الجمهوريـــة الجزائريـــة الديمقراطيـــة الشعبيـــة

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
وزارة التعليـم العالي والبحـث العلمـي
MINISTERE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ BADJI MOKHTAR-ANNABA
FACULTÉ DES SCIENCES DE L'INGÉNIEUR
DÉPARTEMENT D'ÉLECTRONIQUE

جامعة باجي مختار- عنابة
كلية علوم الهندسة
قسم الإلكترونيك

Année 2006

THÈSE

Présentée en vue de l'obtention du diplôme de

DOCTORAT D'ÉTAT

Thème

Commande des systèmes dynamiques linéaires et non linéaires par la stratégie prédictive

Option

AUTOMATIQUE INDUSTRIELLE

Par

Mohamed Larbi SAIDI

Directeur de thèse :	H. A. ABBASSI	Professeur	U. ANNABA

DEVANT LE JURY

Président :	BEDDA M.	Professeur	U. ANNABA
Examinateurs :	DEBBACHE N.E	Professeur	U. ANNABA
	SAAD S.	Maître de conférence	U. ANNABA

Remerciements

Je remercie, tout d'abord, les membres du jury qui ont accepté de juger ce travail :

- *Mr M.Bedda, professeur et directeur du laboratoire d'automatique et des signaux du département d'électronique de l'université de Annaba, pour l'honneur qu'il me fait de présider le jury ;*

- *Mr N.Debbache, professeur et doyen de la faculté des sciences de l'ingénieur de l'université de Annaba, d'avoir accepté d'être examinateur ;*

- *Mr S.Sâad, Maître de conférence au département d'électromécanique de l'université de Annaba, d'avoir accepté d'être examinateur ;*

- *Et enfin, Mr H.A.Abbassi, professeur au département d'électronique de l'université de Annaba et directeur de cette thèse. Je tiens à le remercier pour son dynamisme, pour ses nombreuses discussions et pour m'avoir permis d'accomplir ces travaux dans les meilleures conditions.*

Je tiens également à remercier Mr H.Arioui pour m'avoir invité au sein du laboratoire des systèmes complexes d'Evry –France-, de m'avoir proposé l'application dans le domaine des simulateurs de conduite, et pour ses précieux conseils. Sans oublier bien sûr de remercier le directeur de ce laboratoire et son staff.

Je profite de cette occasion qui m'est offerte pour remercier tous mes amis et mes collègues qui m'ont beaucoup aidé durant cette période.
Enfin, les derniers mots sont pour remercier toute ma famille pour leur aide et leurs encouragements constants, et spécialement ma mère et ma femme pour leur patience et leur soutien permanent.

Mounir

Avant propos

Les travaux présentés dans cette thèse, ont donné lieu aux différentes communications scientifiques suivantes :

Publication dans un journal international

- M.L. Saidi, A.Debbeh, H.Arioui, S.Kermiche, H.A.Abbassi,
  "Predictive control of motion platform in driving simulator", *Asian journal of information technology, ISSN 1682-3915,* Vol.5, Number 2, 2006, pp. 133-138.

Communication à une conférence internationale

- M.L. Saidi, S.Kermiche, A.Debbeh, F.Arbaoui, H.A.Abbassi
  "Neural networks in predictive control", *Conférence Internationale sur la productique, CIP'05*, Tlemcen, Décembre 2005.

Communication à une conférence nationale

- M.L. Saidi, S.Kermiche, H.A.Abbassi, F.Arbaoui,
  "Neural generalized predictive control (study and simulation) », *Conférence nationale sur l'ingénierie de l'electronique, CNIE'04*,Oran, novembre 2004.

Résumé

La notion de prédiction se voit de plus en plus importante dans la commande des systèmes automatiques, les décisions à prendre selon un comportement futur prédit. Dans cette optique, le travail présenté dans cette thèse s'articule autour d'une stratégie de commande utilisant un modèle de prédiction.

Le but de cette thèse est de développer des techniques de commande prédictive pour les systèmes dynamiques linéaires et non linéaires. D'abord, il est énoncé la loi de commande prédictive généralisée pour les systèmes linéaire ainsi que ses différentes caractéristiques. Ensuite, la version non linéaire de cette stratégie de commande est introduite. En effet, deux approches de la commande prédictive non linéaire sont présentées, la première est basée sur l'exploitation du prédicteur neuronal à un pas dans le calcul des prédictions, et la seconde est fondée sur l'utilisation d'un réseau de neurones comme extracteur de modèle linéaire autour d'un (ou plusieurs) point(s) de fonctionnement.

Enfin, dans le cadre de la lutte contre les accidents de la route, cette loi de commande prédictive est appliquée à une plateforme mobile d'un simulateur de conduite, conçu pour développer les réflexes chez les conducteurs dans les diverses situations de conduite. Les performances de cette loi de commande sont évaluées par simulation en exploitant des données réelles du véhicule.

Abstract

The prediction notion has become very important for the automatic control systems, the decisions to take according to the predicted future behaviour.
In this way, the work presented in this thesis is around a strategy using a prediction model.
The objective of this thesis is to develop predictive control techniques for linear and non linear systems.

Firstly, the generalized predictive control law for linear systems is presented with its characteristics.

Secondly, the non linear version of this strategy of control is introduced. Two non linear predictive control strategies are presented, the fist one is based on the one step ahead neural predictor for predictions calculations and the second one uses a neural network to extract a linear model around one (or several) operating point(s).

Finally, in order to decrease the number of road accidents this predictive control law is applied to a mobile platform of driving simulator moving in a restricted workspace, designed to develop driver reflexes in driving situations.
The performances of this control law are evaluated by simulation using real data of the vehicle.

ملخص

فكرة التنبوء أصبحت تأخذ أهمية كبيرة فى مجال أنظمة التحكم الألي، القرارات الواجب أخذوها وفقا لسلوك مستقبلي متنبىء.

في هذا السياق، العمل المقدم في هذه الأطروحة يدور حول استراتيجية تحكم تستعمل نموذج متنبىء.

الهدف من هذه الأطروحة هو تطوير تقنيات تحكم متنبىء للأجهزة الديناميكية الخطية و الاخطية.

أولا: يتم استعراض قانون تحكم متنبىء و خصائصه.

ثانيا: يتم إدخال الوجه الاخطي لقانون هذا التحكم، طريقتين للتحكم متنبىء قدماتا، الأولى تعتمد على استعمال عصب اصطناعي متنبىء لحساب جميع التنبوءات، و الثانية ترتكز على استعمال شبكة عصبية اصطناعية لاستخراج نموذج خطى بجوار نقطة أو نقاط التشغيل.

أخيرا، و فى نطاق التقليل من حوادث المرور، أستعملت هذه التقنية فى التحكم فى قاعدة متحركة لمحاك ممثل لسيارة، صنع لتطوير ردود الفعل عند سائقي السيارات فى مختلف أوضاع السياقة.

فعالية هذا القانون المقدم، أثبت بواسطة استخدام برامج محاكاة و قاعدة بيانات، محصل عليها من أجهزة التقاط للسيارة الحقيقية.

Listes des figures

Sommaire

CHAPITRE IV : *Commande prédictive généralisée non linéaire*……57

Introduction

La commande prédictive a trouvé une large application dans le domaine industriel, et un grand nombre d'algorithmes d'implémentation ont été présentés dans la littérature tel que la commande prédictive fonctionnelle (Predictive Functional Control : PFC) de Richalet en 1978 et la commande prédictive généralisée (Generalized Predictive Control : GPC) développée par Clarke en 1987. La plupart de ces algorithmes de commande utilisent un modèle du procédé pour prédire le comportement futur du système (Model Predictive Control : MPC).

L'une des caractéristiques qui a contribué au succès de la technologie (MPC) est celle qui a permis d'anticiper et d'éliminer les efforts des perturbations et d'avoir une meilleure poursuite du signal de référence.

Nous nous intéressons, ici, à la commande prédictive généralisée qui a montré son efficacité dans plusieurs domaines d'application. La nature non linéaire de la plupart des procédés et des applications industrielles, nous oblige à concevoir des versions non linéaires de cette stratégie de commande.

Les réseaux de neurones artificiels formels (RNF) possèdent la propriété d'être des approximateurs universels parcimonieux, nécessitent moins de paramètres ajustables que les méthodes de régression classiques, et à cause du développement des techniques algorithmiques d'apprentissage, notamment l'utilisation de méthodes d'optimisation non linéaires efficaces, qui associées à l'algorithme de rétro propagation pour l'évaluation du gradient, permettent des apprentissages rapides et précis. Armés de ces propriétés, les réseaux de neurones permettent d'obtenir, lorsqu'ils sont convenablement mis en œuvre, des résultats supérieurs à ceux des méthodes classiques de modélisation des systèmes non linéaires ils sont alors utilisés pour la modélisation des systèmes non linéaires. En plus les réseaux de neurones présentent trois caractéristiques intéressantes, ils sont adaptatifs, massivement parallèles et capables de généralisation. De ce fait, nous avons opté pour une version non linéaire de la GPC basée sur l'emploi des réseaux de neurones.

Dans ce travail, il est aussi mis l'accent sur la commande des plateformes mobiles utilisées dans les simulateurs de conduite d'automobile. Un simulateur de conduite est un outil de réalité virtuelle permettant l'étude comportementale du conducteur dans diverses situations de conduite.

La difficulté ou l'impossibilité de reproduire, dans le réel, les conditions de certaines situations routières accroissent l'intérêt de cet outil, utilisé pour confronter son utilisateur à des situations de conduite aussi proches que possible de la réalité.

D'un point de vu pratique, il est bien admis qu'aucun simulateur, aussi perfectionné soit-il, ne pourra reproduire exactement le mouvement du véhicule simulé. En effet, les véhicules utilisent de grandes distances alors que les plates-formes de restitution sont limitées en terme d'espace de travail, et certaines configurations et transitions ne peuvent, tout simplement, être reproduites à cause des limites technologiques.

Nous nous intéressons, ici, au mouvement longitudinal, qui a pour fonction de simuler une conduite en fil, en restituant des accélérations ou des décélérations sur des courtes courses. Pour donner au conducteur l'illusion de la sensation des effets inertiels du simulateur, la plateforme est équipée d'un algorithme classique washout. Le washout ou le système de restitution de mouvement fait penser le conducteur qu'il effectue des mouvements continus alors que l'espace de déplacement est limité. Les commandes assurent de faibles déplacements et un retour à la position neutre, durant les phases continues du signal d'accélération pour préparer la plateforme à un autre éventuel mouvement.

A cause des limitations de l'espace de déplacement de la plateforme et l'avantage de la commande prédictive généralisée de prendre en compte les contraintes imposées aux signaux d'entrée et de sortie, un régulateur GPC est utilisé pour commander le moteur à courant continu responsable du mouvement de translation.

Notre travail comporte essentiellement trois parties :
Une étude théorique consacré à la GPC linéaire, une autre réservée à la modélisation et à la commande des systèmes dynamiques non linéaires par réseaux de neurones et enfin, une application consacrée à l'utilisation de la GPC dans la commande d'une plateforme mobile d'un simulateur de conduite d'automobile.

Organisation de la thèse

Cette thèse regroupe cinq chapitres, qui vont de la définition de la commande prédictive à la résolution du problème de limitation de l'espace de déplacement des plates-formes mobiles.

Le chapitre I définit la commande prédictive généralisée linéaire (GPC), sa philosophie, ses concepts fondamentaux et présente ses avantages dans la commande des systèmes dynamiques.

Dans le chapitre II, après un aperçu historique sur les réseaux de neurones, et le rappel de quelques théorèmes relatifs à leurs capacités d'approximation, il est décrit les différentes architectures des réseaux de neurones ainsi que leurs techniques d'apprentissage.

Le chapitre III présente les techniques de modélisation à base de réseaux de neurones et les différentes méthodes pour l'estimation des paramètres des modèles non linéaires. Il est montré que le recours à des algorithmes d'estimation (d'apprentissage) qui recherchent une solution suivant une procédure itérative est indispensable.

Le chapitre IV aborde la commande prédictive des processus non linéaires. Après une description des différentes stratégies de commande basées sur les réseaux

de neurones, il est mis l'accent sur des techniques de commande prédictive généralisée non linéaire basées sur les réseaux de neurones artificiels. Les performances de ces stratégies de commande sont vérifiées à travers un exemple illustratif de système dynamique non linéaire.

Enfin, le chapitre V porte sur la résolution des problèmes de limitation de l'espace de travail d'une plateforme mobile d'un simulateur de conduite. Il est présenté dans ce chapitre une description des différentes architectures des simulateurs de conduite et les techniques de restitution des mouvements. Aussi, le modèle de la plateforme du simulateur de conduite étudié est décrit. Une étude par simulation, basée sur l'exploitation des données réelles du véhicule est effectuée.

La conclusion générale regroupe un ensemble de remarques relatives à la commande prédictive généralisée linéaire, la modélisation et la commande prédictive des processus non linéaires, l'efficacité de la GPC dans la commande des plateformes mobiles et les travaux en perspective.

Chapitre I COMMANDE PRÉDICTIVE GÉNÉRALISÉE LINÉAIRE

I.1. Historique

Les techniques de commande prédictive (MPC) *(Model Prédictive Control)* constituent des outils puissants pour affronter le problème de commande avec restrictions. La commande prédictive n'a connu un réel essor que depuis le milieu des années 80, grâce aux travaux de D.W. Clarke et de son équipe à Oxford.

Toutefois cette technique de commande, que l'on peut rattacher à la famille des commandes prédictives par modèle suscite un intérêt dans le domaine industriel depuis la fin des années 70. En effet en 1978, J. Richalet et al. publient les premiers résultats obtenus dans des applications industrielles. En 1982, R. M. C. De Keyser, puis en 1984 B. R. Ydstie proposent leur propre approche et leur apport à ce type de technique. C'est en 1985 que D. W. Clarke et al. présente la première version de la commande prédictive généralisée *(GPC).* Il faut attendre 1987, pour voir publier les premiers résultats obtenus par J. Richalet et al. sur des systèmes électromécaniques rapides, tels que des commandes d'axes d'un robot. Les divers algorithmes, membres de la famille des MPC (appelée également LRPC : long range predictive control), diffèrent seulement par le type de modèle à utiliser pour représenter le processus et les perturbations.

