

TD 2

RESEAUX DE NEURONES - IDENTIFICATION

L'objectif de ce TD est de tester et de comprendre comment marche les réseaux de neurones avec Matlab, ici NARX. Vous avez à votre disposition un cours plus complet sur les réseaux de neurones et l'approfondissement des NARX.

La problématique sera ici de proposer un modèle de comportement d'un système multiphysique par identifications temporelles. La première identification sera « classique » au sens d'une analyse causale : la deuxième se fera par un réseau de neurone de type NARX.

1 MISE EN SITUATION

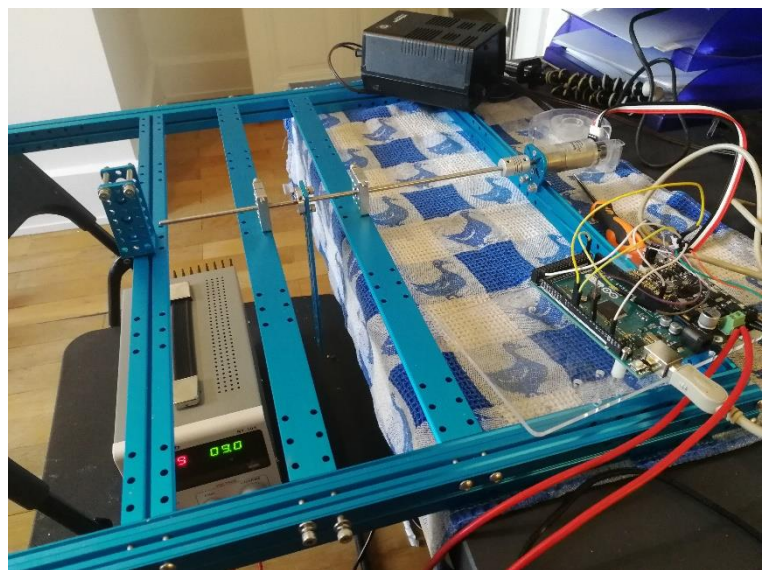
Un bras manipulateur est le bras d'un robot généralement programmable, avec des fonctions similaires à un bras humain. Les liens de ce manipulateur sont reliés par des axes permettant, soit de mouvement de rotation (comme dans un robot articulé) et/ou de translation (linéaire) de déplacement.

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.



2 MODELISATION PAR SCHEMA BLOC

Activité 1

Prendre connaissance du schéma-blocs proposé dans le fichier `Activite_01_Maquette.slx`. Retrouver l'ensemble des théorèmes, principes et hypothèses afin de construire ce schéma-blocs du système en boucle ouverte. Identifier l'influence de la perturbation sur la réponse.

On cherche à identifier quelques paramètres parmi l'ensemble des paramètres physiques.

Activité 2

Proposer l'ensemble des expériences à mettre en place afin de déterminer K_m , J_{eq} , R , L , f_v , m , r et d .

Afin d'améliorer les valeurs numériques des estimations précédentes des paramètres trop éloignées de leur valeurs pratiques, on cherche à superposer une réponse expérimentale à une réponse simulée.

Un essai en Boucle Ouverte a été réalisé sur la maquette, les résultats sont dans `essai_BO_echelon_9V_5s.mat`. Charger les valeurs dans le **workspace(CTRL+H)** avant de lancer l'activité_03.

Activité 3

En exploitant le fichier `Activite_03_Maquette`, réaliser un essai en BO pour une entrée d'une tension de 9V pendant 5s et superposer cette réponse à la réponse simulée. En déduire les deux principaux paramètres mal évalués.

