trouver le gain en boucle ouverte le plus grand. La récapitulation des différents gains est dans le tableau ci-dessous.

N	Gain Bo avec $K_{pi} = 1$ [dB]	$K_{pi}$ pour avoir la marge de phase [dB]	Gain tot [dB]
8	19.74	22	41.74
9	18.72	22.3	41.02
10	17.8	22.7	40.5
15	14.8	24	38.8
20	11.78	24.5	36.28

Le gain maximum est pour  $N = 8$  donc la constante de temps du régulateur vaut :

$$\tau_{ii} = N \cdot 267 \cdot 10^{-6} = 8 \cdot 267 \cdot 10^{-6} = 2.136 \text{ ms} \text{ et } K_{pi} = 22 \text{ dB} = 12.59$$

Le diagramme de Bode avec ce régulateur est le suivant :

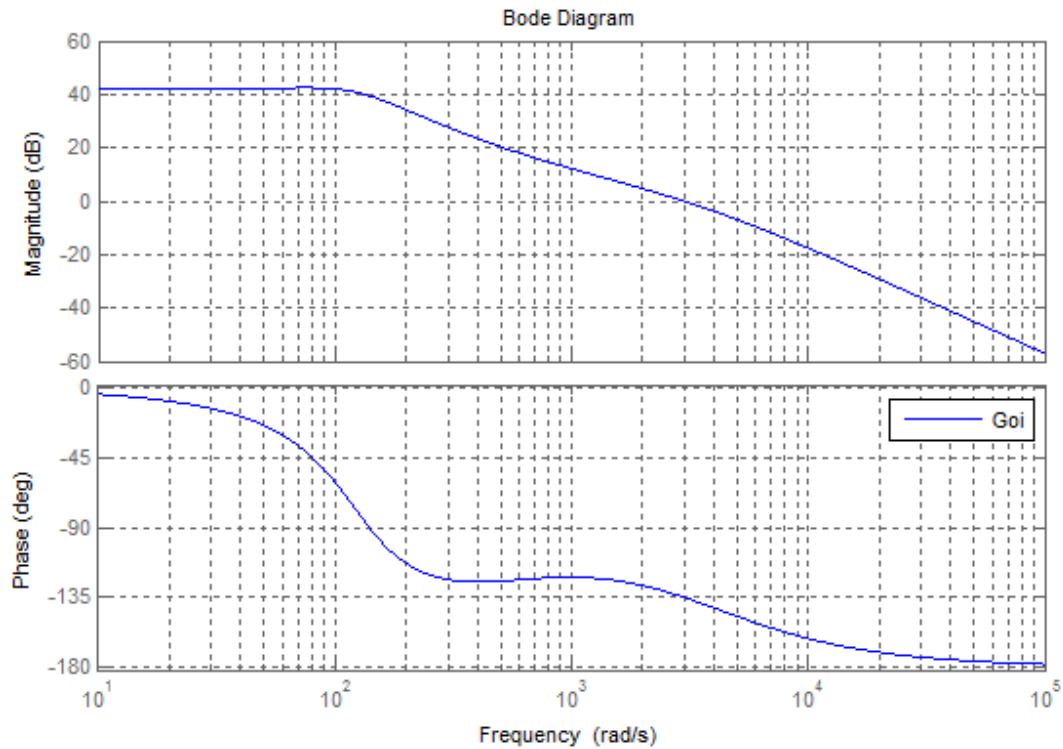


Figure 20: Diagramme de Bode avec le régulateur 2 ajusté

## 6 Implémentation du régulateur PI en C

### 6.1 Loi de commande du régulateur

La loi de commande du régulateur est la suivante :

$$u(t) = K_p \left( e(t) + \frac{1}{T_{ii}} \int_0^t e(t) dt \right)$$

En passant en numérique et en posant que  $e\_int = \int_0^t e(t) dt$ , il vient :

$$u[k] = K_p \left( e[k] + \frac{1}{T_{ii}} e\_int[k] \right)$$

Il est possible d'écrire l'intégrale de l'erreur sous la forme suivante :

$$e\_int[k] = e\_int[k-1] + e[k] \cdot h$$

Pour ne pas diviser l'intégrale de l'erreur par  $T_{ii}$  et donc pour gagner du temps, il faut définir une constante  $G_{ii}$  qui vaut :

$$G_{ii} = \frac{1}{T_{ii}}$$

Il vient donc :

$$u[k] = K_p (e[k] + G_{ii} \cdot e\_int[k])$$

## **6.2 Implémentation en C**

L'implémentation en C se fait en traduisant les équations ci-dessus en langage C.

Le calcul de l'erreur est :

erreur = consigne - iMoteur ;

$e\_int[k] = e\_int[k-1] + e[k] \cdot h$  devient :

erreur\_int += erreur \* h ;

Et  $u[k] = K_p(e[k] + G_{ii} \cdot e\_int[k])$  devient

Tension = Kp \* (erreur + Gii \* erreur\_int) ;

## **6.3 Anti windup**

L'anti windup entre en action si la sortie de la commande devient limitée.

Dans ce cas il faut intégrer une erreur fictive à la place de l'erreur réelle.

L'erreur fictive à intégrer est :

$$e_{lim} = e - \frac{(U_{commande} - U_{max\ variateur})}{K_p}$$

Comme avant pour ne pas faire de division il faut définir une constante  $G_p = \frac{1}{K_p}$

## **6.4 Programme complet**

Le programme complet se trouve en annexe au point 9.2 à la page 32.