I.2. Introduction

Le terme commande prédictive ne désigne pas une stratégie de commande spécifique mais un ensemble de méthodes de l'automatique qui utilisent explicitement un modèle du processus à commander, afin d'obtenir le signal de commande par la minimisation d'une fonction de coût.

Ces méthodes donnent des correcteurs linéaires qui ont pratiquement tous la même structure et qui se basent sur les idées suivantes :

- utilisation d'un modèle du système pour prévoir les sorties à des instants futurs (notion d'horizon de prédiction) ;

- calcul des actions optimales de commande basé sur la minimisation d'une fonction de coût dans le futur (notion d'horizon de commande) ;

- à chaque instant d'échantillonnage, l'horizon de prédiction est déplacé vers le futur, et seule la première des commandes calculées est effectivement appliquée au système (notion d'horizon fuyant).

La commande prédictive présente un certain nombre d'avantages, par rapport aux autres méthodes, parmi lesquels on trouve :

- son principe intuitif et le réglage relativement facile de ses paramètres la rendent accessible aux personnes avec des connaissances limitées en automatique ;

- elle peut être utilisée pour commander une grande variété de processus, ceux avec des dynamiques simples à ceux plus complexes, par exemple les systèmes à grand retard, à phases non minimales ou instables ;

- elle est capable intrinsèquement de compenser les retards ou les temps morts ;

- le correcteur obtenu est une loi de commande linéaire facile à implémenter et qui requiert peu de temps de calcul ;

- le traitement de contraintes sur le système à commander peut être inclus systématiquement dans la définition du correcteur;

- elle est très utile lorsque les consignes à suivre sont connues à l'avance.

I.3. Philosophie de la commande prédictive

La stratégie de la commande prédictive est très similaire à la stratégie utilisée pour la conduite automobile. Le conducteur connaît la trajectoire de référence désirée (le tracé de la route) sur un horizon de commande fini (celui du son champ visuel), et en prenant en compte les caractéristiques de la voiture (modèle mental du comportement du véhicule), il décide quelles actions (accélérer, freiner ou tourner le volant) il faut réaliser afin de suivre la trajectoire désirée. Seule la première action de conduite est exécutée à chaque instant, et la procédure est répétée à nouveau pour les prochaines actions (Fig.I.1).

Fig.I.1 : Comportement naturel d'un conducteur au volant

Cette conception consiste à prendre en compte, à l'instant présent, le comportement futur, en utilisant explicitement un modèle numérique du système afin de prédire la

sortie dans le futur sur un horizon fini. Cependant, il n'existe pas une stratégie unique mais plutôt tout un ensemble de méthodes de commande prédictive, assez similaires, bâties autour de principes communs, mais présentant néanmoins quelques différences dans l'interprétation des concepts clés.

Une des richesses de ces méthodes provient du fait que, pour une consigne connue ou pré calculée (au moins sur un certain horizon), il est ainsi possible d'exploiter pleinement les informations de trajectoires prédéfinies situées dans le futur, puisque le but de la stratégie prédictive est de faire coïncider la sortie du processus avec cette consigne dans le futur, sur un horizon fini, en accord avec le diagramme temporel de la Fig.I.2, c'est pourquoi cette méthode apparaît tout indiquée dans les problèmes de poursuite et plus spécialement de suivi de trajectoire. C'est le cas de nombreux servomécanismes et notamment de la commande d'axes en machine outil ou en robotique, domaines où les trajectoires à suivre sont parfaitement connues.

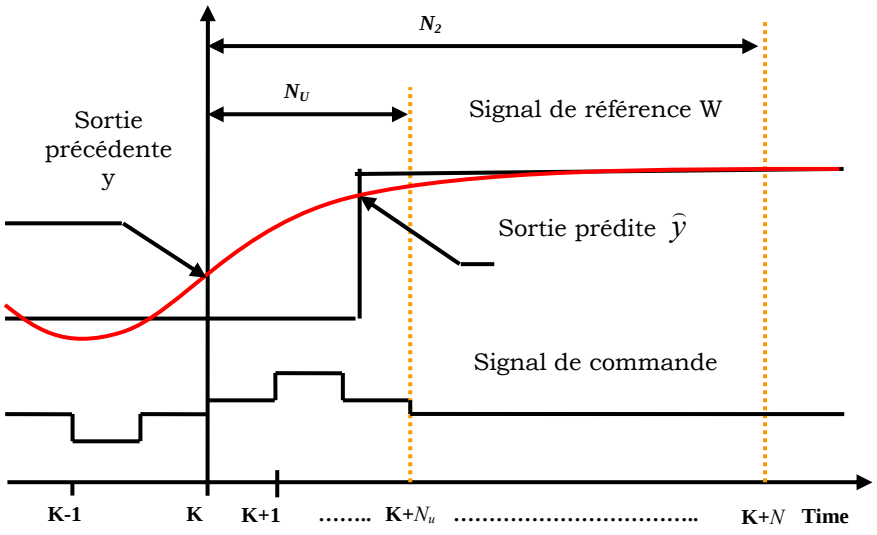


Fig.I.2 : Schéma de principe de la stratégie de la commande prédictive

Un schéma bloc simple pouvant caractériser une commande prédictive à base de modèle est représenté ci-dessous (Fig.I.3):

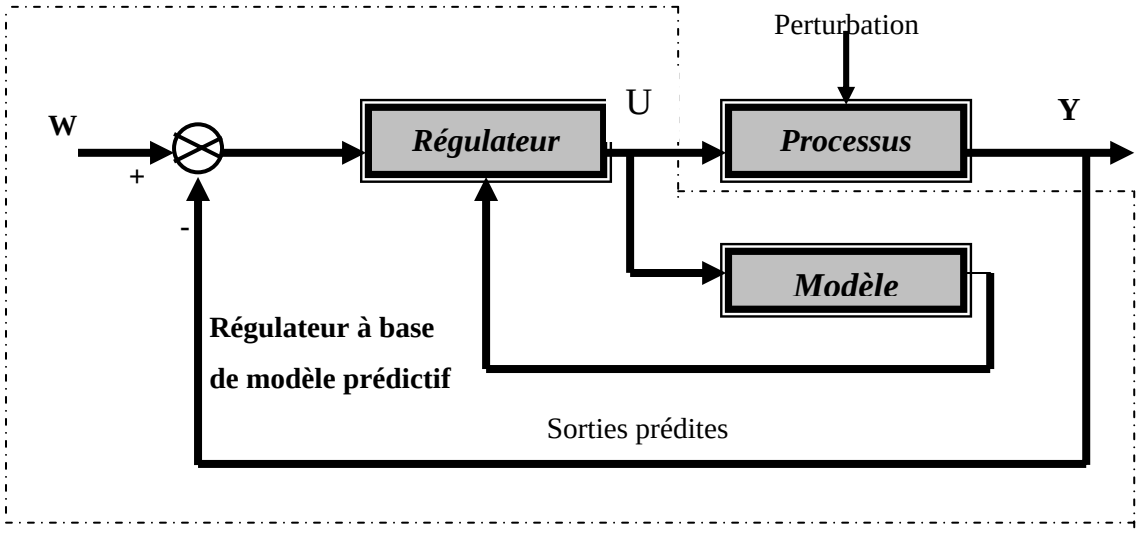


Fig.I.3 : Schéma de principe d'une commande prédictive à base de modèle

I.4. Commande prédictive généralisée

Développée par Clarke et al en 1987, la commande prédictive généralisée (GPC) est devenu l'un des algorithmes de commande prédictive les plus répandus.

Cette partie du chapitre présente la procédure pour obtenir une loi de commande GPC et ses principales caractéristiques. L'idée de base de la GPC est de calculer une séquence de commandes futures de telle façon qu'une fonction de coût à plusieurs composantes soit minimale sur un certain horizon de prédiction. L'indice à optimiser est une fonction quadratique qui mesure la distance entre la sortie prédite du système et une séquence de référence, plus une fonction quadratique qui mesure l'effort de commande.

Les spécificités de la commande GPC sont l'existence d'une solution optimale analytique, le fait qu'elle soit compatible avec des systèmes instables ou à phase non minimale, et enfin la notion de l'horizon de commande et d'incréments de commande.

I.4.1. Modélisation du système

Toute commande prédictive nécessite la connaissance d'un modèle afin de prédire le comportement futur du système. Dans la commande prédictive généralisée, le modèle utilisé est le modèle *CARIMA* (Controlled Autoregressive Integrated Moving Average), de la forme :

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) + C(q^{-1})\varepsilon(t)/\Delta(q^{-1}) \qquad \textbf{(I.1)}$$

Avec $y(t)$: sortie du système ;

$u(t)$: commande appliquée à l'entrée;

q^{-1} : opérateur retard ;

$\varepsilon(t)$: séquence aléatoire centrée non corrélée avec l'entrée.

L'introduction de l'opérateur différence $\Delta(q^{-1}) = 1 - q^{-1}$ dans le modèle de bruit assure une action intégrale dans le correcteur et permet d'annuler toute erreur statique vis-à-vis d'une entrée ou d'une perturbation en échelon. L'utilisation de ce modèle de perturbation est en fait une conséquence de la présence de perturbations de charge en échelon dans de nombreux processus industriels. Le modèle CARIMA est représenté ci-après (Fig.I.4).

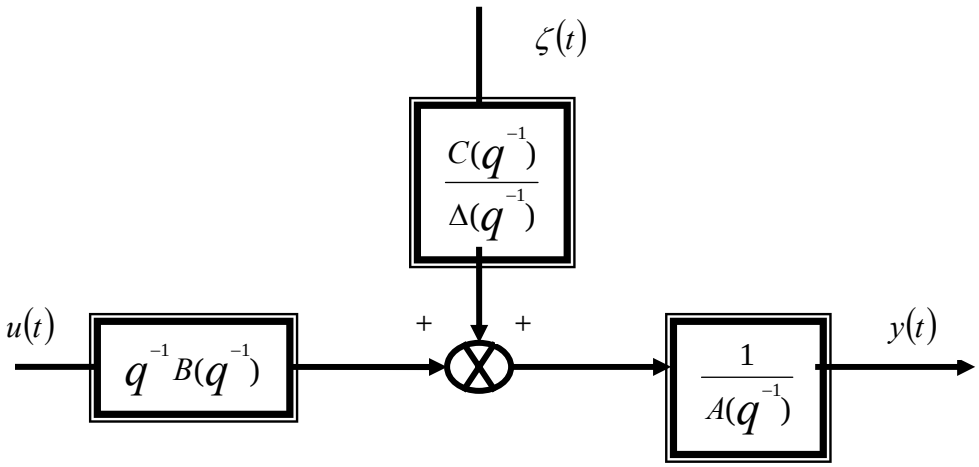


Fig.I.4. : Modèle CARIMA

Les polynômes $A(q^{-1})$, $B(q^{-1})$ et $C(q^{-1})$ sont définis par :

$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_{na} q^{-na}$$
$$B(q^{-1}) = b_0 + b_1 q^{-1} + \ldots + b_{nb} q^{-nb}$$
$$C(q^{-1}) = 1 + c_1 q^{-1} + \ldots + c_{nc} q^{-nc}$$

I.4.2. Fonction de coût

L'objectif du correcteur GPC est de minimiser un critère quadratique portant sur les erreurs futures entre les prédictions de la sortie et les consignes futures avec un terme de pondération sur les incréments de commande.

$$J = \sum_{j=N_1}^{N_2} (w(t+j) - \hat{y}(t+j))^2 + \lambda \sum_{j=1}^{N_u} \Delta u(t+j-1)^2 \qquad \textbf{(I.2)}$$

Sous l'hypothèse : $\Delta u(t+j) = 0 \ \forall \ j \geq N_u$ **(I.3)**

Avec :

$w(t+j)$: consigne;

$\hat{y}(t+j)$: sortie prédite;

$\Delta u(t+j-1)$: incrément de commande;

L'équation I.3 signifie que lorsque le pas de prédiction j atteint la valeur fixée pour l'horizon de commande N_u, la variation de commande s'annule et donc la commande future va se stabiliser.

Le critère nécessite la définition de quatre paramètres de réglage :

- N_1 : horizon de prédiction minimal ;
- N_2 : horizon de prédiction maximal ;
- N_u : horizon de commande ;
- λ : facteur de pondération sur la commande.

Ce critère comprend donc un terme quadratique sur l'erreur et sur l'incrément de commande. Sa minimisation analytique fournit la séquence de commandes futures dont seule la première sera effectivement appliquée.

Remarques

- L'aspect incrémental du modèle se retrouve dans le critère par l'intermédiaire de Δu ;

- Le coefficient λ permet de donner plus ou moins de poids à la commande par rapport à la sortie, de façon à assurer la convergence lorsque le système de départ présente un risque d'instabilité.

I.4.3. Calcul des prédictions de la sortie

I.4.3.1. Prédicteur optimal

La méthodologie prédictive requiert la définition d'un prédicteur optimal à *j*-pas qui permet d'anticiper le comportement du processus dans le futur sur un horizon fini. Pour cela, à partir de la forme du modèle Eq. I.1, nous élaborons la sortie estimée à l'instant $t+j$, connaissant la sortie à l'instant t. La sortie prédite $y(t+j/t)$ est décomposée de façon classique en réponse libre et en réponse forcée, incluant une

forme polynomiale pour mener à bien la synthèse polynomiale finale, sous la forme :

$$\underbrace{\hat{y}(t+j) = F_j(q^{-1})y(t) + H_j(q^{-1})\Delta u(t-1)}_{réponse\ \ libre} + \underbrace{G_j(q^{-1})\Delta u(t+j-1) + J_j(q^{-1})\xi(t+j)}_{réponse\ \ forcée} \qquad \textbf{(I.4)}$$

Avec G_j représentant le futur, F_j, H_j, correspondant respectivement au présent et au passé, J_j lié aux perturbations.

Le premier terme de Eq. (I.4) représentant la réponse libre est dû aux incréments de commandes passées, le deuxième terme représentant la réponse forcée correspond aux incréments de commandes futures et présente, et à l'influence des perturbations.

