CYCLE 5

RESOLUTION DE PROBLEMES PAR UTILISATION DE L'INGENIERIE NUMERIQUE OU L'APPRENTISSAGE AUTOMATISE

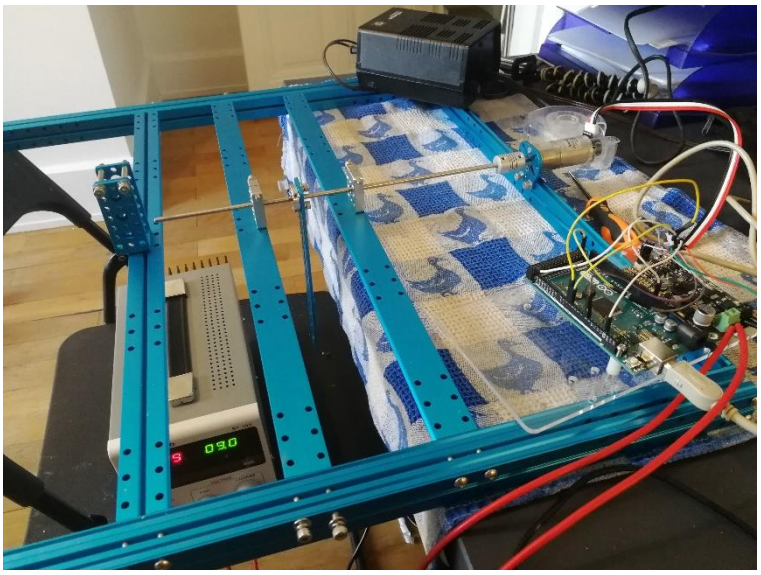
TD3 - PSI

TD 3

RESEAUX DE NEURONES - CORRECTEUR

L'objectif de ce TP est de comparer la correction de système par solution PID et par un correcteur de type réseau de neurones.

1 MISE EN SITUATION



Un bras manipulateur est le bras d'un robot généralement programmable, avec des fonctions similaires à un bras humain. Les liens de ce manipulateur sont reliés par des axes permettant, soit de mouvement de rotation (comme dans un robot articulé) et/ou de translation (linéaire) de déplacement.

Dans le cas d'une imitation complète d'un bras humain, un bras manipulateur a donc 3 mouvements de rotation et 3 mouvements de translation sur son élément terminal.

Il peut être autonome ou contrôlé manuellement et peut être utilisé pour effectuer une variété de tâches avec une grande précision.

Les bras manipulateurs peuvent être fixes ou mobiles (c'est-à-dire à roues) et peuvent être conçus pour des applications industrielles.

La maquette présentée ci-dessus permet de comprendre les tenants et les aboutissants d'un tel système.

Le cahier des charges fixe les paramètre suivants pour une entrée échelon :

— une rapidité à 5% de 1 s ;

— pas de dépassement ;

— une erreur statique nulle.

2 Correction par PID

Activité 1

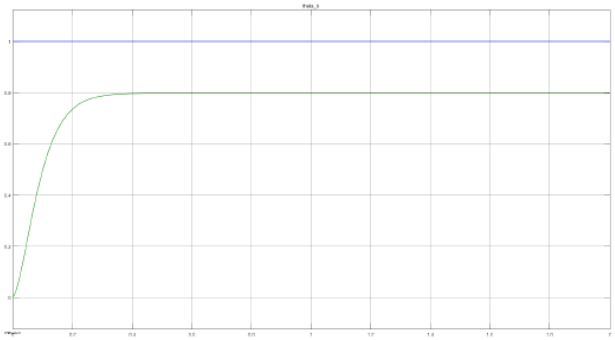
Prendre connaissance du schéma-blocs proposé dans le fichier `Activite_01_Maquette.slx`. Identifier les « parties » modélisation de la machine à courant continu, modélisation du couple résistant et structure de l'asservissement.

Du classique. Voir TP précédent.