# **7 Test du régulateur de courant**

## **7.1 Procédure**

Pour faire les mesures il faut suivre la même procédure que pour les mesures précédentes, mais il faut charger le programme au point 9.2 à la page 32.

Pour cette mesure l'allure de la consigne est un saut de tension appliqué pendant un certain temps puis la consigne redevient nulle.

Il faut configurer la console comme sur l'image suivante :

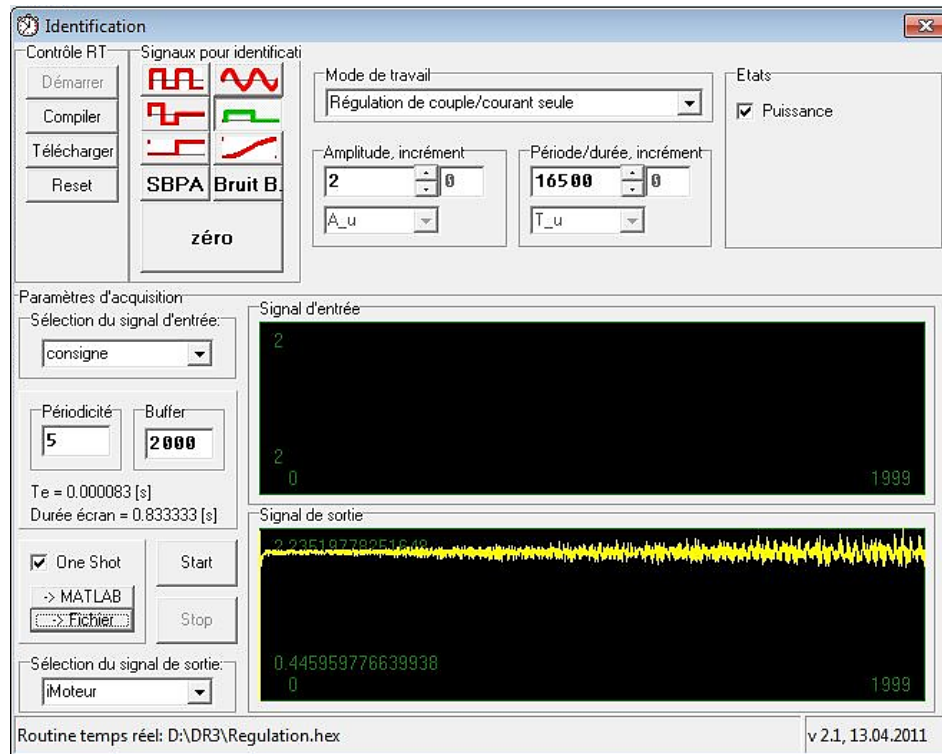


Figure 21: Fenêtre de commande pour le test des régulateurs

Puis il faut enregistrer les mesures dans un fichier.

Pour pouvoir passer d'un régulateur à un autre il n'est pas nécessaire de changer le code du programme puis de le recharger et faire les mesures. Il est possible de changer les paramètres du programme en allant dans l'onglet Tools puis Variable List et depuis cette fenêtre il est aisé de changer les paramètres du régulateur ou de sélectionner l'anti windup. Il ne faut toutefois pas oublier de cliquer sur le bouton d'actualisation. La liste des variables se présente sous la forme suivante :

Variables List					
Name	Value	Periodicity	Buffer	Type	Size
Kp	12.58930015	0	1	float	1
Gii	468.1706846	0	1	float	1
erreur	-0.00668578	0	1	float	1
anti_windup	10	0	1	float	1
user_var5	0	0	1	int	1
user_var6	0	0	1	int	1
user_var7	0	0	1	int	1
user_var8	0	0	1	int	1
user_var9	0	0	1	int	1
user_var10	0	0	1	int	1
user_var11	0	0	1	int	1
user_var12	0	0	1	int	1
user_var13	0	0	1	int	1
user_var14	0	0	1	int	1
user_var15	0	0	1	int	1
user_var16	0	0	1	int	1
uCons	0	0	1	float	1
iMoteur	2.010203836	0	2000	float	1
omegaMoteur	-0.00039269	0	1	float	1
thetaMoteur	18523.32226	0	1	float	1
nombreTours	2948	0	1	int	1
udcm	101.1853027	0	1	float	1
mode	3	0	1	int	1

Figure 22: Liste des variables dans RTP Watch



## 7.2 Régulateur 1

Le régulateur 1 est dimensionné pour une marge de phase de  $60^\circ$  et sa constante de temps est basée sur la compensation du pôle dominant.