L'équation du modèle CARIMA Eq. (I.1), combinée avec celle du prédicteur, Eq. (I.4), fournit le système d'équations ci-dessous :

$$A(q^{-1})\Delta(q^{-1})y(t+j) = B(q^{-1})\Delta u(t+j-1) + \xi(t+j)$$

$$(1-q^{-j}F_j(q^{-1}))y(t+j) = (G_j(q^{-1}) + q^{-j}H_j(q^{-1}))\Delta u(t+j-1) + J_j(q^{-1})\xi(t+j)$$

L'équivalence des fonctions de transfert donne alors les deux équations suivantes :

$$\Delta(q^{-1})A(q^{-1})J_j(q^{-1}) + q^{-j}F_j(q^{-1}) = 1$$

$$G_j(q^{-1}) + q^{-j}H_j(q^{-1}) = B(q^{-1})J_j(q^{-1}) \qquad \textbf{(I.5)}$$

La première équation est une équation diophantine, se résolvant de façon récursive, et donnant des solutions explicites et claires, car les polynômes $\Delta(q^{-1})A(q^{-1})$ et q^{-j} sont premiers entre eux.

En supposant que la meilleure prédiction du terme lié aux perturbations est nulle (le cas du bruit blanc centré), le prédicteur optimal est défini de façon unique, dès que les polynômes F_j, G_j, H_j et J_j sont connus, par la relation **:**

$$\hat{y}(t+j) = F_j(q^{-1})y(t) + G_j(q^{-1})\Delta u(t+j-1) + H_j(q^{-1})\Delta u(t-1) \qquad \textbf{(I.6)}$$

$$deg[J_j(q^{-1})] = deg[G_j(q^{-1})] = j\text{-}1$$

$$deg[F_j(q^{-1})] = deg[A(q^{-1})]$$

$$deg[H_j(q^{-1})] = deg[B(q^{-1})]\text{-}1$$

I.4.3.2. Prédicteur optimal sous forme matricielle

L'équation I.6 donnant le prédicteur optimal est utilisée dans le critère Eq. I.2 entre les horizons N_1 et N_2 .

Pour simplifier les notations, il est possible d'utiliser une représentation matricielle de ce prédicteur.

Posons :

$$if(q^{-1}) = \left[F_{N_1}(q^{-1}),\ldots,F_{N_2}(q^{-1})\right]'$$

$$ih(q^{-1}) = \left[H_{N_1}(q^{-1}),\ldots,H_{N_2}(q^{-1})\right]'$$

$$\tilde{u} = \left[\Delta u(t),\ldots,\Delta u(t+N_u-1)\right]'$$

$$\hat{y} = \left[\hat{y}(t+N_1),\ldots,\hat{y}(t+N_2)\right]'$$

Avec ces notations, le prédicteur optimal à *j*-pas peut s'écrire sous la forme matricielle suivante :

$$\hat{y} = G\tilde{u} + if(q^{-1})y(t) + ih(q^{-1})\Delta u(t-1) \qquad \textbf{(I.8)}$$

$$G = \begin{bmatrix} g_{N_1}^{N_1} & g_{N1-1_1}^{N_1} & \cdots & \cdots \\ g_{N_1+1}^{N_1+1} & g_{N_1}^{N_1+1} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ g_{N_2}^{N_2} & g_{N_2-1}^{N_2} & \cdots & g_{N_2-N_U+1}^{N_2} \end{bmatrix} \quad \textbf{(I.7)}$$

La matrice G formée à partir des coefficients des polynômes $\{ g_i^j \}$ correspondant aux valeurs des coefficients $\{ g_i \}$ de la réponse indicielle du modèle.

I.4.4. Détermination de la solution optimale

Avant de formuler l'expression de la loi de commande de la méthode GPC, nous allons d'une part expliciter le principe sur lequel se base la loi de commande ,et d'autre part expliciter le critère de performance que celle –ci est appelée à satisfaire .

I.4.4.1. Principe de la loi de commande

Soit une séquence de référence (ou consigne) $w(t+j)$ $(j = 1...N_2)$. L'objectif de la loi de commande GPC, est de calculer à l'instant courant t (c'est-à-dire à chaque instant d'échantillonnage), une grandeur de commande $u(t)$ dont le but sera de rapprocher la sortie future $y(t+j)$ autant que possible de la séquence de consigne $w(t+j)$. Ceci est réalisé en utilisant une commande à horizon fuyant. A chaque instant d'échantillonnage, on doit exécuter les étapes suivantes:

***Etape 1* :** calcul ou lecture de la trajectoire de référence (ou consigne) $w(t+j)$;

***Etape 2* :** détermination des prédictions de la sortie du système ;

***Etape 3* :** calcul de la séquence de commandes futures $\Delta u(t+j-1)$ $(j = N_1 \cdots N_2)$.

***Etape 4* :** à partir des commandes futures, prendre la quantité $u(t)$ et l'injecter au système à contrôler.

I.4.4.2. Loi de commande

Le critère quadratique Eq.I.2 peut se combiner avec la relation Eq. I.8 pour obtenir l'expression matricielle de ce critère :

$$J = \left[G\tilde{u} + if(q^{-1})y(t) + ih(q^{-1})\Delta u(t-1) - w \right]' \left[G\tilde{u} + if(q^{-1})y(t) + ih(q^{-1})\Delta u(t-1) - w \right] + \lambda \tilde{u}'\tilde{u} \quad \textbf{(I.9)}$$

La séquence de la commande optimale s'obtient enfin par la minimisation analytique de ce critère:

$$\tilde{u}_{op} = -M \left[if(q^{-1})y(t) + ih(q^{-1})\Delta u(t-1) - w \right] \quad \textbf{(I.10)}$$

Où :

$$M = \left[G'G + \lambda I \right]^{-1} G' \quad \textbf{(I.11)}$$

De façon classique en commande prédictive, seule la première valeur de la séquence est appliquée au système, en accord avec la stratégie de l'horizon fuyant, l'ensemble de la procédure étant effectué de nouveau à la période d'échantillonnage suivante.

$$u_{op}(t) = u_{op}(t-1) - m_1 \left[if(q^{-1})y(t) + ih(q^{-1})\Delta u(t-1) - w \right] \quad \textbf{(I.12)}$$

avec m_1 première ligne de la matrice M .

Comme le critère utilisé dans la GPC est quadratique, alors les techniques de la programmation quadratique (QP) sont souhaitables pour la résolution des problèmes des contraintes sur la commande, la sortie, ou les incréments de la commande.

$$U_{\min} \leq U \leq U_{\max} \quad \textbf{(I.13)}$$

$$Y_{\min} \leq Y \leq Y_{\max} \quad \textbf{(I.14)}$$

$$\Delta U_{\min} \leq \Delta U \leq \Delta U_{\max} \quad \textbf{(I.15)}$$

I.4.5. Structure RST du régulateur

La structure RST polynomiale est introduite afin d'obtenir une relation entre la sortie $y(t)$, la commande $u(t)$ et la consigne $w(t)$. À partir de la relation Eq. I.12, il vient :

$$\Delta u_{op}(t)\left[1 + m_1 ih(q^{-1})q^{-1}\right] = -m_1 if(q^{-1})y(t) + m_1\left[q^{N_1}\ldots q^{N_2}\right]' w(t) \qquad \textbf{(I.16)}$$

Cette relation doit correspondre d'après la fig.I.4 à l'équation :

$$S(q^{-1})\Delta(q^{-1})u(t) = -R(q^{-1})y(t) + T(q^{-1})w(t) \qquad \textbf{(I.17)}$$

Ce qui fournit par identification les trois polynômes R, S et T constituant le régulateur linéaire équivalent :

$$\begin{aligned} S(q^{-1}) &= 1 + m_1 ih(q^{-1})q^{-1} \\ R(q^{-1}) &= m_1 if(q^{-1}) \end{aligned} \qquad \textbf{(I.18)}$$

$$T(q) = m_1\left[q^{N_1}\ldots q^{N_2}\right]'$$

Avec :

$deg\left[S(q^{-1})\right] = \deg\left[B(q^{-1})\right]$, $deg\left[R(q^{-1}\right] = \deg\left[A(q^{-1})\right]$, $deg\left[T(q^{-1})\right] = N_2$

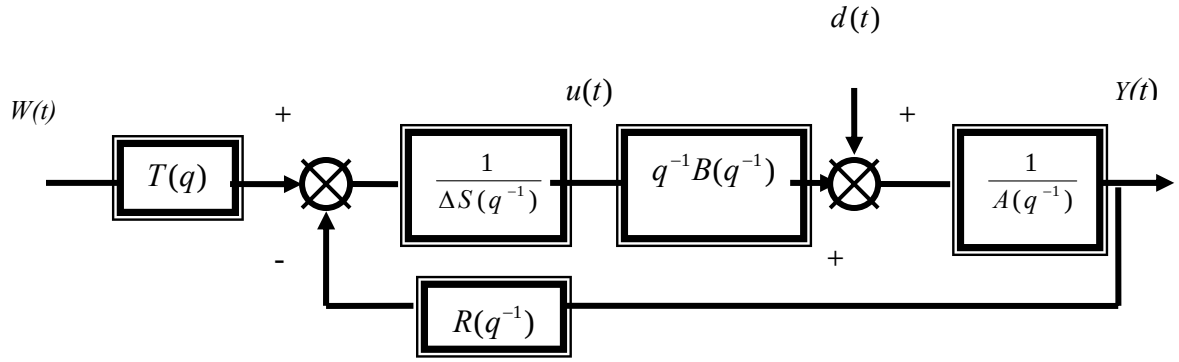


Fig.I.4. : Structure RST de la commande prédictive généralisée

Plusieurs remarques peuvent alors être formulées. Tout d'abord, le polynôme $T(q)$ renferme la structure non causale (puissances positives de q) inhérente à la commande prédictive, créant ainsi l'effet anticipatif désiré.

Ensuite, l'intérêt qui se dégage de la représentation RST est que finalement la boucle temps réel consomme moins de temps de calcul, puisque la commande appliquée au système se calcule par une simple équation aux différences Eq. I.12.
Les trois polynômes R, S, T sont en effet élaborés hors ligne et définis de façon unique, une fois que les quatre paramètres de réglage N_1, N_2 , N_u et λ sont choisis.
Un autre intérêt majeur de cette structure RST concerne l'étude de la stabilité de la boucle corrigée, et donc la caractérisation de la stabilité de la commande prédictive élaborée, qui est désormais possible pour un jeu de paramètres du critère fixé.
Le polynôme caractéristique déterminant la position des pôles en boucle fermée, est donné par l'équation ci-dessous:

$$P(q^{-1}) = A(q^{-1})S(q^{-1})\Delta(q^{-1}) + q^{-1}B(q^{-1})R(q^{-1}) \qquad \textbf{(I.19)}$$

Avec cette représentation, il est possible de tester la stabilité avant l'implantation de la loi de commande sur le système réel.

I.4.6. Choix des paramètres de synthèse de réglage

La définition du critère quadratique Eq. I.2 a montré que l'utilisateur doit fixer quatre paramètres de réglage. Ce choix des paramètres s'avère cependant délicat pour une personne non spécialiste, car il n'existe pas de relations empiriques permettant de relier ces paramètres à des 'indicateurs' classiques en automatique, tels que les marges de stabilité ou la bande passante [Ramond et al 2001].

I.4.6.1. Règles de choix

Nous présentons ci-dessous quelques idées guidant le choix des paramètres de réglage, obtenues à partir de l'étude d'un grand nombre de systèmes [Boucher et al 1996] [Ramond et al 2001].

- **Choix de l'horizon minimal de prédiction** N_1

Le produit $N_1 T_e$ (T_e période d'échantillonnage) est choisi égal au retard pur du système. Ainsi, pour un système ne présentant pas de retard ou un retard mal connu ou variable, N_1 est choisi égal à 1.

- **Choix de l'horizon maximal de prédiction** N_2

N_2 est choisi de sorte que le produit $T_e N_2$ soit limité par la valeur du temps de réponse souhaité. En effet, augmenter la prédiction au delà du temps de réponse n'apporte aucune information supplémentaire et complexifie la résolution. Par ailleurs, plus N_2 est grand, plus le système corrigé est stable et lent.

- **Choix de l'horizon de prédiction sur la commande** N_u

N_u est égal au nombre des pôles (instables) ou mal amortis mais la valeur $N_u = 1$ est très souvent suffisante pour beaucoup d'applications relativement simples. Dans ce dernier cas, le calcul de la séquence de commandes futures se réduit au simple calcul du scalaire $\tilde{u}_{op}$. En effet, N_u fixe la dimension des matrices à inverser dans le calcul du régulateur. La valeur de l'horizon de commande ne doit en aucun cas être supérieure à celle de l'horizon maximal de prédiction.

- **Choix du facteur de pondération de la commande** λ

On peut interpréter le facteur de pondération λ comme 'l'équilibre de la balance'. En effet, si $\lambda = 0$, on minimise uniquement dans le critère quadratique,

Eq.1.2, la différence entre la consigne et la sortie prédite. Il peut donc en résulter une commande très forte pouvant faire diverger le processus réel.
D'autre part, si λ est très élevé, on pondère alors excessivement la commande qui n'est plus assez 'dynamique' pour obtenir le ralliement à la consigne.
Dans le cadre mono variable, partant donc de la constatation que plus le gain d'un système est grand, plus la commande doit être pondérée (c'est à dire plus λ est important, et vice et versa), il apparaît alors clairement que λ est lié au gain du système.

Une relation permettant de déterminer rapidement la valeur de λ apportant au système le maximum de stabilité est donnée ci-dessous :

$$\lambda_{op} = \text{trace}\ (G'G) \qquad \textbf{(I.20)}$$

où G est la matrice décrite par la relation Eq. 1.7.

Enfin, les quatre paramètres de réglage du contrôleur GPC sont à sélectionner, pour procurer au système un comportement désiré :

- $N_1 = \dfrac{retard\quad pur\quad du\quad système}{période\quad d'échantillonnage}$
- $N_2 \leq \dfrac{temps\quad de\quad réponse\quad du\quad système}{période\ d'échantillonnage}$
- $N_u = 1$
- $\lambda_{opt} = \text{trace}\,(G'G)$

I.4.7. Exemple illustratif

La commande prédictive généralisée sous contraintes est appliquée à un servomécanisme, constitué d'un moteur à courant continu, d'un engrenage, d'un arbre et d'une charge non spécifiée (Fig.I.5) [Alberto et al 2004].