Afin de réduire les écarts entre le modèle et le réel, un algorithme de types $f(x) = 0$ va être utilisé sur les deux paramètres identifiés précédemment. L'algorithme proposé est celui de Newton introduit sous la forme :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (5)$$

qui se généralise pour un système d'équations à plusieurs variables :

$$\underline{x}_{n+1} = \underline{x}_n - \mathbf{J}^{-1}(\underline{x}_n) \cdot \underline{f}(\underline{x}_n) \quad (6)$$

où \mathbf{J} représente la matrice jacobienne définie par :

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_{p-1}} & \frac{\partial f_1}{\partial x_p} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_{p-1}} & \frac{\partial f_2}{\partial x_p} \\ \dots & \dots & \dots & \dots & \dots \\ \frac{\partial f_{m-1}}{\partial x_1} & \frac{\partial f_{m-1}}{\partial x_2} & \dots & \frac{\partial f_{m-1}}{\partial x_{p-1}} & \frac{\partial f_{m-1}}{\partial x_p} \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \dots & \frac{\partial f_m}{\partial x_{p-1}} & \frac{\partial f_m}{\partial x_p} \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \dots & \frac{\partial f_m}{\partial x_{p-1}} & \frac{\partial f_m}{\partial x_p} \end{bmatrix} \quad (7)$$

Activité 4

À la lecture du code proposé dans le fichier `Activite_04_Maquette.m`, retrouver les étapes suivantes :

- construction de la matrice jacobienne – comment est-elle construite ?
- première itération ;
- boucle ;
- conditions de fin d'itération.

Activité 5

Exécuter le code et vérifier que la correction des deux paramètres identifiés permet de coller au mieux à la réponse expérimentale. Conclure quant à la qualité du modèle à prédire une réponse du système en BO à une sollicitation à un échelon.

3 Modélisation par un réseau de neurones

Il s'agit en préambule d'avoir lu le cours sur les réseaux de type NARX.

Activité 6

Faire une synthèse des réseaux de type NARX.

Consulter le document.

Il s'agit d'entraîner le réseau de neurones. On retrouve les étapes classiques d'un entraînement :

- choix des données ;
- structure du réseau ;
- entraînement ;
- analyse des résultats
- itération
- fermeture du modèle.

On rappelle ici que la modélisation par un réseau de neurones est un processus itératif.

3.1 Pré-entraînement

Activité 7

Proposer un type de signal d'entrée à imposer au système afin d'entraîner celui-ci. préciser les amplitudes de variation des caractéristiques de ce signal : amplitude, durée, nombres et échantillonnage. Penser également à l'amplitude du signal de sortie.

Générer alors un signal d'entrée permettant d'entraîner convenablement le système en BO.

Piloter le système/modèle par ce signal d'entrée.

Activité 8

En s'appuyant sur le cours, proposer une première structure du réseau de neurones avec le nombre d'entrées différées, nombre de couches, nombres de neurones et fonctions d'activation.

3.2 Entraînement

On entrainera le modèle avec deux types d'entraînement :

- régularisation de Bayes ;
- Levenberg-Marquardt.

Dans un premier temps, la régularisation de Bayes est utilisé afin d'exploiter son résultat intéressant de proposition du nombre de paramètres utilisés par rapport par rapport nombre de paramètres disponibles.

Activité 9

Mettre à jour le script Matlab `Activite_09_NN.mlx` avec les paramètres fixés aux activités [7](#) et [8](#). Exécuter ce script. Observer la qualité prévisionnelle du modèle. Quel est le nombre de paramètres utilisés ? Est-il nécessaire d'augmenter ou de diminuer le nombre de neurones ?

Activité 10

Ré-itérer le processus d'entraînement afin de valider la modélisation.

3.3 Post-entraînement

Activité 11

Observer les produits de corrélation. En déduire si le nombre d'entrées retardées est suffisant. Puis ré-itérer l'ensemble du processus afin d'améliorer la qualité du modèle.

La qualité doit être à cette étape très bonne. Néanmoins, il a été construit sur l'hypothèse que le modèle était parfait et que par conséquent la deuxième entrée du modèle était exactement la sortie. Afin de prendre en compte un retour non parfait, il faut « boucler » réellement le NARX.

Activité 12

Boucler le modèle par réseau de neurones. Commenter la précision de celui-ci modèle. Entraîner-le de nouveau si nécessaire.

4 Conclusion

Activité 13

Tester le réseau de neurones entraîné sur un nouveau jeu de données.

Activité 14

Proposer une synthèse sur les avantages et les inconvénients des deux types de modélisations mises en place dans ce TP.