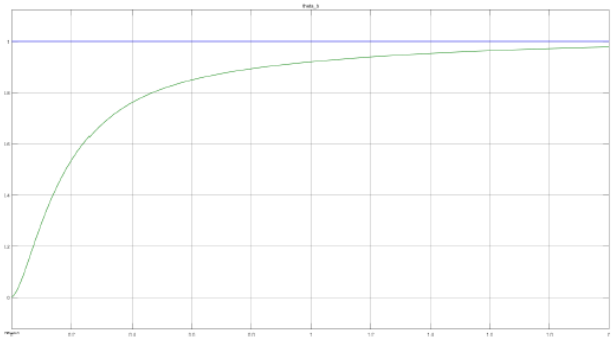
Activité 2

Exécuter le modèle. Quelles sont les performances qui ne sont pas validées ? En déduire un type de correction nécessaire. Choisir ce correcteur et utiliser le « PID tuner » du logiciel afin de satisfaire le cahier des charges. Expliquer les écarts entre prévision du « PID tuner » et de la simulation.

Par défaut, la performance non validée est la précision fig. 2a. Une correction de type PI permet alors de compenser ce défaut en garantissant le non-dépassement. Le « PID tuner » permet de proposer des valeurs pour les coefficients K_p et K_i. La réponse prédite par ce module répond parfaitement aux objectifs. Après simulation, la réponse n'est pas aussi efficace que la prédiction le supposer (fig. 2b). En effet, le logiciel a linéarisé le comportement du système enfin de proposer les valeurs numériques du correcteur.



(a) Système bouclé sans correction



(b) Système bouclé avec correction optimale

Activité 3

Ouvrir le fichier `Activite_03_Maquette_Commande`. Copier-coller le bloc PI déterminé à l'activité 2. Quantifier et qualifier les écarts entre modèle et réel.

L'implémentation de ce correcteur produit le résultat de la à laquelle a été superposée la réponse simulée. L'allure globale est est satisfaisante même si des comportements locaux n'ont pas été bien modélisés.

3 Correction par réseau de neurones

Il s'agit en préambule d'avoir lu le cours sur les réseaux de type NARX et la section concernant le contrôle des systèmes dynamiques.

Activité 4

Faire une synthèse des Modele Reference Adaptative Controller (MRAC).

Consulter le document.

3.1 Pré-entrainement

Activité 5

Proposer un modèle idéal à identifier afin de satisfaire les exigences du cahier des charges ? On pourra se demander quel est le comportement souhaité, en régime permanent et en régime transitoire, quel ordre de fonction permet de vérifier ceci, etc.

La réponse temporelle d'un système d'ordre 1 ou d'ordre 2 avec $z > 1$ permet de valider l'exigence de non dépassement. L'exigence de précision peut être atteinte quel que soit l'ordre avec une amplification statique de 1 ; enfin la rapidité n'influence pas le choix. Les auteurs proposent de choisir plutôt l'ordre 2 par la non-discontinuité de sa dérivée en 0^+ afin de rendre plus souple le comportent du correcteur par réseau de neurones. Ainsi la fonction de transfert objectif à suivre est :

$$H_{obj}(p) = \frac{1}{1 + 2 \times 1\frac{p}{5} + \frac{p^2}{5^2}} \tag{1}$$

De la même manière que lors du TP d'identification, une clé de la bonne modélisation par réseau de neurones réside dans les données.

Activité 6

Proposer et générer des données permettant un bon entraînement du réseau de neurones correcteur. Préciser les amplitudes de variation des caractéristiques de ce signal : amplitude, durée, nombres et échantillonnage.

Le système sera sollicité essentiellement en réponse à des échelons de positions. On se fixe une période d'échantillonnage liée aux caractéristiques de la carte de commande soit $T_e = 0{,}01\,\text{s}$. La durée de ces échelons doivent être compris entre 0,5 s (choix arbitraire) et 5 s (choix permettant der mettre en évidence le régime permanent). Les amplitudes de sortie correspondent aux possibilités du système. Ici, nous souhaitons entraîner le réseau de neurones sur une amplitude $[-2\pi; 2\pi]$ rad. Le nombre de 20 000 points permet d'obtenir globalement une petite centaine d'échelons, ce qui semble être un nombre correct vis-à-vis de entraînement qui suit.

Dans le TP précédent, l'identification du système par un réseau de neurone a déjà été effectué. Par conséquent, on peut passer à l'étape suivante : l'entraînement.