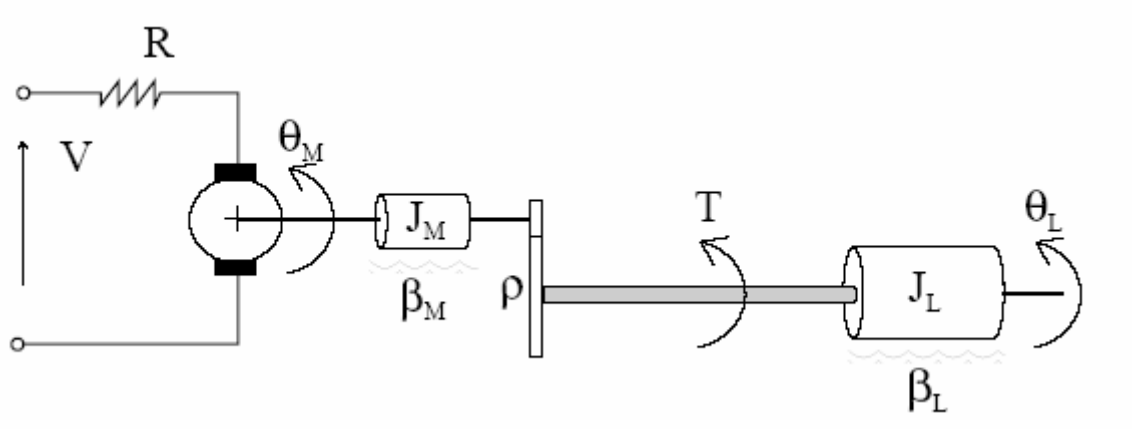


Fig.I.5. : Schéma du servomécanisme

Les spécifications techniques imposent des contraintes sur le couple de torsion de l'arbre T et sur la tension électrique U. Soient θ_M, θ_L respectivement la position angulaire du moteur et celle de la charge, et en considérant un vecteur d'état $x_p = [\theta_L \dot{\theta}_L \theta_M \dot{\theta}_M]'$, le modèle peut être décrit par la représentation d'état suivante :

$$\dot{x}_p = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_\theta}{J_L} & -\frac{\beta_L}{J_L} & \frac{k_\theta}{\rho J_L} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_\theta}{\rho J_M} & 0 & -\frac{k_\theta}{\rho^2 J_M} & -\frac{\beta_M + k_T^2 / R_M}{J_M} \end{bmatrix} x_p + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{k_T}{R J_M} \end{bmatrix} \mathrm{V}$$

$$\theta_L = [1 \quad 0 \quad 0 \quad 0] x_p$$

$$T = \begin{bmatrix} k_\theta & 0 & -\frac{k_\theta}{\rho} & 0 \end{bmatrix} x_p$$

Des contraintes sur Le couple de torsion T et la tension électrique U appliquée au moteur sont définies comme suit :

$|T| \leq 78.5398\ [Nm]$ et $|U| \leq 220\ [V]$

Ce modèle est transformée dans le domaine discret en utilisant une période d'échantillonnage $T_e = 0.1\ [s]$ et un bloqueur d'ordre zéro sur la tension U.

$$G(Z) = 1000 \frac{9.7929Z^3 - 2.1860Z^2 - 7.2663Z + 2.5556}{10Z^4 - 2.7282Z^3 - 3.5585Z^2 - 1.3029Z - 0.0853}$$

La structure du schéma de commande est donnée ci-après :

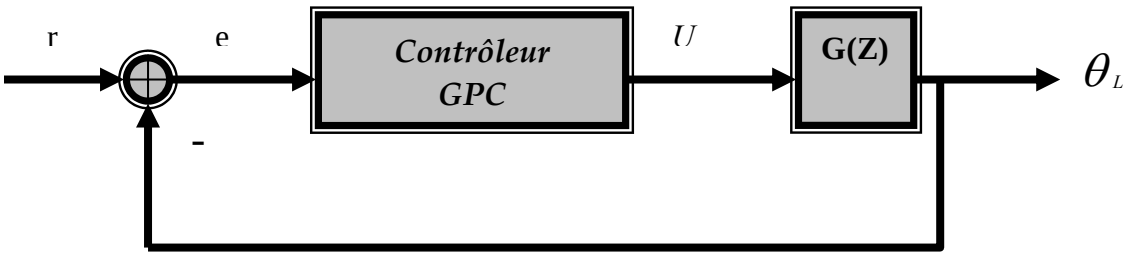


Fig.I.6. : Schéma de commande du système

r : le Signal de référence

e : l'erreur

U : la tension électrique

θ_L : la position angulaire de la charge

Au début, nous avons appliquée au système la commande prédictive généralisée sans contraintes. Les résultats de simulation du système en boucle fermée montrent que nous obtenons avec les valeurs des paramètres de synthèse $N_1 = 1$, $N_2 = 5$, $N_u = 2$ *et* $\lambda = 0.05$, une réponse rapide mais avec des inadmissibles valeurs de tension et de couple de torsion (Fig.I.7, 8 et 9) pour un signal de référence r = 32 degrés.

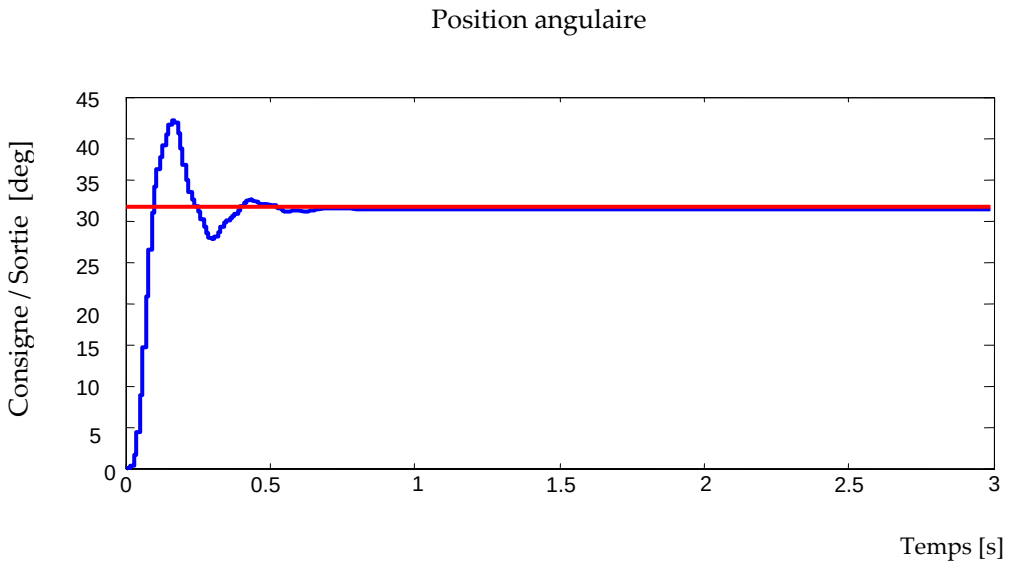


Fig.I.7. : Courbes des positions de la consigne et de la charge

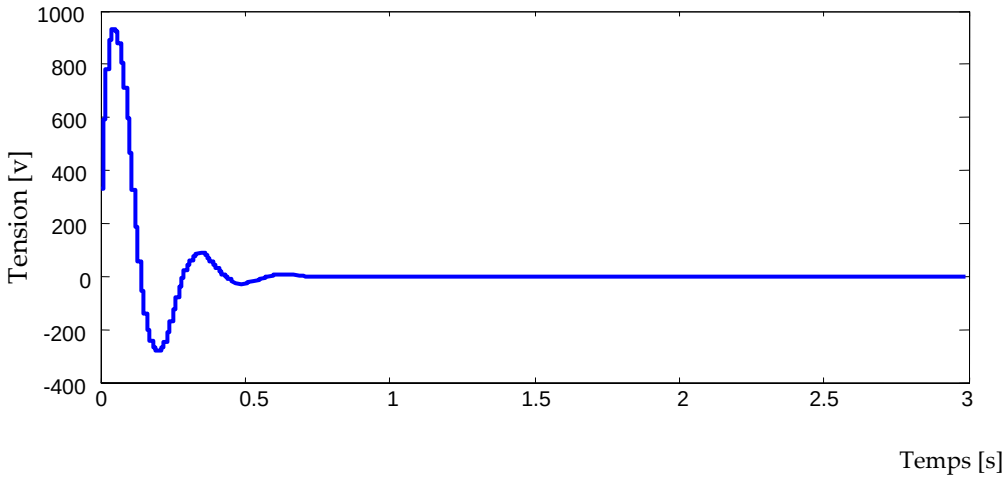


Fig.I.8. : Courbe de la tension électrique à appliquer au moteur

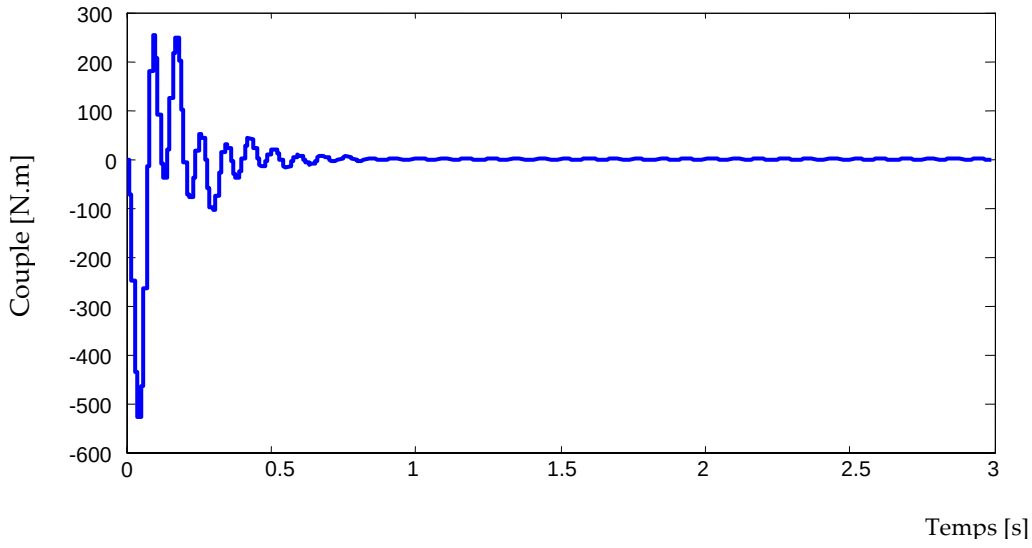


Fig.I.9. : Courbe du couple de torsion

Pour respecter les contraintes imposées sur la tension électrique et le couple, la commande prédictive généralisée sous contraintes est utilisée. Nous remarquons maintenant d'après les courbes des figures (I.10, 11 et 12) qu'avec les mêmes valeurs des paramètres de synthèse que la poursuite du signal de référence est meilleure (réduction du dépassement) et que les contraintes sont prises en compte.