La comparaison entre la simulation et la mesure est la suivante :

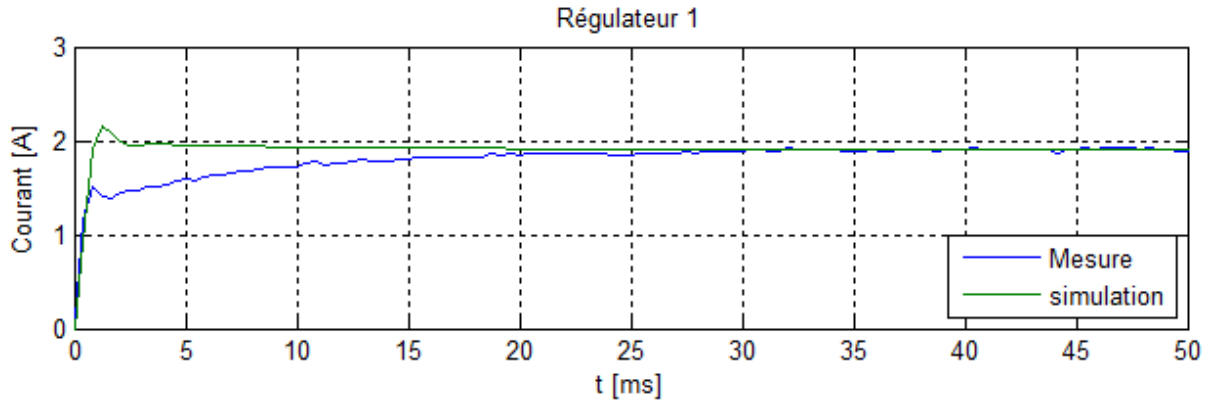


Figure 23 : Comparaison de la réponse indicielle mesurée et simulée pour une marge de  $60^\circ$  (compensation du pôle dominant)

Pour la mesure du temps de réglage, on a effectué une moyenne glissante sur 5 mesures pour lisser l'ondulation du courant et obtenir de meilleurs résultats car à l'oscillation pouvait sortir des  $\pm 5\%$  même après le transitoire.

	Treg [ms]	Erreur statique [%]
Simulation	0.8	4.55
Mesure	32.5	4.5

- Pour ce cas, l'erreur statique mesurée est très proche de la valeur de simulation. Mais le temps de régulation de la mesure est très supérieur à celui de la simulation.

## 7.3 Régulateur 2

Le régulateur 2 est dimensionné pour une marge de phase de  $60^\circ$  et sa constante de temps est basée sur la compensation des petites constantes de temps.

La comparaison entre la simulation et la mesure est la suivante :

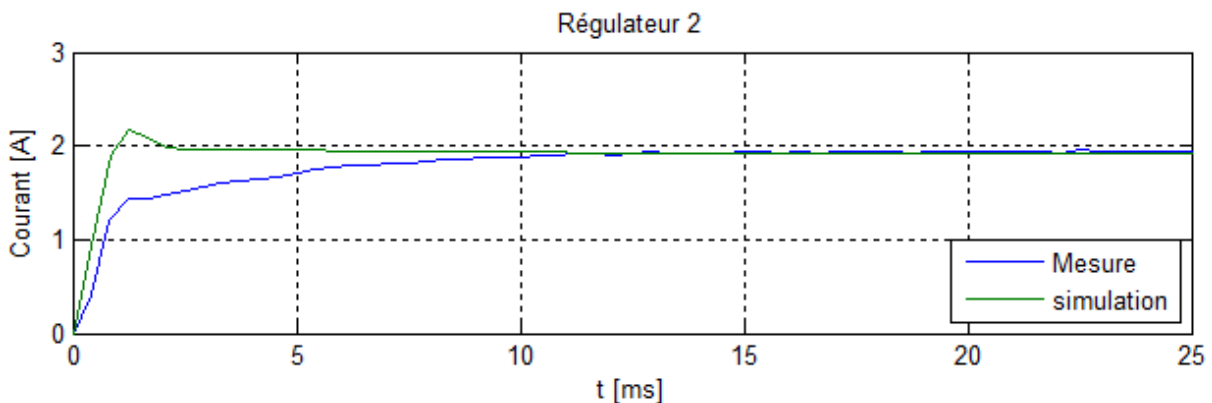


Figure 24 : Comparaison de la réponse indicielle mesurée et simulée pour une marge de  $60^\circ$  (la compensation des petites constantes de temps)

	Treg [ms]	Erreur statique [%]
<b>Simulation</b>	0.8	4.55
<b>Mesure</b>	10.8	4

- Pour ce cas, l'erreur statique de simulation est un peu supérieure de celle de la mesure.
- Mais le temps de régulation de la mesure est très supérieur à celui de la simulation.
- Il apparaît que la compensation des petites constantes de temps est donne un meilleur résultat que la compensation du pôle dominant en terme de temps de réglage et d'erreur statique.

### 7.4 Régulateur 3

Le régulateur 3 est dimensionné pour une marge de phase de  $45^\circ$  et sa constante de temps est basée sur la compensation du pôle dominant.

La comparaison entre la simulation et la mesure est la suivante :