3.2 Entraînement

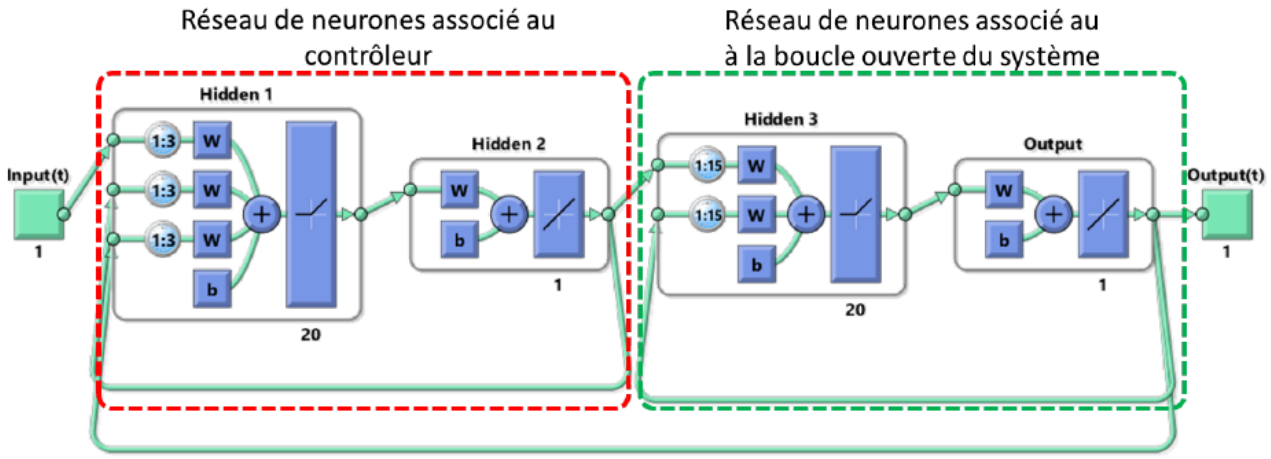
Activité 7

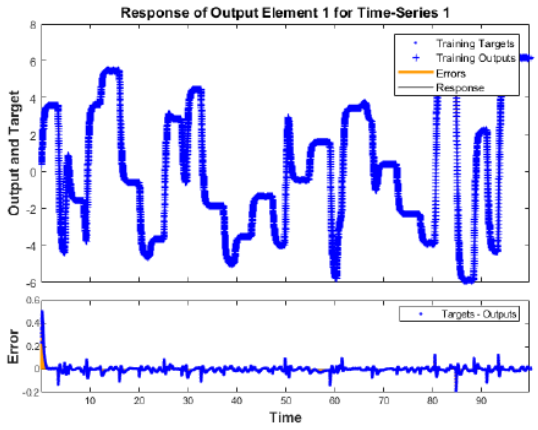
À l'aide du fichier `Activite_07.mlx`, créer le réseau de neurone du contrôleur. Pour cela :

- renseigner les caractéristiques des données d'entrée ;
- configurer le configuration du système attendu en boucle fermée ;
- configurer les caractéristiques des échelons d'entrée ;
- configurer les caractéristiques du réseau de neurone associé au contrôleur.

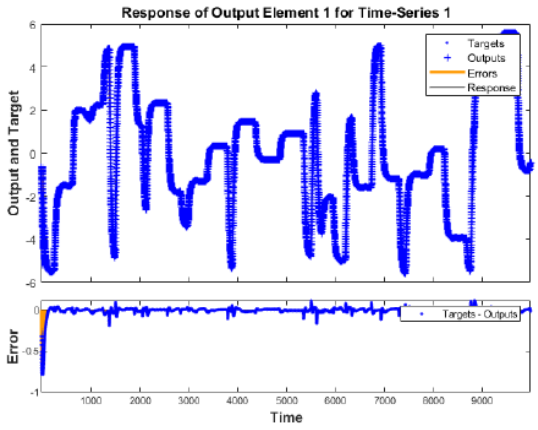
Lancer alors l'entraînement du réseau avec le contrôleur.

La structure proposée pour le réseau de neurone donnée figure 3.
Les résultats après entraînement et sur un jeu de données de test sont données dans la figure 4.