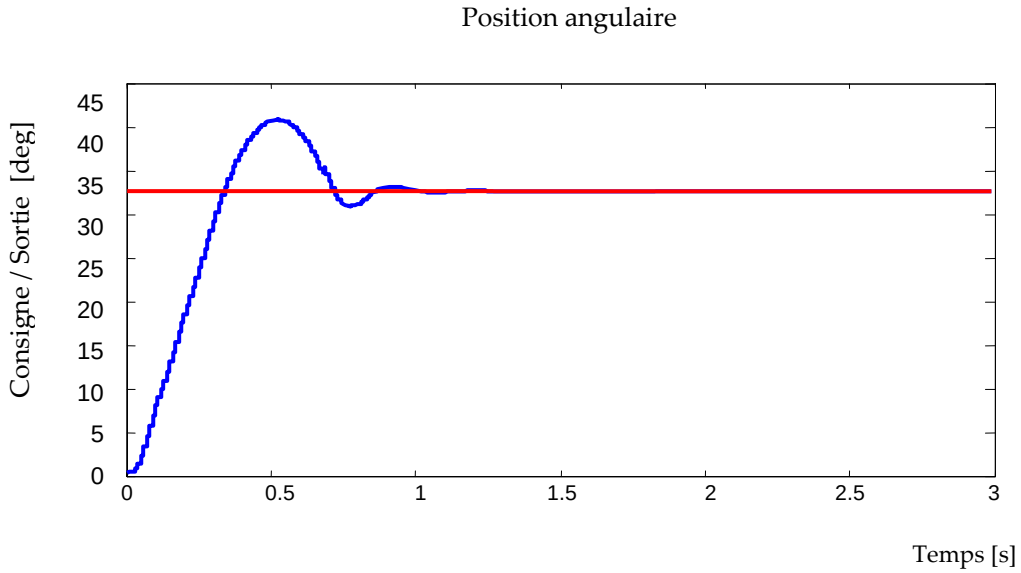


Fig.I.10. : Courbes des positions de la consigne et de la charge

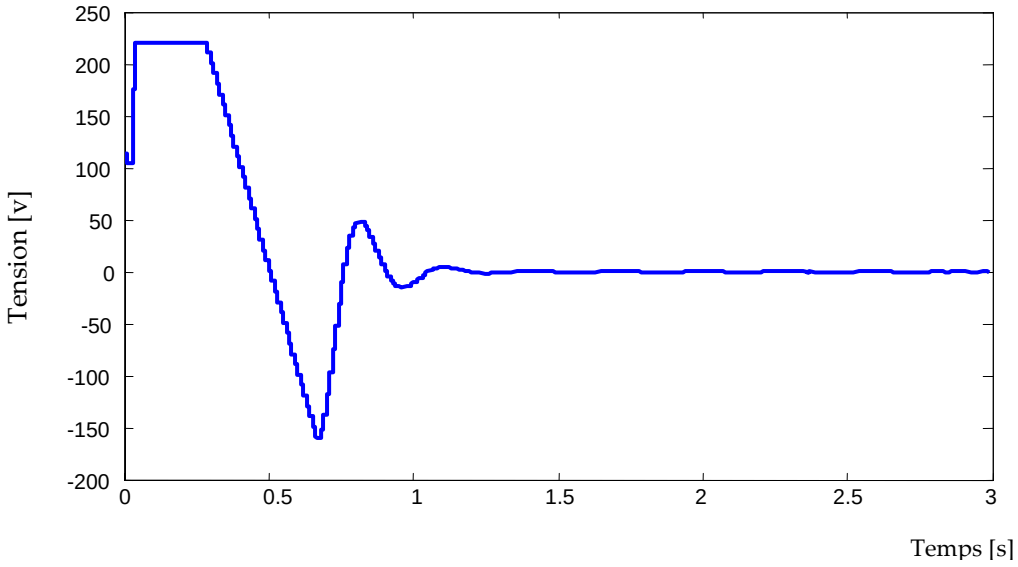


Fig.I.11. : Courbe de la tension électrique à appliquer au moteur

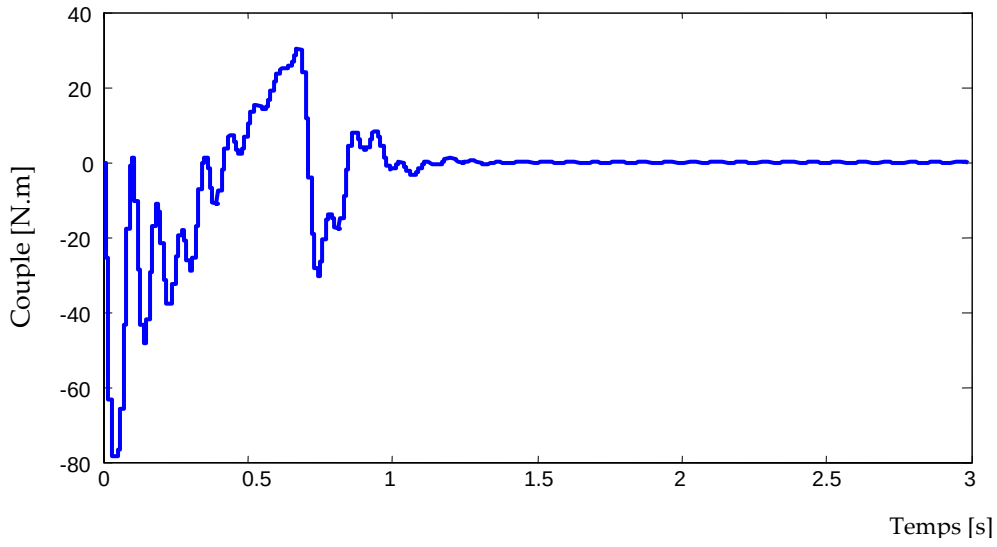


Fig.I.12. : Courbe du couple de torsion

I.5. Conclusion

Dans ce chapitre, nous avons présenté la philosophie de la commande prédictive généralisée ainsi que ses concepts de base et ses avantages qui s'avèrent prometteurs dans le domaine de la commande des systèmes.

La souplesse de la synthèse d'un contrôleur GPC, qui réside dans la sélection des paramètres de synthèse, pour agir sur la dynamique des systèmes en présence de contraintes sur les signaux d'entrée et de sortie, ont été illustrées à travers un exemple de servomécanisme.

Chapitre II RÉSEAUX DE NEURONES ARTIFICIELS

II.1. Introduction

Le développement, dans les années 40, des réseaux de neurones artificiels ou réseaux de neurones formels est issu d'une volonté de l'homme de comprendre et d'imiter les capacités du cerveau. Mémoire, apprentissage, intelligence, traitement parallèle massif d'informations et plasticité sont autant de qualités attribuées au cerveau, recherchées pour la synthèse de systèmes artificiels capables de remplacer l'homme dans la réalisation de tâches complexes.

Les réseaux de neurones artificiels doivent tout autant leur essor considérable récent à la biologie, qui en constitue une source d'inspiration, qu'aux sciences de l'ingénieur. L'intérêt des neurologues et des biologistes pour les réseaux de neurones artificiels est orienté vers la compréhension du cerveau humain à partir de l'élaboration et de l'étude de modèles artificiels complexes, réalistes et plausibles au sens biologique, qui tentent de reproduire certaines caractéristiques du cerveau.

Les ingénieurs et les informaticiens, en revanche, privilégient l'aspect calcul avec la construction de réseaux de neurones simplifiés, arborant une puissance de calcul élevée. Selon cette approche, les modèles neuronaux sont usuellement dépouillés et réduits à l'essentiel, l'efficacité primant sur la plausibilité biologique. Opérationnels très vite, les réseaux de neurones se sont heurtés à des questions fondamentales.
Après l'enthousiasme de la fin des années 80, les chercheurs et ingénieurs ont été confrontés à des problèmes que les statisticiens connaissent depuis toujours : la taille des bases d'exemples, la représentativité des données, la signification et l'interprétation des résultats et des estimations. Ces problèmes soulèvent l'insuffisance

des bases théoriques des pratiques heuristiques qui longtemps ont tenu lieu de méthodes pour le développement des techniques neuronales.
Les liens qui unissent la statistique et les réseaux de neurones sont forts, et l'intersection des deux disciplines importante. Le processus d'apprentissage d'un réseau est en effet un processus stochastique (aléatoire) qui dépend des propriétés statistiques de la distribution des exemples de la base d'apprentissage.
Un réseau de neurones artificiels est assimilé, dans le contexte de l'identification et du contrôle de processus, à un système ou une boîte noire comportant des entrées et des sorties dont la finalité est la modélisation d'un problème par apprentissage au moyen d'une base d'exemples. Le réseau, ignorant au départ, réalise un apprentissage à partir de ces exemples et bâtit un modèle spécifique à l'application.

Nous allons explicité dans ce chapitre : la nature, les principales architectures et les mécanismes d'apprentissage des réseaux de neurones artificiels.

II.2. Réseaux de neurones artificiels

Un réseau de neurones artificiels est un processeur parallèle de traitement d'informations distribuées, qui présente une propension naturelle à la mémorisation et à l'exploitation de connaissances relatives à l'environnement dans lequel il est immergé, connaissances acquises à partir de l'expérience.
Sa structure repose sur une interconnexion massive de cellules élémentaires de traitement d'information, appelées neurones formels, dont la représentation est un graphe dirigé [Hecht-Nielsen 90]. L'analogie entre un réseau de neurones formels et le cerveau est suscitée par deux remarques [Haykin 94] :

- un réseau de neurones artificiels acquiert la connaissance de son environnement par l'intermédiaire d'un apprentissage qui "simule" la plasticité du cerveau.
  En cours d'apprentissage, le réseau peut être amené à modifier sa structure en créant ou en supprimant des neurones ou des liaisons entre neurones.

- la connaissance acquise par un réseau de neurones est encodée par les forces ou Intensités évolutives des connexions établies entre neurones formels.
  Les forces des connexions liant les neurones réfèrent aux valeurs des poids ou coefficients synaptiques qui définissent les paramètres du modèle interne du réseau.

Il apparaît que l'intérêt des réseaux de neurones artificiels réside dans le parallélisme de leur structure, leur capacité d'adaptation ainsi que leur mémoire distribuée.
Il est important de citer également la capacité de généralisation des réseaux de neurones qui émerge de l'apprentissage. La capacité de généralisation désigne l'aptitude d'un réseau à présenter un comportement acceptable en réponse à des stimuli externes de son environnement non rencontrés lors de l'apprentissage (interpolation et extrapolation).

A la lumière des propriétés citées, il est possible de déterminer les caractéristiques des problèmes propices à une résolution par les réseaux de neurones formels [Davalo *et al.* 89] :

- le modèle du problème considéré est inconnu ou difficile à formaliser.
  Un ensemble d'exemples, composé d'entrées du problème auxquelles sont associées des solutions fournies par un expert, est néanmoins accessible.

- les données du problème sont entachées de bruit,

- le problème est de nature évolutive,

- le problème nécessite un traitement temps réel,

- il n'existe pas de solutions technologiques courantes au problème.

Les domaines d'application privilégiés, présentant les caractéristiques d'une résolution neuronale exposées ci-dessus, concernent le regroupement et la classification de données, le traitement du signal, la modélisation et l'identification de processus, le contrôle (surveillance) et la commande de processus, la prédiction et l'aide à la décision [Davalo *et al.* 89], [Fausett 94], [Taylor 93].
La tâche dévolue à un réseau de neurones est l'élaboration, par apprentissage, d'un modèle de connaissance de l'environnement dans lequel il est immergé. Le terme connaissance désigne au sens général une information mémorisée, utilisée par une personne ou une machine pour interpréter, prédire, et répondre adéquatement au monde environnant.

Dans un réseau de neurones, la représentation de la connaissance est déterminée par les valeurs de ses poids synaptiques (paramètres internes). Le problème de l'apprentissage d'un réseau est par conséquent celui de la construction, par ajustement de ses paramètres, d'une forme appropriée de représentation de la connaissance.
La connaissance mémorisée par un réseau de neurones artificiels résulte d'observations de l'environnement au moyen de capteurs.

Généralement, les observations sont intrinsèquement entachées de bruit en raison des imperfections des capteurs et de l'environnement lui-même.
Les observations de l'environnement recueillies constituent un réservoir d'informations à partir duquel des exemples d'apprentissage utilisés pour entraîner le réseau sont extraits. Un exemple d'apprentissage consiste, au sens général, en un couple d'entrée-sortie comprenant un stimulus de l'environnement auquel est associée une réponse désirée traduisant le comportement souhaité du réseau.

En résumé, l'apprentissage désigne un processus d'adaptation des paramètres internes (connexions) d'un réseau de neurones formels. La finalité de ce processus est l'obtention d'un état stable du réseau correspondant à un modèle de connaissance de son environnement dont une base d'exemples fournit une description.

Deux facteurs contribuent à la forme de représentation de la connaissance mémorisée dans le réseau au cours de l'apprentissage :

- le modèle des neurones et le schéma de leur interconnexion, prédéfinis en phase de conception initiale du réseau. La conception d'un réseau s'avérant tributaire de son application, cette donnée structurelle reflète en quelque sorte une part de connaissance innée de l'environnement intégrée au réseau.

- le mécanisme d'apprentissage mis en oeuvre pour assurer la maturation du réseau. Il détermine la part de connaissance que le réseau acquiert de l'environnement par l'expérience.

II.3. Architectures des réseaux de neurones

II.3.1. Définition

Aujourd'hui de nombreux termes sont utilisés dans la littérature pour désigner le domaine des réseaux de neurones artificiels.

Il n'y a pas de définition universellement acceptée de réseau de neurones. On considère généralement qu'un réseau de neurones est constitué d'un grand ensemble d'unités (ou neurones), ayant chacune une petite mémoire locale. Ces unités sont reliées par des canaux de communication (les connexions, aussi appelées synapses d'après le terme biologique correspondant), qui transportent des données numériques. Les unités peuvent uniquement agir sur leurs données locales et sur les entrées qu'elles reçoivent par leurs connexions. Nous distinguons deux types de réseaux : les réseaux non bouclés et les réseaux bouclés.

II.3.2. Réseaux non bouclés

Les neurones peuvent être ordonnés d'une façon telle qu'il n'y a aucune connexion "vers l'arrière" (terme anglais : "Feed-Forward" neural network).

Un réseau de neurones non bouclé est donc représenté graphiquement par un ensemble de neurones connectés entre eux, l'information circulant des entrées vers les sorties sans retour en arrière : si l'on représente le réseau comme un graphe dont les nœuds sont les neurones et les arêtes les connexions entre ceux-ci, le graphe d'un réseau non bouclé est *acyclique* : si l'on se déplace dans le réseau, à partir d'un neurone quelconque, en suivant les connexions, on ne peut pas revenir au neurone de départ (Fig.II.1). Les neurones qui effectuent le dernier calcul de la composition de fonctions sont les *neurones de sortie* ; ceux qui effectuent des calculs intermédiaires sont les *neurones cachés.*

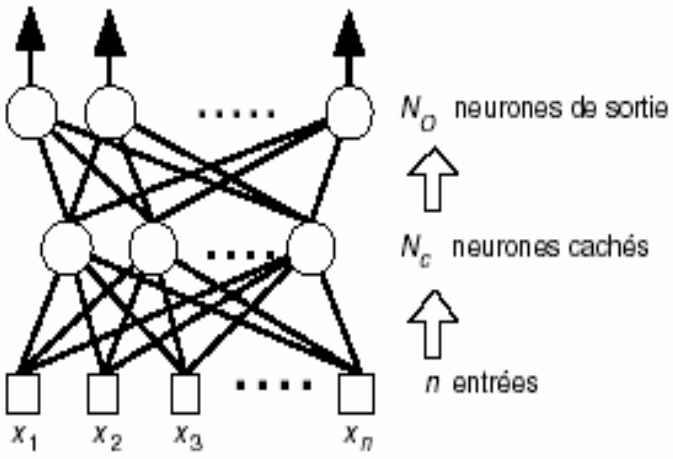


Fig.II.1: Exemple d'un réseau à couche non bouclé

II.3.3. Réseaux bouclés

A l'opposé des réseaux non bouclés, les réseaux bouclés sont le siège de contre réaction synchrone ou asynchrone en fonction du temps (Fig.II.2).

Ils sont particulièrement adaptés pour construire des réseaux de type Hoppfield ou Boltzman avec des procédures d'apprentissage non supervisés.

Pour ces réseaux le temps intervient et le comportement des cellules du réseau est régi en général par des équations différentielles non linéaires. Pour des conditions initiales données qui correspondent à l'exemple à mémoriser, le réseau évolue au cours du temps pour atteindre un état d'équilibre stable ou instable. Comme en automatique des systèmes non linéaires, un état instable se manifeste par des cycles d'oscillations autour d'un état donné.

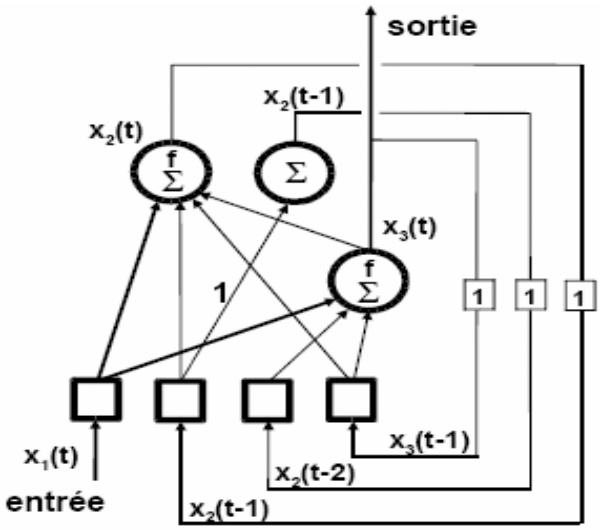


Fig.II.2: Exemple d'un réseau bouclé

II.4. Propriété fondamentale des réseaux de neurones

La propriété fondamentale des réseaux de neurones est l'approximation parcimonieuse. Cette expression traduit deux propriétés distinctes : d'une part, les réseaux de neurones sont des approximateurs universels, et d'autre part, une approximation à l'aide de réseau de neurones nécessite, en général, moins de paramètres ajustables que les approximateurs usuels.