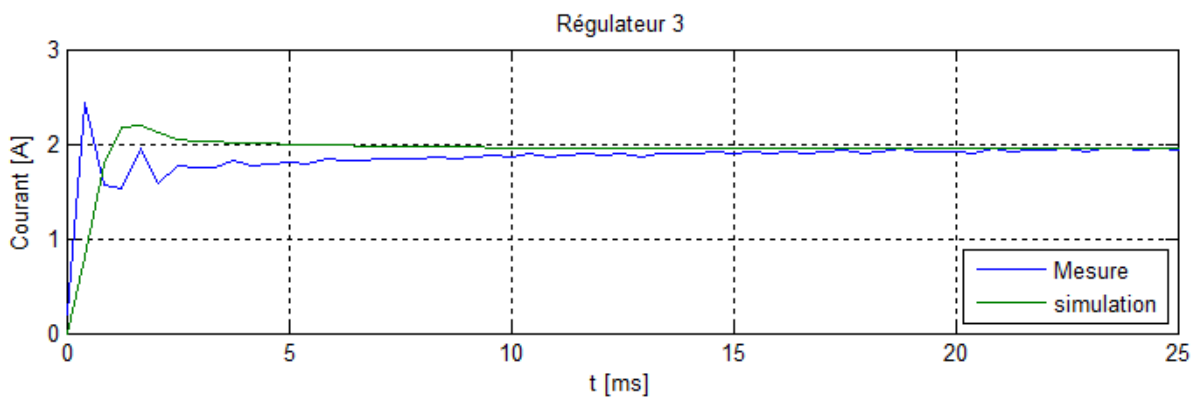


Figure 25 : Comparaison de la réponse indicielle mesurée et simulée pour une marge de  $45^\circ$  (compensation du pôle dominant)

	Treg [ms]	Erreur statique [%]
<b>Simulation</b>	1.7	2.65
<b>Mesure</b>	16	2.2

- Il y a une différence entre les deux courbes de mesure et de la simulation, la mesure est plus rapide pour la montée du courant mais le dépassement est plus grand.
- L'erreur statique de simulation est un peu supérieure de celle de la mesure.
- Mais le temps de régulation de la mesure est très supérieur à celui de la simulation.
- Il apparaît qu'avec une marge de phase de  $45^\circ$  le système est plus rapide et plus précis qu'avec une marge de phase de  $60^\circ$ .

### 7.5 Régulateur 4

Le régulateur 4 est dimensionné pour une marge de phase de  $45^\circ$  et sa constante de temps est basée sur la compensation des petites constantes de temps.

La comparaison entre la simulation et la mesure est la suivante :

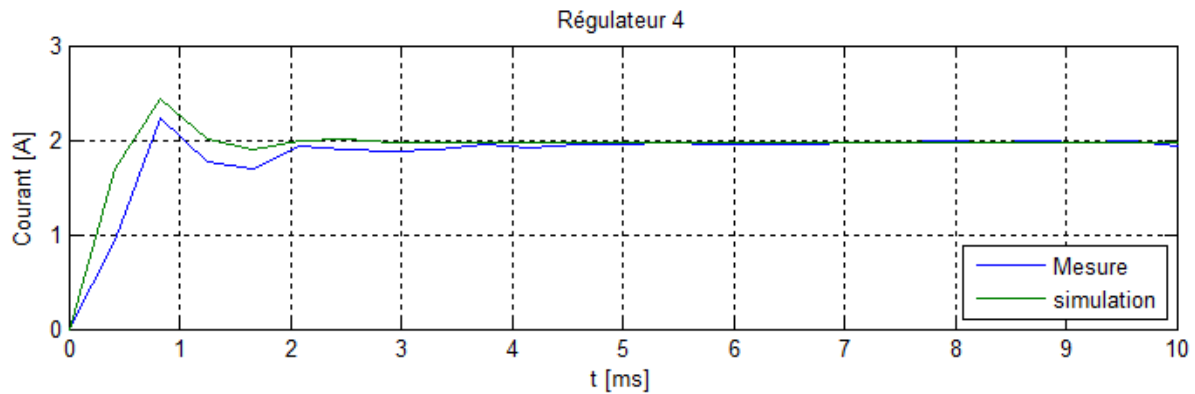


Figure 26 : Comparaison de la réponse indicielle mesurée et simulée pour une marge de  $45^\circ$  (la compensation des petites constantes de temps)

	Treg [ms]	Erreur statique [%]
Simulation	1	2.2
Mesure	3.4	1.1

- Pour ce cas, la courbe de mesure suit bien la courbe de la simulation.
- L'erreur statique de simulation est le double de celle mesurée.
- Le temps de régulation de la mesure est très légèrement supérieur à celui de la simulation.
- Le système est plus précis pour une marge de phase de  $45^\circ$  qu'avec une marge de  $60^\circ$ . Mais le système reste très proche de l'instabilité.
- Il apparaît que la compensation des petites constantes de temps donne un meilleur résultat que la compensation du pôle dominant en terme de temps de réglage et d'erreur statique.

## 7.6 Anti windup

La courbe suivante illustre l'efficacité de l'anti windup.