(a) Résultat de l'entraînement du contrôleur



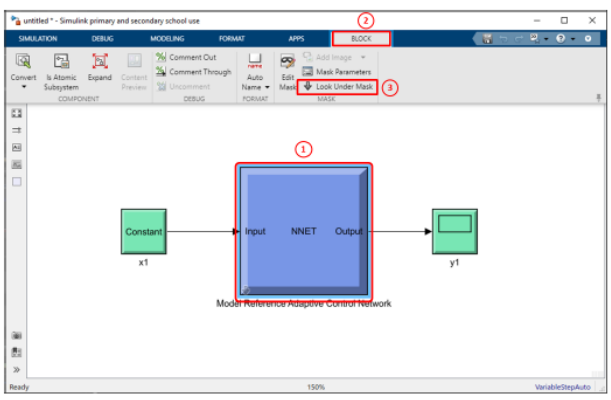
(b) Simulation du réseau avec des données de test

3.3 Implémentation sur le système

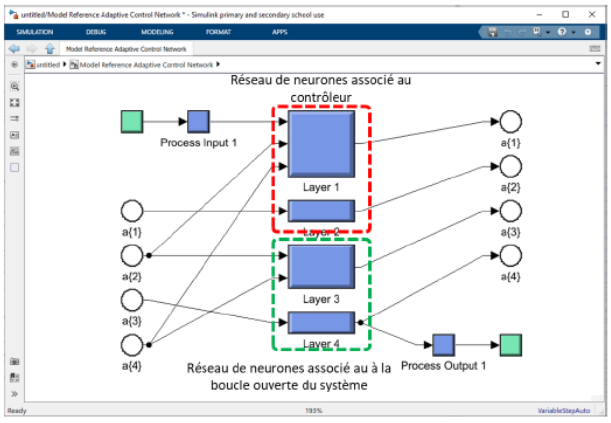
Activité 8

En utilisant les consignes suivantes, générer le fichier Simulink permettant de piloter le système par un réseau de neurones. On utilisera la fichier `Activite_08_PilotageANN_Vide.slx`.

Pour implémenter le réseau de neurone associé au contrôleur, nous allons copier les blocs Simulink associés au réseau de neurone du contrôleur et les coller sur une feuille permettant de piloter le système. Commençons par générer le bloc Simulink associé au MRAC en utilisant la commande `gensim(mrac_net)` et affichons ensuite la structure du réseau (Figure 5).