La propriété d'approximation universelle [Cybenko 89] peut s'énoncer de la façon suivante :

Toute fonction bornée suffisamment régulière peut être approchée uniformément, avec une précision arbitraire, dans un domaine fini de l'espace de ses variables, par un réseau de neurones comportant une couche de neurones cachés en nombre fini, possédant tous la même fonction d'activation, et un neurone de sortie linéaire.

Lorsque l'on veut modéliser un processus à partir des données, on cherche toujours à obtenir les résultats les plus satisfaisants possibles avec un nombre minimum de paramètres ajustables. Dans cette optique, [Hornik, 1994] a montré que : Si le résultat de l'approximation est une fonction non linéaire des paramètres ajustables, elle est plus parcimonieuse que si elle est une fonction linéaire des ces paramètres.

De plus, pour des réseaux de neurones à fonction d'activation sigmoïdale, l'erreur commise dans l'approximation varie comme l'inverse du nombre de neurones cachés, et elle est indépendante du nombre de variables de la fonction à approcher.

Par conséquent, pour une précision donnée, donc pour un nombre de neurones cachés donné, le nombre de paramètres du réseau est proportionnel au nombre de variables de la fonction à approcher.
Ce résultat s'applique aux réseaux de neurones à fonction d'activation sigmoïdale puisque la sortie de ces neurones n'est pas linéaire par rapports aux poids synaptiques.
La spécificité des réseaux de neurones réside donc dans le caractère parcimonieux de l'approximation : à précision égale, les réseaux de neurones nécessitent moins de paramètres ajustables (les poids des connexions) que les approximateurs universels couramment utilisés ; plus précisément, le nombre de poids varie linéairement avec le nombre de variables de la fonction à approcher, alors qu'il varie exponentiellement pour la plupart des autres approximateurs [Hornik et al., 1994].

Qualitativement, la propriété de parcimonie peut s'énoncer de la manière suivante : lorsque l'approximation est une combinaison *linéaire* de fonctions élémentaires fixées (des monômes par exemple, où des gaussiennes à centres et écarts-types fixes), on ne peut ajuster que les coefficients de la combinaison ; en revanche, lorsque l'approximation est une combinaison linéaire de fonctions non linéaires à paramètres ajustables (un perceptron multicouche par exemple), on ajuste à la fois les coefficients de la combinaison et la forme des fonctions que l'on combine. Ainsi, dans un perceptron multicouche, les poids de la première couche déterminent la forme de chacune des sigmoïdes réalisées par les neurones cachés, et les poids de la seconde couche déterminent une combinaison linéaire de ces fonctions. On conçoit facilement que cette souplesse supplémentaire, conférée par le fait que l'on ajuste la forme des fonctions que l'on superpose, permet d'utiliser un plus petit nombre de fonctions élémentaires, donc un plus petit nombre de paramètres ajustables. Nous allons voir ultérieurement pourquoi cette propriété de parcimonie est précieuse dans les applications industrielles.

II.5. Apprentissage des réseaux de neurones

II.5.1. Mécanismes d'apprentissage

L'apprentissage d'un réseau de neurones artificiels est induit par une procédure itérative d'ajustement ou d'adaptation de ses paramètres internes au moyen d'un processus de stimulation par l'environnement. En d'autres termes, le mécanisme d'apprentissage d'un réseau comprend la récurrence des phases suivantes :

- le réseau est stimulé par l'environnement,
- en réponse à cette stimulation, le réseau adapte son comportement,
- le réseau réagit alors différemment à l'environnement en fonction de la nouvelle expérience acquise consécutivement à la stimulation.

La procédure d'adaptation des paramètres internes d'un réseau est décrite par un algorithme d'apprentissage (annexe 1). Idéalement, un réseau acquiert davantage de connaissance à chaque itération de l'algorithme d'apprentissage. Celui-ci comprend un ensemble de règles destinées à la recherche d'une solution au problème de l'apprentissage que constitue l'assimilation de la connaissance.

L'algorithme d'apprentissage détermine le comportement du réseau. Il en existe de multiples formes qui se distinguent par la nature de la connaissance de l'environnement acquise par le réseau. Ainsi, le comportement d'un même réseau diffère selon l'algorithme d'apprentissage utilisé pour modifier ses paramètres.
Deux philosophies d'apprentissage coexistent, dépendant de l'information relative à l'environnement disponible : l'apprentissage supervisé et l'apprentissage non supervisé.

II.5.1.1. Apprentissage supervisé

L'apprentissage supervisé ("*Supervised Learning*") suppose l'existence d'un expert (ou éducateur) qui possède une connaissance innée de l'environnement.
Le rôle de l'expert est de fournir les informations relatives à l'environnement nécessaires à l'apprentissage du réseau, sous la forme d'un ensemble d'exemples composés de stimuli auxquels sont associées des réponses désirées (ou comportements souhaités).
Dans ce mode d'apprentissage, les réponses désirées fournies par l'expert décrivent la fonction du réseau. Le rôle de l'algorithme d'apprentissage est d'amener le réseau à remplir cette fonction.
Au cours de l'apprentissage, l'environnement soumet conjointement le réseau et l'expert à des stimuli. La sortie produite par le réseau, en réponse à un stimulus donné, est comparée à la réponse désirée fournie par l'expert. La différence entre la réponse désirée et la réponse du réseau est alors utilisée pour adapter les paramètres du réseau de façon à corriger son comportement (Fig.II.3). Ce processus est ainsi répété jusqu'à émulation de l'expert par le réseau de neurones artificiels.

Sous la forme décrite, l'apprentissage supervisé présente une forte contrainte opératoire : l'intervention d'un expert qui fournit précisément les réponses désirées. Une forme d'apprentissage supervisé moins contraignante consiste à instruire ou entraîner le réseau par tâtonnement en procédant par essais et erreurs. Le réseau est alors stimulé par l'environnement et ses réponses sont sanctionnées ou récompensées afin de l'inciter à adopter le bon comportement. Cette variante d'apprentissage supervisé est qualifiée d'apprentissage renforcé ("*Reinforcement Learning*") [Haykin 94] [Hertz *et al.* 91].

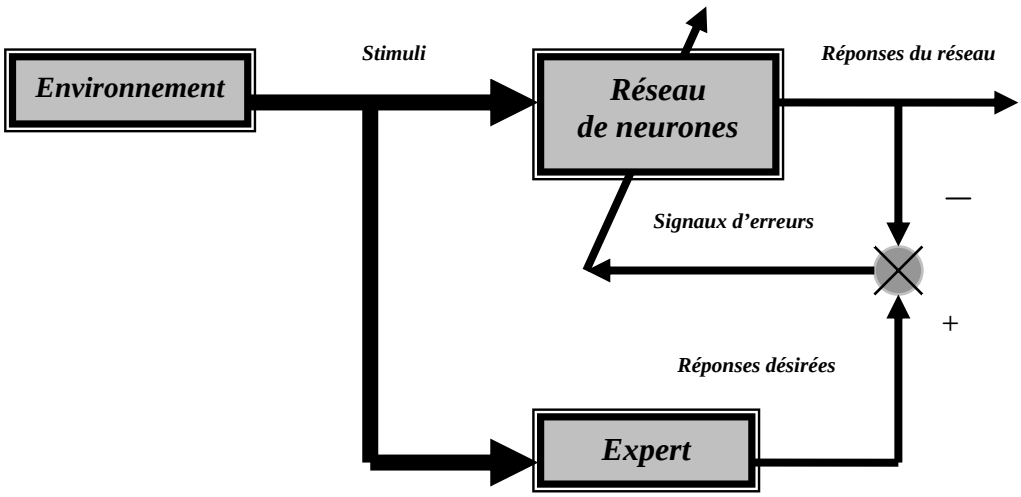


Fig.II.3: Illustration de l'apprentissage supervisé

II.5.1.2. Apprentissage non supervisé

Contrairement à l'apprentissage supervisé effectué sous contrôle d'un expert, l'apprentissage non supervisé ("*Unsupervised Learning*") est autodidacte. L'ensemble des exemples d'apprentissage ne comprend que des stimuli. Aucune réponse désirée n'est associée.

Par nature, cet apprentissage est destiné à l'élaboration d'une représentation interne de la connaissance de l'environnement, identifiant la structure statistique sous-jacente des stimuli sous une forme plus simple ou plus explicite. L'algorithme d'apprentissage

exploite pour ce faire une mesure prédéterminée de la qualité de représentation de la connaissance afin d'ajuster les paramètres du réseau. Ce type d'apprentissage est représenté schématiquement en Fig.II.4.

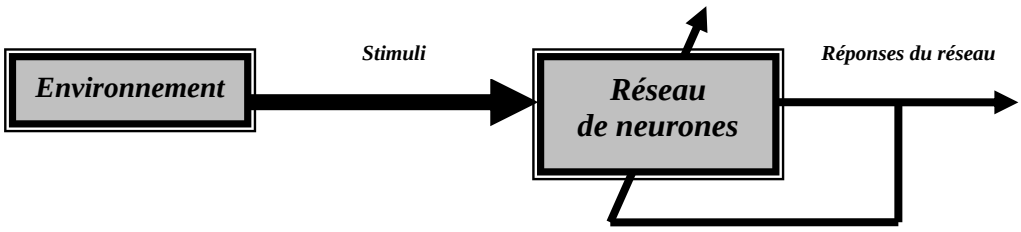


Fig.II.4: Illustration de l'apprentissage non supervisé

II.5.2. Apprentissage et adaptation

Si le terme adaptation, au sens de l'ajustement des paramètres internes d'un réseau de neurones artificiels, réfère à l'assimilation et la mémorisation de connaissances, il dénote également la capacité du réseau à modifier dynamiquement son comportement afin de répondre à de nouvelles attentes ou de nouvelles situations.

La nuance de sens accordée au terme adaptation dans le domaine des réseaux de neurones est liée à la prise en considération ou non du facteur temps dans le processus d'apprentissage. L'apprentissage, qu'il soit supervisé ou non supervisé, est en effet conventionnellement réalisé soit en ligne (temps réel), soit hors ligne (temps différé).
L'apprentissage hors ligne convient aux réseaux de neurones qui opèrent en environnement stationnaire. En raison de l'invariance d'un tel environnement au cours du temps, les paramètres du réseau peuvent être déterminés par un apprentissage limité à un ensemble fini représentatif de stimuli.
Une fois l'apprentissage achevé, l'identification du modèle de connaissance stationnaire de l'environnement justifie le gel des paramètres du réseau en phase de restitution de la connaissance, le réseau présente alors un comportement statique.

Généralement, l'environnement dans lequel est immergé le réseau est non stationnaire, dans ce cas, l'état et le comportement de l'environnement sont sujets à transformation au cours du temps. Par nature, l'apprentissage hors ligne ne permet pas de réactualiser le modèle de connaissance d'un environnement évolutif. Afin de surpasser cette limitation, l'apprentissage doit permettre l'ajustement en continu des paramètres du réseau. Ceci est réalisé en amenant le réseau à considérer chaque stimulus produit par l'environnement comme un nouveau stimulus à assimiler. La dépendance temporelle de la connaissance est ainsi implicitement intégrée dans le modèle du réseau par le biais d'un apprentissage en ligne exploitant une base d'exemples ordonnés dans le temps.

Le réseau présente alors un comportement adaptatif, au sens fort du terme, lié à la nature spatiotemporelle implicite de l'apprentissage en ligne. Aux dimensions de l'espace des paramètres du réseau s'ajoute la dimension du temps.

II.6. Conclusion

Nous avons exposé les éléments essentiels qui permettent de comprendre et de mettre en oeuvre des réseaux de neurones. Les réseaux de neurones sont des outils statistiques, qui permettent d'ajuster des fonctions non linéaires très générales à des ensembles de points. Comme toute méthode statistique, l'utilisation de réseaux de neurones nécessite que l'on dispose de données suffisamment nombreuses et représentatives.

Les réseaux de neurones permettent de modéliser des phénomènes statiques (réseaux non bouclés) et dynamiques (réseaux bouclés), il est toujours souhaitable, et souvent possible, d'utiliser, pour la conception du réseau, les connaissances mathématiques dont nous disposons sur le phénomène à modéliser : les réseaux de neurones ne sont pas nécessairement des "boîtes noires".

Chapitre III MODÉLISATION PAR RÉSEAUX DE NEURONES

III.1. Introduction

Dans ce chapitre, nous rappelons les notions de processus et de modèle, ainsi que les modèles discrets car dans les applications réelles, les réseaux de neurones formels pour la modélisation dynamique sont programmés sur des ordinateurs ou réalisés par circuits numériques. Enfin, nous aborderons les modèles dynamiques les plus utilisés, en présence de bruit dans la boucle ou au niveau de la sortie.

III.2. Définition d'un processus et d'un modèle

III.2.1. Processus

Un processus est caractérisé par une ou plusieurs grandeurs de sortie, mesurables, qui constituent le résultat du processus et une ou plusieurs grandeurs d'entrée, qui peuvent être des entrées de commande ou des perturbations. Ces dernières peuvent être aléatoires ou déterministes, mesurables ou non mesurables.

Les processus peuvent être de toute nature physique, chimique, financier, etc.

III.2.2. Modèles

Un modèle est une représentation mathématique du fonctionnement d'un processus, il représente les relations entre les entrées et les sorties du processus par des équations.

Si ces équations sont algébriques, le modèle est dit statique. Si ces équations sont des équations différentielles ou des équations aux différences récurrentes, le modèle est dit dynamique, respectivement à temps continu ou à temps discret.