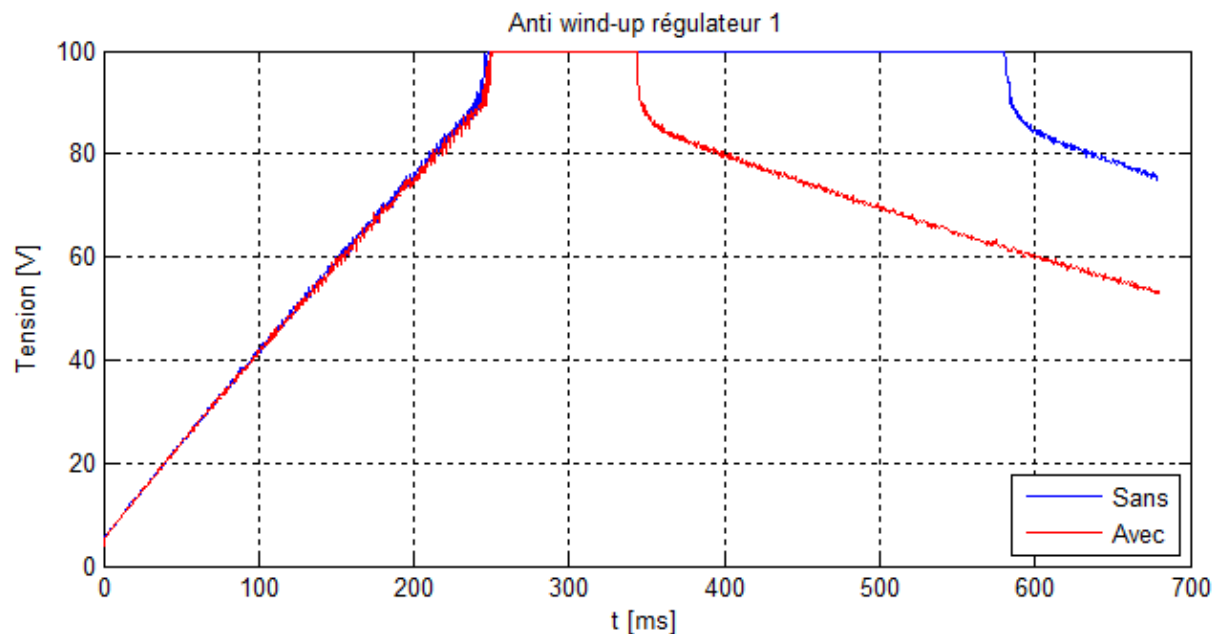


Figure 27: Anti windup

- Il est visible que l'anti windup est efficace car lorsqu'il est actif (courbe rouge) la commande reste saturée beaucoup moins de temps que lorsqu'il est inactif (courbe bleu).

## 7.7 Test à rotor bloqué

Pour pouvoir mettre en évidence la partie proportionnelle du régulateur il faut commencer par mettre le gain  $G_{ii}$  à 0 dans la liste des variables. Puis il faut lancer la mesure en maintenant le rotor bloqué. Pour cela il suffit simplement de le retenir à la main. La courbe relative à cet essai est la suivante

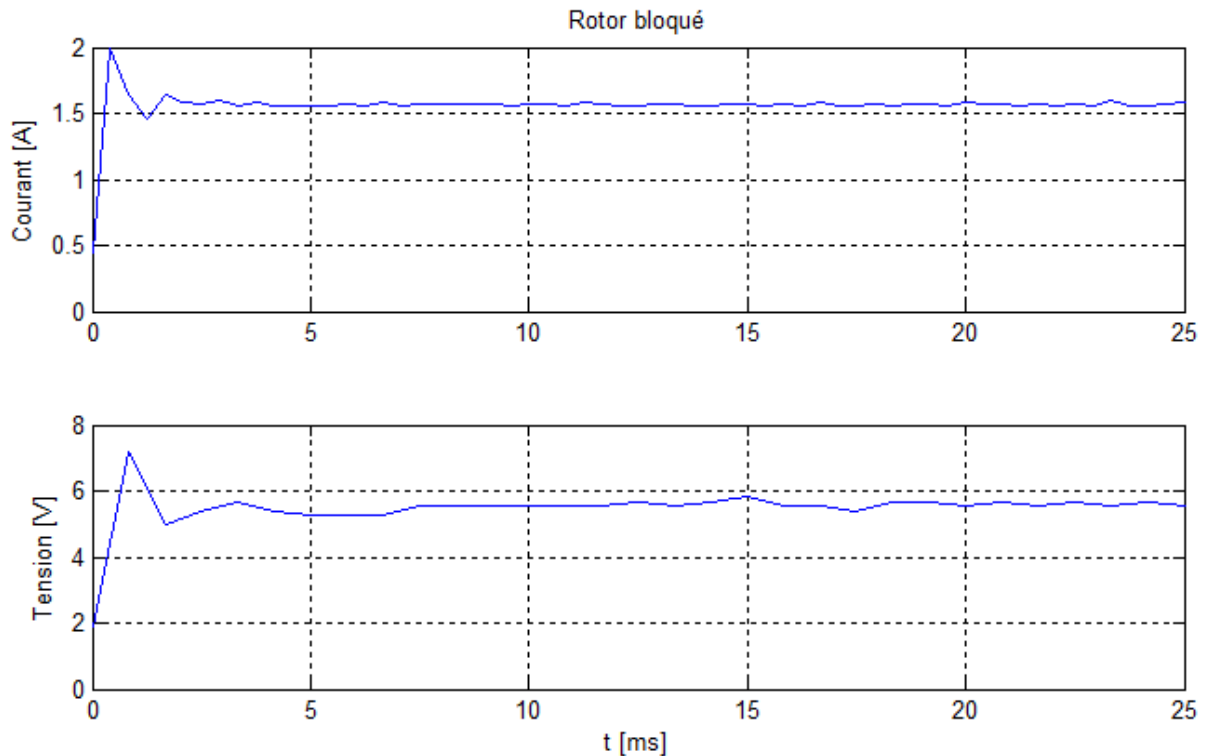


Figure 28: Test à rotor bloqué

Treg [ms]	Erreur statique [%]
$\infty$	20

- Il est visible que lorsqu'il n'y a qu'un régulateur P le système a une grande erreur statique de l'ordre de 20%. Et que du fait de cette grande erreur statique le temps de réglage  $T_{reg}$  est égal à l'infini. Ce type de régulateur n'est donc pas adapté pour avoir un réglage précis du courant dans le moteur.