(a) Bloc Simulink du MRAC

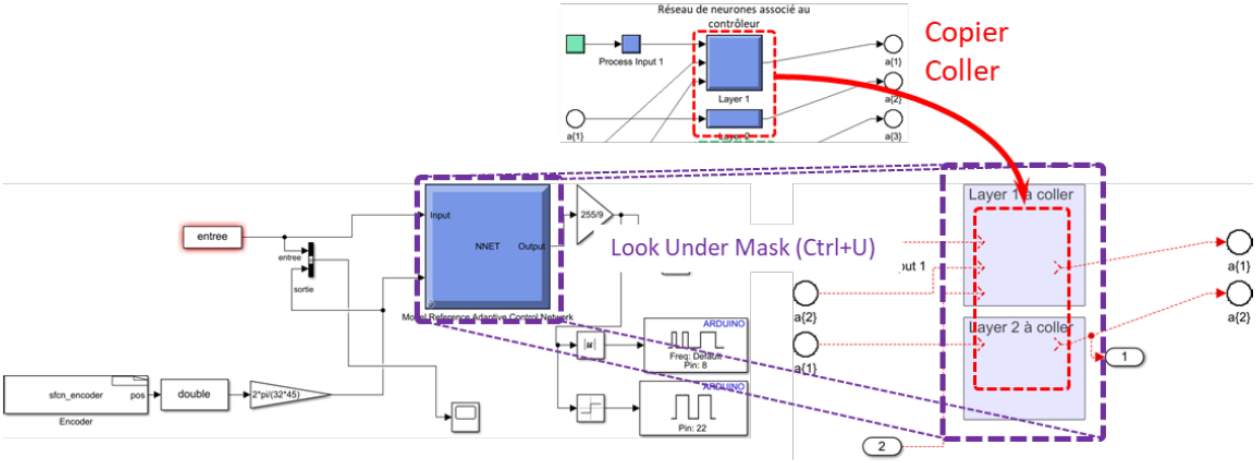


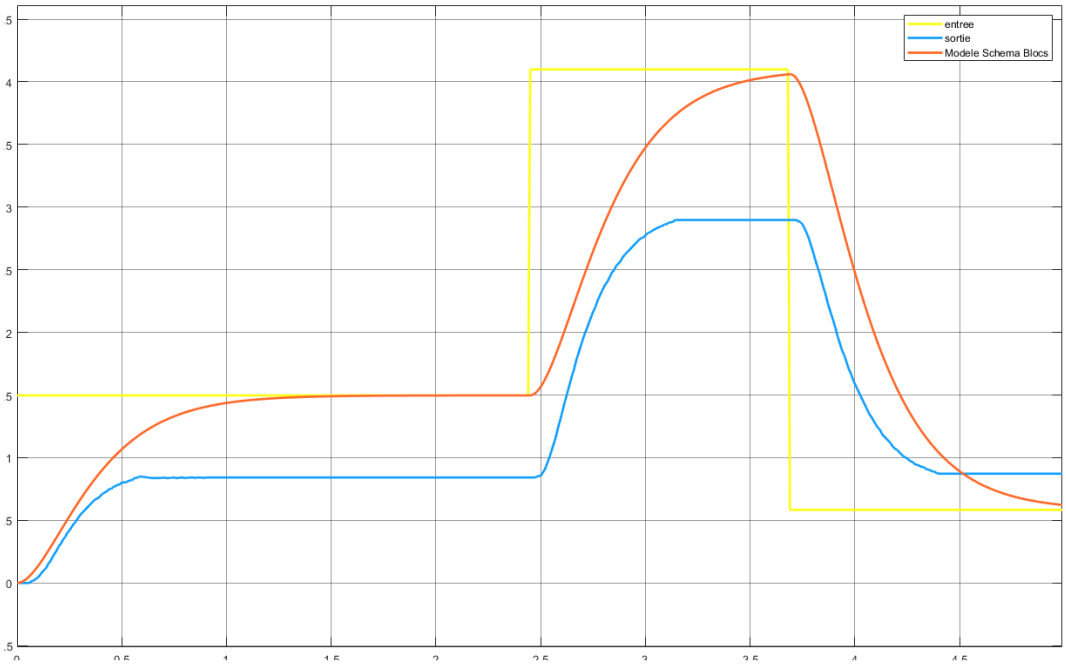
(b) Structure Simulink du réseau de neurones

La figure 6 (il faut avouer qu'elle est super belle) illustre la zone où copier le réseau de neurones dans le fichier `Activite_08_PilotageANN_Vide.slx`.

Activité 9

Implémenter sur le système réel et évaluer les écarts.





Évaluation des écarts par rapport au cahier des charges :

- Écart statique souhaité : 0 rad. Écart statique mesuré :0,9 rad s^{-1}.
- Temps de réponse souhaité 1 s. Temps de réponse mesuré : < 1 s.

On observe un écart, notamment sur la valeur de l'écart statique. Cela peut s'expliquer par une « mauvaise » identification du comportement du système lors de l'entraînement du réseau de neurones. Il est en effet difficile d'obtenir des résultats satisfaisants avec une identification sur un modèle non linéaire. Une mauvaise modélisation de la tension de seuil peut donc expliquer que le contrôleur ne parvienne pas à supprimer l'écart statique.

Il faudrait donc déterminer des paramètres de chacun des réseaux de neurones permettant d'obtenir de meilleurs résultats.

4 Synthèse

Activité 10

Proposer une synthèse sur les avantages et les inconvénients sur la correction mise en œuvre dans ce TP par rapport à l'utilisation d'un correcteur PI(D).

Avantages :

- Relative facilité de mise en œuvre du processus de corrections.
- Possibilité de réglage sur PC avant implémentation sur cible.
- Pas/Peu d'essais nécessaires.
- Possibilité d'entraîner le correcteur avec une fonction de transfert directement choisie par le concepteur.

Inconvénient :

- Nécessité de disposer d'un modèle relativement fiable.
- Difficulté de choisir les caractéristiques du réseau de neurones.
- Difficulté de choisir le type d'entrées.
- Processus d'entraînement peu répétable et qui peut échouer.