III.2.2.1. Objectifs de la modélisation

Un modèle peut être utilisé soit pour simuler un processus à des fins pédagogiques, de détection d'anomalies de fonctionnement, de diagnostic de pannes, de conception assistée par ordinateur, etc. soit pour effectuer la synthèse d'une loi de commande, ou pour être incorporé dans un dispositif de commande.

III.2.2.2. Classification des modèles

Nous présenterons ci-dessous deux types de classification des modèles :

III.2.2.2.1. Classification selon le mode de conception

On distingue trois sortes de modèles en fonction des informations mises en jeu pour leur conception :

• **Les modèles de connaissance :** les modèles de connaissance sont construits à partir d'une analyse physique, chimique, biologique (ou autre suivant le type du processus), en appliquant soit les lois générales, fondées sur des principes (lois de la mécanique, de l'électromagnétisme, de la thermodynamique, de la physique quantique, etc.), soit les lois empiriques (finance, économie), qui régissent les phénomènes intervenant au sein des processus étudiés. Ces modèles ne comportent généralement pas de paramètres ajustables, ou des paramètres ajustables en très petit nombre.

Dans la pratique, il est toujours souhaitable d'établir un modèle de connaissance des processus que l'on étudie. Néanmoins, il arrive fréquemment que le processus soit trop complexe, ou que les phénomènes qui le régissent soient trop mal connus, pour qu'il soit possible d'établir un modèle de connaissance suffisamment précis pour l'application considérée. On est alors amené à concevoir des modèles purement empiriques, fondés exclusivement sur les résultats de mesures effectuées sur le processus.

• **Les modèles “boîte noire” :** les modèles “boîte noire” sont construits essentiellement sur la base de mesures effectuées sur les entrées et les sorties du processus à modéliser. La modélisation consiste alors à utiliser, pour représenter les relations entre les entrées et les sorties, des équations (algébriques, différentielles, ou récurrentes) paramétrées, et à estimer les paramètres, à partir des mesures disponibles, de manière à obtenir la meilleure précision possible avec le plus petit nombre possible de paramètres ajustables.
Le domaine de validité d'un tel modèle ne peut pas s'étendre au-delà du domaine des entrées qui est représenté dans les mesures utilisées pour l'apprentissage.

• **Les modèles “boîte grise” :** lorsque des connaissances, exprimables sous forme d'équations, sont disponibles, mais insuffisantes pour concevoir un modèle de connaissance satisfaisant, on peut avoir recours à une modélisation "boîte grise" (ou modélisation semi physique) qui prend en considération à la fois les connaissances et les mesures. Une telle démarche peut concilier les avantages de l'intelligibilité d'un modèle de connaissance avec la souplesse d'un modèle comportant des paramètres ajustables.

III.2.2.2.2. Classification selon l’utilisation

Indépendamment de la classification précédente, on peut distinguer deux types de modèles en fonction de l'utilisation qui en est faite.

• **Les modèles de simulation (ou simulateurs) :** un modèle de simulation est utilisé de manière indépendante du processus qu’il représente. Il doit donc posséder un comportement aussi semblable que possible à celui du processus. De tels modèles sont utilisés pour valider la conception d'un système avant sa fabrication (conception assistée par ordinateur en mécanique, en microélectronique, ...), pour la formation de personnels (simulateurs de vols), pour la prévision à long terme, etc.
Du point de vue de la structure du modèle, les sorties passées, mesurées sur le processus à modéliser, ne peuvent constituer des entrées du modèle. L'estimation des

paramètres et l'utilisation du modèle constituent deux phases successives et distinctes (apprentissage non adaptatif).

• **Les modèles de prédiction (ou prédicteurs) :** un modèle de prédiction est utilisé en parallèle avec le processus dont il est le modèle. Il prédit la sortie du processus à une échelle de temps courte devant les constantes de temps du processus. Les prédicteurs sont utilisés pour la synthèse de lois de commande, ou dans le système de commande lui-même (commande avec modèle interne).
Du point de vue de la structure du modèle, les sorties passées, mesurées sur le processus, peuvent constituer des entrées du modèle. L'estimation des paramètres et l'utilisation du modèle peuvent être effectuées simultanément si nécessaire (apprentissage adaptatif, utile notamment si les caractéristiques du processus dérivent dans le temps).

Cette partie présente la mise en oeuvre de plusieurs types de réseaux de fonctions paramétrées pour la modélisation dynamique de processus. Il s'agira donc exclusivement de modèles de type “boîte noire” qui peuvent être utilisés indifféremment comme simulateurs ou comme prédicteurs.

III.3. Conception d’un modèle

Lors de la conception d’un modèle de connaissance, la relation entre les entrées et la (ou les) sortie(s) du modèle découlent directement de la mise en équation des phénomènes physiques (chimiques, ou autres) qui régissent le fonctionnement du processus. Une fois le modèle obtenu sous forme analytique, des approximations peuvent être faites pour simplifier son expression (par exemple "linéariser" le modèle pour passer d'un modèle non linéaire à un modèle linéaire) si une telle approximation est justifiée.

Dans le cas d'une modélisation de type "boîte noire", la construction du modèle nécessite les trois éléments suivants :

- Une hypothèse sur l'existence d'une relation déterministe liant les entrées à la (ou aux) sortie(s). Cette relation est caractérisée par une fonction appelée fonction de régression ou régression). L'expression formelle supposée adéquate pour représenter cette relation est appelée modèle-hypothèse ;

- Une séquence de mesures des entrées - sorties du processus ;

- Un algorithme d'apprentissage.

Dans la suite de ce paragraphe, nous présentons les différents aspects qui doivent être pris en considération lors du choix d'un modèle-hypothèse.

III.3.1. Choix d'un modèle-hypothèse

Les connaissances dont on dispose a priori sur le processus doivent guider le concepteur dans le choix de la modélisation la plus appropriée (statique ou dynamique, linéaire ou non linéaire, ...). L'élaboration du modèle-hypothèse nécessite d'effectuer les choix suivants :

. Modèle statique ou dynamique : Lorsque l'on cherche à modéliser un processus physico-chimique ou biologique, il est généralement facile de savoir si l'application envisagée nécessite de modéliser la dynamique du processus (c'est-à-dire si l'on doit considérer une échelle de temps petite devant les constantes de temps du processus) ou si une modélisation statique suffit.

. Modèle linéaire ou non linéaire : Il n'est pas douteux que la plupart des processus que l'on peut rencontrer nécessiteraient des modèles non linéaires s'il fallait les décrire de manière précise dans la totalité de leur domaine de fonctionnement : la plupart des modèles linéaires constituent des approximations valables dans un domaine plus ou moins restreint. Il est donc important de pouvoir élaborer un modèle non linéaire pour rendre compte du comportement d'un processus, non seulement autour de ses points de fonctionnement "habituels", mais également lors des passages d'un point de fonctionnement à un autre.

. Modèle entrée-sortie ou modèle d'état : Dans le cas où l'on opte pour une modélisation dynamique, deux représentations sont possibles pour le modèle : il s'agit de la représentation d'état ou de la représentation entrée–sortie. L'état d'un processus est définit comme la quantité d'information minimale nécessaire pour prédire son comportement, étant données les entrées présentes et à venir. Il s'agit généralement d'un vecteur de grandeur égale à l'ordre du modèle. La représentation entrée–sortie est un cas particulier de la représentation d'état où le vecteur des états est constitué par la sortie et ses valeurs retardées dans le temps.

Si le but de la modélisation est de prédire le comportement entrée–sortie du processus, il existe généralement une infinité de représentations d'état (au sens d'états ayant des trajectoires différentes) solutions du problèmes. En revanche, la représentation entrée–sortie est unique.

. Présence de perturbations déterministes : Lorsque l'on cherche à réaliser un modèle dynamique, les perturbations déterministes peuvent être modélisées par une entrée supplémentaire (échelon, signal carré, sinusoïde).

En particulier, si le modèle est construit pour la synthèse d'une loi de commande, la prise en considération de l'existence d'une perturbation pendant la phase de modélisation peut améliorer les performances de la commande pour le rejet de cette perturbation. Par exemple, il est proposé dans [Mukhopa93] une approche qui consiste à considérer la perturbation comme la sortie d'un processus. La modélisation de ce

processus a pour effet d'introduire de nouvelles variables d'état, donc d'augmenter l'ordre du modèle.

. Présence d'un bruit : Lorsque l'on cherche à réaliser un modèle dynamique, une perturbation de type "bruit" est modélisée par une séquence de variables aléatoires. Un bruit peut agir de différentes manières sur un processus.
On distingue notamment le bruit de sortie (bruit additif qui affecte la mesure de la sortie du processus), et le bruit d'état (bruit additif qui affecte l'état du processus). Comme, en général, on ne connaît pas avec précision la nature du bruit qui affecte le processus, on doit effectuer des hypothèses sur celle-ci; on déduit de celles-ci la structure du modèle-hypothèse, et l'algorithme utilisé pour l'ajustement des paramètres. Une hypothèse erronée peut dégrader considérablement les performances du modèle. Ces problèmes ont été très largement étudiés dans le cas de la modélisation linéaire [Ljung 1987]. Dans le cadre de la modélisation non linéaire par réseaux de neurones, ces considérations sont développées dans [Nerrand 1994].

III.3.2. Du modèle-hypothèse au prédicteur ou au simulateur

Un modèle-hypothèse ayant été choisi, l'étape suivante consiste à établir l'expression du prédicteur théorique, c'est-à-dire l'expression de la prédiction de la sortie du processus à l'instant $n+d$ en fonction des données disponibles à l'instant n (entrées et sorties du processus et/ou du prédicteur à l'instant n et aux instants antérieurs). Enfin, la dernière étape consiste à établir l'expression du prédicteur (ou du simulateur) proprement dit : dans le cas d'une modélisation "boîte noire", ce prédicteur utilise une fonction paramétrée, dont on estime les paramètres, à partir de mesures effectuées préalablement sur le processus, de telle manière qu'il constitue la meilleure approximation possible du prédicteur théorique. A l'issue de la procédure d'estimation des paramètres (apprentissage), il faut évaluer la performance du prédicteur (ou du simulateur).

III.3.3. Modèles-hypothèses et leurs prédicteurs associés

Nous présentons dans ce paragraphe quelques exemples de modèles-hypothèses ainsi que les prédicteurs qui leurs sont associés, pour l'élaboration d'un modèle dynamique entrée-sortie. L'un des principaux paramètres qui interviennent dans le choix d'un modèle-hypothèse est la présence d'un bruit et la manière dont il agit sur le processus.

III.3.3.1 Modèles- hypothèses dynamiques et prédicteurs associés

Un modèle-hypothèse dynamique à temps discret postule l'existence d'une équation récurrente fournissant la sortie $y_p(k)$ du processus. Le bruit est susceptible d'intervenir sur la sortie ou sur l'état ou encore sur les deux. Un prédicteur est une équation récurrente déstinée à représenter les relations déterministes du comportement du processus. Le prédicteur théorique associé à un modèle-hypothèse, le prédicteur exprimant de l'espérance mathématique de la sortie du processus. Ce prédicteur est optimal au sens où, si le modèle-hypothèse est vrai, la variance de son erreur de prédiction est minimale.

Pour réaliser un modèle prédictif, il faut rechercher une approximation du prédicteur théorique dans une famille paramétrée de prédicteurs ayant les mêmes entrées que le prédicteur théorique, et dont les paramètres sont estimés en minimisant une fonction de coût portant sur les séquences mesurées sur le processus.

. Modèle- hypothèse entrée-sortie avec bruit de boucle

Le modèle-hypothèse avec bruit dan la boucle, appelé NARX (non-linéaire autorégressif avec entrée exogène ou extérieure) constitue une extension non linéaire du modèle linéaire ARX. Ce modèle-hypothèse s'exprime par :

$$y_p(k) = \varphi\left(y_p(k-1),\ldots,y_p(k-n),u(k-1),\ldots,u(k-m)\right) + w(k) \qquad \textbf{(III.1)}$$

Où les {w(k)} ont les réalisations d'un bruit pseudo blanc d'espérance nulle.

En supposant que la fonction φ est l'espérance mathématique de $y_p(k) / k\text{-}1$ et que la sortie dépend du bruit de l'instant *k-n* jusqu'à l'instant *n* : le bruit intervient dans la boucle.

Le prédicteur théorique associé à ce modèle est non bouclé :

$$y(k+1) = \varphi\left(y_p(k),\ldots,y_p(k-n+1),u(k),\ldots,u(k-m+1)\right) \qquad \textbf{(III.2)}$$

Il est optimal au sens ou la variance de l'erreur de prédiction $e(k+1) = y_p(k+1) - y(k+1)$ est minimale (égale à $w(k+1)$ qui est imprédictible).

Pour obtenir une estimation non biaisée, il faut alors mettre en œuvre un réseau de neurone possédant les mêmes entrées que le prédicteur théorique, exemple réseau de neurone formel non bouclé de paramètres θ :

$$y(k+1) = h\left(y_p(k),\ldots,y_p(k-n+1),u(k),\ldots,u(k-m+1),\theta\right) \qquad \textbf{(III.3)}$$

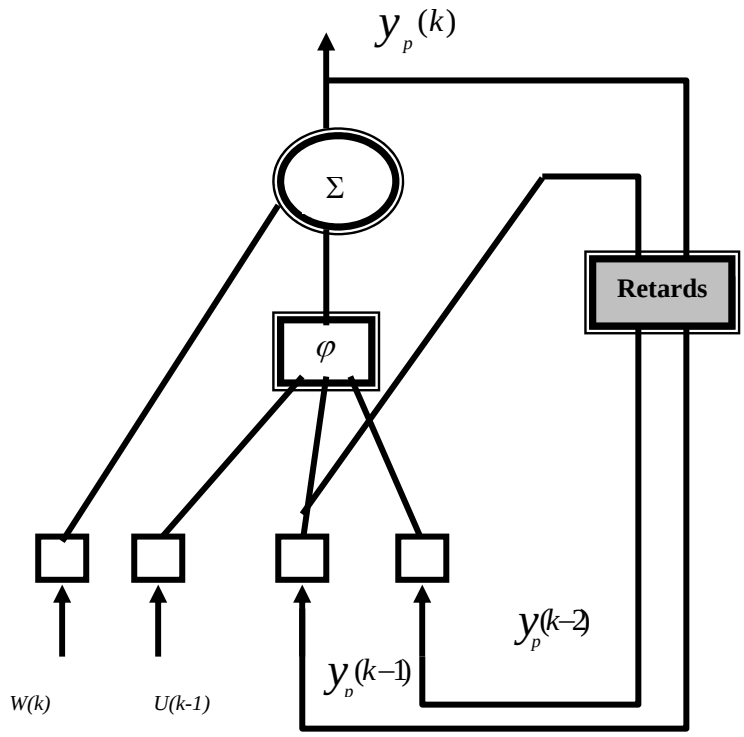


Fig.III.1. : Schéma d'un modèle NARX

.Modèle- hypothèse avec bruit de boucle et de sortie

Le modèle-hypothèse entrée-sortie le plus général comportant un bruit dan la boucle et un bruit sur la sortie est connu sous le nom de modèle NARMAX (non linéaire autorégressif à moyenne ajustée avec entrée extérieure), extension non linéaire du modèle linéaire ARMAX.