## 8 Conclusion

Ce laboratoire a été très intéressant car il nous a permis de mieux assimiler la théorie vu en cours de mécatronique 2. En effet, nous avons appris à caractériser un servomoteur DC à aimants permanent et à définir un modèle théorique réaliste à l'aide de matlab qui nous aide à dimensionner le régulateur approprié.

Dans une première partie, nous avons utilisé une régulation en courant. Le régulateur utilisé est du type PI.

Dans une seconde partie, nous avons pu acquérir la méthode de synthèse d'un régulateur PI de courant.

Nous avons constaté en conclusion que la méthode des petites constantes de temps est meilleure que celle de compensation du pôle au niveau de la régulation de courant, car elle permet d'avoir un meilleur temps de réglage et une erreur statique plus faible.

Nous avons conclu aussi lors de nos mesures et de nos simulations que le système est plus précis et plus rapide avec une marge de phase de  $45^\circ$  qu'une marge de  $60^\circ$ .

Donc le meilleur des quatre régulateurs est celui qui a une marge de phase de  $45^\circ$  et utilise la compensation des petites constantes de temps. Cependant plus la marge de phase est faible plus le risque que le système devienne instable à cause d'une perturbation est grand.

Yverdon le 18 novembre 2013

D.Houmard

R.Rejeb

## 9 Annexes

### 9.1 Code Matlab

```

1. %% Nettoyage
2. clc; close all; clear all;
3.
4. %% Constantes
5. fe = 12e3; % [Hz]
6. % Moteur
7. Kt = 0.34; % [Nm/A]
8. Ke = Kt; % [V*s/rad]
9. Ra = 0.45; % [ohm]
10. La = 3.3e-3; % [H]
11. % Mécanique
12. Jm = 0.0023; % [kgm^2]
13. Rext = 0.5 * 50e-3; % [m]
14. RtrouInt = 0.5 * 16e-3; % [m]
15. RtrouExt = 0.5 * 6e-3; % [m]
16. Rdtrou = 0.5 * 37e-3; % [m]
17. l = 20e-3; % [m]
18. rho = 8500; % [kg/m^3]
19.
20. %% Calcul de l'inertie de la charge
21. % Inertie du grand cylindre
22. Jcylg = 1/2 * (rho*pi*Rext^2*l) *
    Rext^2;
23. % Inertie du cylindre central
24. Jcylp = 1/2 * (rho*pi*RtrouInt^2*l) *
    RtrouInt^2;
25. % Inertie des trous de goupilles
26. Jtrou = 4 * (1/2 *
    (rho*pi*RtrouExt^2*l) * RtrouExt^2 +
    ...
27. (rho*pi*RtrouExt^2*l)*Rdtrou^2);
28. % Inertie totale
29. Jch = Jcylg - Jcylp - Jtrou;
30. Jtot = Jm + Jch;
31.
32. %% Calcul des constantes de temps
    électriques et mécanique
33. tauE = La/Ra; % [s]
34. tauM = Jtot * Ra / (Kt * Ke); % [s]
35.
36. %% Fonctions de tranfert théorique
    du moteur
37. % Variateur
38. Kcmth = 1;
39. tauCmth = 1/fe;
40. Gcmth = tf(Kcmth, [tauCmth, 1]);
41. % Moteur
42. Gui = tf([Jtot/(Kt*Ke), 0],
    [tauM*tauE, tauM, 1]);
43. % Système à régler théorique
44. Gaith = Gui * Gcmth;
45. % Affichage du système théorique
46. figure('color', 'w');
47. bode(Gaith);
48. legend('Gai théorique');
49. grid;
50.
51. %% Identification de Gai avec
    sysquake
52. K = 1;
53. fcMi = 1000;
54. Tr = 1.078416701116164e-4; % Retard
    pur du variateur
55. % Filtre PB
56. Kmi = 1;
57. Kcm = 1;
58. Kpb = Kcm * Kmi;
59. tauMi = 1/(2*pi*fcMi);
60. tauPB = tauMi + Tr;
61. Gpb = tf(Kpb, [tauPB, 1]);
62.
63. %% Système à régler Gai depuis
    Sysquake
64. Gai = Gui * Gpb;
65. Kai = Kcm*Kmi*Jtot/(Kt*Ke);
66. % Affichage
67. figure('color', 'w');
68. bode(Gai, Gaith);
69. legend('Gai avec sysquake', 'Gai
    théorique');
70. grid;
71.
72. %% Identification avec ident
73. %% chargement des données
74. % mesure du double saut
75. mes = load('mesures45V20h.dat');
76. tIdent = mes(:,1);
77. uIdent = mes(:,2);
78. iIdent = mes(:,5);
79. tIdent = tIdent(1:40);
80. uIdent = uIdent(1:40);
81. iIdent = iIdent(1:40);
82. % fonctions de transferts
83. newData1 = load('-mat',
    'identification2');
84. % Create new variables in the base
    workspace from those fields.
85. vars = fieldnames(newData1);
86. for iIdent = 3:length(vars)
87.     assignin('base', vars{iIdent},
        newData1.(vars{iIdent}));
88. end
89. clc;
90. % Adaptation de la phase
91. [Agai2, Pgai2, Wgai2] = bode(Gai2);
92. Agai2 = squeeze(Agai2);
93. Pgai2 = squeeze(Pgai2);
94. Wgai2 = squeeze(Wgai2);
95. Agai2 = 20*log10(Agai2);
96. Pgai2 = Pgai2-360;
97. % Affichage
98. figure('color', 'w');
99. bode(Gai2);
100. legend('Gai depuis ident');
101. grid;
102. figure('color', 'w');
103. subplot(2, 1, 1);
104. semilogx(Wgai2, Agai2);
105. title('Diagramme de bode');
106. legend('Gai depuis ident');
107. ylabel('Amplitude [dB]');
108. xlabel('Pulsation [rad/s]');
109. grid;
110. subplot(2, 1, 2);

```

```

111. semilogx(Wgai2, Pgai2);
112. ylabel('Phase [°]');
113. xlabel('Pulsation [rad/s]');
114. grid;
115. % Détermination des pôles
116. p = pole(Gai2);
117. tau1 = 1/abs(p(1, 1));
118. tau2 = 1/abs(p(2, 1));
119. tau3 = 1/abs(p(3, 1));
120. tau = [tau1, tau2, tau3];
121. Kstat = dcgain(Gai2);
122. % Comparaison des trois Gai
123. figure('color', 'w');
124. bode(Gai2, Gai, Gaith);
125. legend('Gai avec ident', 'Gai avec
    sysquake', 'Gai théorique');
126. grid;
127.
128. %% Dimensionnement du régulateur
    PI
129. tauMax = abs(2*tauM*tauE/(tauM -
    sqrt(tauM^2 - 4 * tauM * tauE)));
130. % Régulateur 60° + tau max
131. Kpi1 = 10^(18.5/20);
132. Gri1 = tf(Kpi1*[tauMax, 1],
    [tauMax, 0]);
133. % Régulateur 60° + tau pct
134. Kpi2 = 10^(17/20);
135. N2 = 15;
136. tauPct2 = N2 * tauPB;
137. Gri2 = tf(Kpi2*[tauPct2, 1],
    [tauPct2, 0]);
138. % Régulateur 45° + tau max
139. Kpi3 = 10^(24.9/20);
140. Gri3 = tf(Kpi3*[tauMax, 1],
    [tauMax, 0]);
141. % Régulateur 45° + tau pct
142. Kpi4 = 10^(22/20);
143. N4 = 8;
144. tauPct4 = N4 * tauPB;
145. Gri4 = tf(Kpi4*[tauPct4, 1],
    [tauPct4, 0]);
146. % Fonction globale
147. Goi = Gai * Gri4;
148. % Affichage
149. w = logspace(1, 5, 1e4);
150. figure('color', 'w');
151. bode(Goi, w);
152. legend('Goi');
153. grid;
154. figure('color', 'w');
155. step(Goi, 0.5);
156. legend('Goi');
157. grid;
158.
159. %% Mesures
160. % Chargement des mesures
161. % Mesures des courants avec les
    régulateurs
162. I1 = load('Reg1_imot.dat');
163. I2 = load('Reg2_imot.dat');
164. I3 = load('Reg3_imot.dat');
165. I4 = load('Reg4_imot.dat');
166. t = I1(1:1633,1);
167. I1 = I1(1:1633,5);
168. I2 = I2(1:1633,5);
169. I3 = I3(1:1633,5);
170. I4 = I4(1:1633,5);

171. % Test de l'anti windup
172. U1 = load('Reg1_ucons.dat');
173. U1 = U1(1:1633,5);
174. U1AW = load('Reg1_ucons_w.dat');
175. U1AW = U1AW(1:1633,5);
176. % Régulateur P
177. I = load('Reg4_queP.dat');
178. I = I(1:1633,5);
179. % Test à rotor bloqué
180. IRbloque =
    load('Reg4_rotor_bloque_imot.dat');
181. URbloque =
    load('Reg4_rotor_bloque_ucons.dat');
182. IRbloque = IRbloque(1:1633,5);
183. tU = URbloque(1:1633,1);
184. URbloque = URbloque(1:1633,5);
185. % Réponse impulsionnelle
186. % Création des fonctions de
    transfert en boucle fermée
187. GbfReg1 = feedback(Gai*Gri1,1);
188. GbfReg2 = feedback(Gai*Gri2,1);
189. GbfReg3 = feedback(Gai*Gri3,1);
190. GbfReg4 = feedback(Gai*Gri4,1);
191. GbfRegP = feedback(Gai*Kpi4,1);
192. % Modification de l'amplitudde de
    la fonction Step
193. % opt =
    stepoptions('StepAmplitude',2);
194. % Calcul de la réponse
    impulsionnelle pour tout les
    régulateurs
195. u = 2.*ones(size(t));
196. [ISimReg1,tSim] = lsim(GbfReg1, u,
    t);
197. [ISimReg2] = lsim(GbfReg1, u, t);
198. [ISimReg3] = lsim(GbfReg2, u, t);
199. [ISimReg4] = lsim(GbfReg3, u, t);
200. [ISimRegP] = lsim(GbfReg4, u, t);
201. % Affichage
202. % Réponse impulsionnelle des
    quatre régulateurs
203. % Régulateur 1
204. figure('color', 'w');
205. subplot(2,1,1);
206. plot(t*1000, I1, tSim*1000,
    ISimReg1);
207. % axis([0 50 0 3]);
208. ylabel('Courant [A]');
209. xlabel('t [ms]');
210. title('Régulateur 1');
211. legend('Mesure', 'simulation',
    'location', 'southeast')
212. grid;
213. % Régulateur 2
214. subplot(2,1,2);
215. plot(t*1000, I2, tSim*1000,
    ISimReg2);
216. % axis([0 25 0 3]);
217. ylabel('Courant [A]');
218. xlabel('t [ms]');
219. title('Régulateur 2');
220. legend('Mesure', 'simulation',
    'location', 'southeast')
221. grid;
222. figure('color', 'w');
223. % Régulateur 3
224. subplot(2,1,1);

```

```

225. plot(t*1000, I3, tSim*1000,
        ISimReg3);
226. % axis([0 25 0 3]);
227. ylabel('Courant [A]');
228. xlabel('t [ms]');
229. title('Régulateur 3');
230. legend('Mesure', 'simulation',
        'location', 'southeast')
231. grid;
232. % Régulateur 4
233. subplot(2,1,2);
234. plot(t*1000, I4, tSim*1000,
        ISimReg4);
235. % axis([0 10 0 3]);
236. ylabel('Courant [A]');
237. xlabel('t [ms]');
238. title('Régulateur 4');
239. legend('Mesure', 'simulation',
        'location', 'southeast')
240. grid;
241. xlabel('t [ms]');
242. % Efficacité de l'anti windup
243. figure('color', 'w');
244. plot(t*1000, U1, t*1000,
        U1AW, 'r');
245. % axis([0 75 0 2]);
246. title('Anti wind-up régulateur
        1');
247. ylabel('Tension [V]');

248. xlabel('t [ms]');
249. legend('Sans', 'Avec', 'location',
        'southeast')
250. grid;
251. % Régulateur P
252. figure('color', 'w');
253. plot(t*1000, I, tSim*1000,
        ISimRegP);
254. % axis([0 0.13 0 2]);
255. title('Régulateur P');
256. ylabel('Courant [A]');
257. xlabel('t [ms]');
258. legend('Mesure', 'simulation',
        'location', 'northeast')
259. grid;
260. % Essai à rotor bloqué
261. figure('color', 'w');
262. subplot(2,1,1);
263. plot(t*1000, IRbloque);
264. axis([0 25 0 2]);
265. title('Rotor bloqué');
266. ylabel('Courant [A]');
267. grid;
268. subplot(2,1,2);
269. plot(tU*1000, URbloque);
270. axis([0 25 0 8]);
271. ylabel('Tension [V]');
272. xlabel('t [ms]');
273. grid;

```

## 9.2 Implémentation du régulateur PI

```

#define RTPWATCH_MODE_NONE 1
#define RTPWATCH_MODE_IDENT_BCL_INTERNE 2
#define RTPWATCH_MODE_REG_BCL_INTERNE 3
#define RTPWATCH_MODE_IDENT_BCL_EXT 4
#define RTPWATCH_MODE_REG_BCL_EXT 5
#define RTPWATCH_MODE_REG_BCL_EXT_PID 6
#define h 0.000083333
#define UDC 100.
float Kp;
float Gii;
float Gp;
float erreur;
float erreur_int;
float Kp_prec;
float coeff_lim;
float anti_windup;
void InitialisationEtudiant()
{
    FLOAT_WATCH(Kp);
    FLOAT_WATCH(Gii);
    FLOAT_WATCH(erreur);
    FLOAT_WATCH(anti_windup);
    // initialisation des variables
    Kp=8.4140;
    Gii=120.9007;
    Gp = 1./Kp;
    anti_windup = 0;
    erreur = 0.;
    erreur_int = 0.;
    Kp_prec = 0.;
    coeff_lim = 0.;
}
float Regulation(FlagsRegulation flags, float consigne, float iMoteur, float
omegaMoteur,
        float thetaMoteur, float udcM)

```



```
{
    float tension;
    // Calcul de 1/Kp si Kp a changé
    if (Kp != Kp_prec)
    {
        Gp = 1/Kp;
        Kp_prec = Kp;
    }
    switch(flags.bit.Mode)
    {
        case RTPWATCH_MODE_IDENT_BCL_INTERNE:
        {
            tension=consigne;
        }

        break;
        case RTPWATCH_MODE_REG_BCL_INTERNE:
        { // Calcul de l'erreur
            erreur = consigne-iMoteur;
            // Calcul de l'intégrale de l'erreur
            erreur_int += (erreur-coeff_lim) * h;
            // Calcul de la tension de sortie
            tension = Kp * (erreur + Gii * erreur_int);
            // Si la tension est hors des limite du variateur
            if (tension > UDC || tension < -UDC)
            {
                // Anti windup
                if (anti_windup > 0.5)
                {
                    coeff_lim = Gp * (tension-UDC);
                }
                else
                {
                    coeff_lim = 0.;
                }
                // Limitation de tension
                if (tension > 0)
                {
                    tension = UDC;
                }
                else
                {
                    tension = -UDC;
                }
            }
            else // Si pas de limitation
            {
                coeff_lim = 0.;
            }
        }
        break;
        default:
        {
            tension = 0;
        }
    }
    return tension;
}
```