Ce modèle s'écrit :

$$y_p(k) = \varphi\left(y_p(k-1),...,y_p(k-n),u(k-1),...,u(k-m),w(k-1),...,w(k-p)\right) + w(k) \quad \textbf{(III.4)}$$

La séquence {w(k)} sont les réalisation d'un bruit pseudo blanc centré.

Le prédicteur théorique associé à ce type de modèle est donné par :

$$y(k+1) = \varphi\left(y_p(k),...,y_p(k-n+1),u(k),...,u(k-m+1),e(k),...,e(k-p+1)\right) \quad \textbf{(III.5)}$$

Avec $[e(k),\ e(k\text{-}1),\ \ldots,\ e(k\text{-}p\text{+}1)] = [w(k),\ w(k\text{-}1),\ \ldots,\ w(k\text{-}p\text{+}1)]$.

Où $e(k) = y_p(k) - y(k)$ (l'erreur de prédiction).

Le réseau de neurone formel correspondant s'exprime par:

$$y(k+1) = h\left(y_p(k),...,y_p(k-n+1),u(k),...,u(k-m+1),e(k),...,e(k-p+1),\theta\right) \quad \textbf{(III.6)}$$

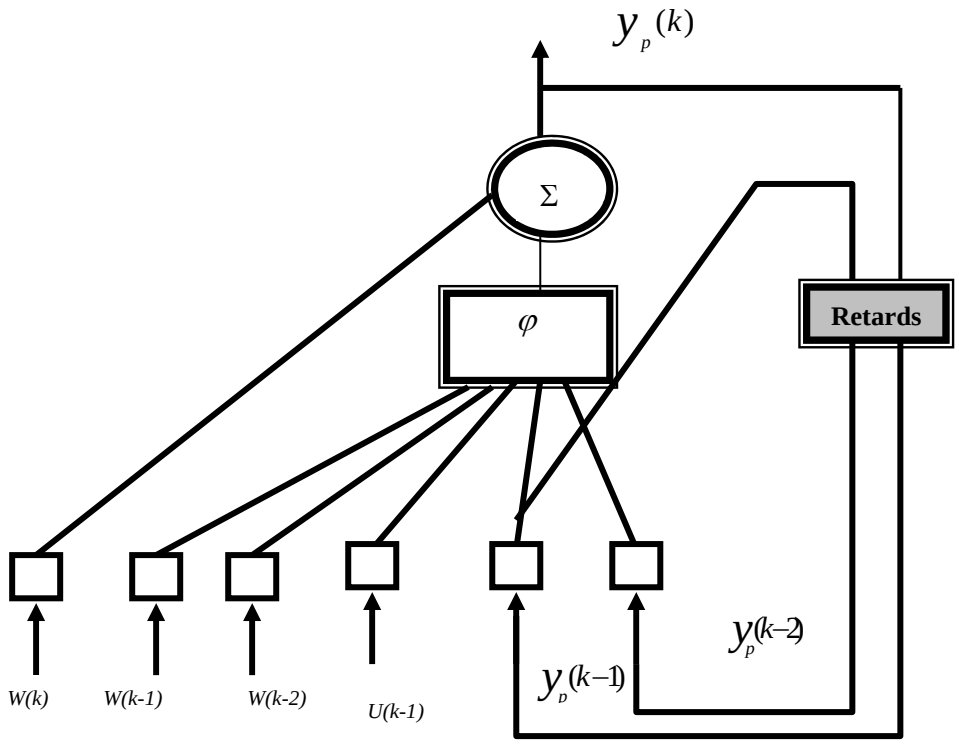


Fig.III.2. : Schéma d'un modèle NARMAX

On constate, en comparant les formes prédicteurs obtenues pour les modèles NARMAX, NARX l'influence de la modélisation des perturbations sur la détermination de la forme prédicteur.
Si le modèle hypothèse est exact, la forme prédicteur théorique fournit une variance de l'erreur de prédiction minimale et égale à la variance du bruit [Ljung, 1987].
Ceci est illustré par Nerrand sur des exemples NARX et NBSX dans le cas où les prédicteurs sont des prédicteurs neuronaux [Nerrand, 1992].

III.4. Conception de modèles NARMAX

En général, la procédure de modélisation se décompose en quatre étapes :

- conception d'un ensemble de modèles hypothèses candidats;
- définition des formes prédicteurs théoriques associées aux modèles hypothèses;
- pour chaque modèle hypothèse candidat : définition et apprentissage du modèle prédictif, défini par la forme prédicteur théorique, à l'aide de séquences d'entrées-sorties du processus (séquences d'apprentissage);
- sélection du meilleur candidat.

Dans le cas de modèles NARMAX, nous venons de montrer que l'on peut facilement déduire la structure du prédicteur théorique de l'expression du modèle hypothèse. Cependant, les valeurs de *n, m et p*, ainsi que l'expression de $\varphi(.)$, sont généralement inconnues. Lors de l'identification du processus, on doit donc trouver de bonnes valeurs de ces caractéristiques, et déterminer, dans une famille de fonctions $\phi(.,\theta)$ que l'on se donne (un réseau de neurones artificiels, par exemple), la fonction qui approche au mieux $\varphi(.)$.

III.5. Conclusion

Dans ce chapitre, nous avons présenté les principes de la modélisation « boîte noire » de processus dynamiques non linéaires. Les connaissances à priori nous conduisent à faire l'hypothèse que le processus peut être décrit par un modèle NARMAX. L'expression et les arguments de la fonction $\varphi(.)$ définissant ce modèle doivent être déterminées. Grâce au développement des méthodologies rigoureuses pour la conception de modèles, les réseaux de neurones sont devenus des outils de modélisation puissants dont les domaines d'applications sont multiples. Ils permettent de réaliser, de manière simple et efficace, des modèles précis, statiques ou dynamiques.

Chapitre IV COMMANDE PRÉDICTIVE GÉNÉRALISÉE NON LINÉAIRE

IV.1. Introduction

Durant les dernières années, la commande neuronale a évoluée vers des techniques puissantes, généralisées et extrêmement prometteuses pour la commande des procédés très complexes. Dans les sections qui suivent, nous traitons essentiellement des systèmes et des méthodes qui peuvent tirer un réel bénéfice de l'utilisation des techniques neuronales. Il s'agit de méthodes pour lesquelles : un modèle dynamique du processus est disponible ; le comportement spécifié par un cahier des charges peut se mettre sous la forme d'un modèle de référence ou bien peut s'obtenir par la minimisation d'un coût quadratique.

IV.2. Revue de la commande neuronale

IV.2.1. Approches de la commande neuronale

Deux classes d'approches de la commande neuronale peuvent être distinguées: la commande neuronale directe et la commande neuronale indirecte.

IV.2.1.1. Commande neuronale directe

Dans cette approche, le réseau fournit directement les nouvelles commandes qui sont appliquées au procédé. On retrouve dans cette catégorie, la commande supervisée et la commande par modèle inverse.

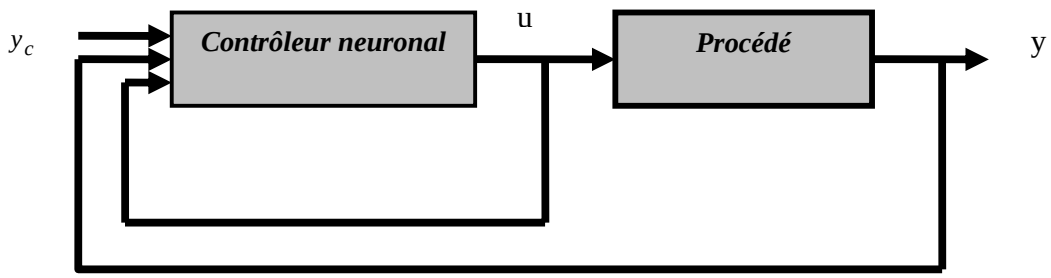


Fig.IV.1.a: Schéma de commande neuronale directe

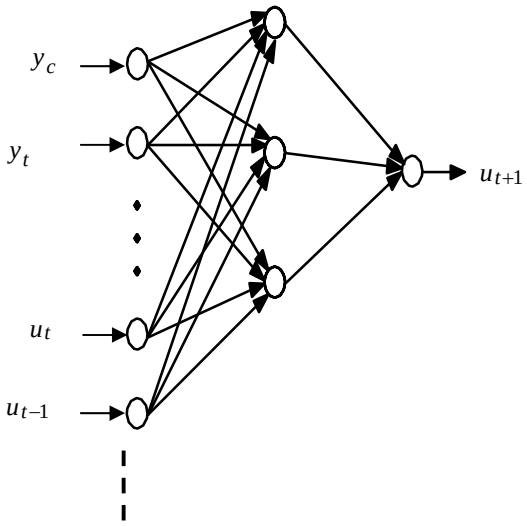


Fig.IV.1.b

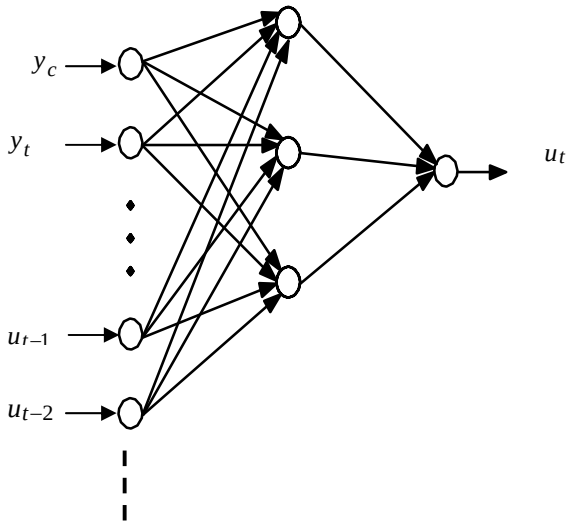


Fig.IV.1.c

La Fig.IV.1.a montre un schéma de commande supervisée. Dans ce schéma, un réseau de neurones comme celui montré à la Fig.IV.1.b, représente les transformations qui existent entre les signaux des senseurs et les actions de commande.

Les exemplaires utilisés pour I'entraînement de ce réseau peuvent être obtenus simplement en remplaçant le contrôleur neuronal par un opérateur humain.
Dans ce cas**,** l'opérateur doit déterminer minutieusement les actions de commande non pas en se basant sur ses propres sens et son expérience mais plutôt en se basant uniquement sur les informations fournies par les senseurs et les estimations de performance dérivées à partir de ces senseurs. En entraînant a priori le réseau à l'aide de ces exemplaires, la dynamique du procédé et le temps de réponse de l'opérateur humain seront communiqués au contrôleur neuronal. La sortie du procédé y_c (la consigne) est implicitement définie dans la pensée de l'opérateur comme étant un objectif. Néanmoins, cet objectif est présent dans les exemplaires et dans les connaissances acquises par le réseau de neurones.

Une des propriétés les plus importantes des réseaux de neurones, est leur aptitude à apprendre presque n'importe quelle relation à partir d'un ensemble d'exemplaires d'entrées / sorties (E/S). Cette propriété est vraie aussi pour l'apprentissage de l'inverse de ces relations. En effet, si I'on considère un ensemble d'exemplaires représentant le comportement physique et dynamique d'un procédé, alors le réseau de neurones peut apprendre l'inverse de cette relation simplement en commutant les entrées et les sorties dans les exemplaires avant de commencer l'opération d'apprentissage. Un modèle inverse de la dynamique du procédé sera ainsi obtenu.
En utilisant cette dernière propriété, le schéma de commande de la Fig.IV.1 pourrait devenir prédictif si l'on représente explicitement la consigne comme entrée supplémentaire au réseau de neurones (Fig.IV.1 c). En effet, en étudiant le procédé, il serait possible d'obtenir la sortie y du procédé au temps t+l en fonction des commandes u et des sorties y aux temps t-i où i = 0 à n. La valeur de n dépend spécialement de l'ordre de la dynamique du procédé.

Après avoir appris l'inverse de cette dernière relation, le contrôleur neuronal peut être intégré dans différents schémas de commande directe comme celui de la Fig.IV.1.a. Contrairement à un système de commande neuronal supervisé, le contrôleur neuronal

inverse apprend explicitement la commande à fournir pour obtenir une certaine sortie désirée du procédé.

Il est important de noter que l'approche que nous venons de discuter est généralement utilisée pour effectuer le suivi d'un modèle de référence ou bien pour maintenir le procédé à un point de référence comme dans un régulateur auto-ajustable.

L'adaptation en continu des modèles inverse et supervisé se fait en rétropropageant l'erreur qui correspond à la différence entre la commande désirée et la commande prédite par le modèle neuronal.

IV.2.1.2 Commande neuronale indirecte

Dans une approche de commande neuronale indirecte, le réseau de neurones est utilisé typiquement pour paramétrer ou modéliser le procédé. L'objectif de la commande est exprimé explicitement tel qu'il est montré à la Fig.IV.2. Dans ce schéma, le modèle neuronal fournit une estimation de la prochaine sortie du procédé y. Le régulateur compare la sortie estimée à la consigne et propose une meilleure action de commande en se basant sur une certaine stratégie.

Le modèle neuronal du procédé est entraîné à priori en utilisant un certain nombre d'exemplaires d'E/S prédéterminés, alors que l'adaptation en continu se fait en rétropropageant l'erreur qui correspond à la différence entre la sortie du procédé et la sortie prédite par le modèle neuronal (Fig.IV.2).

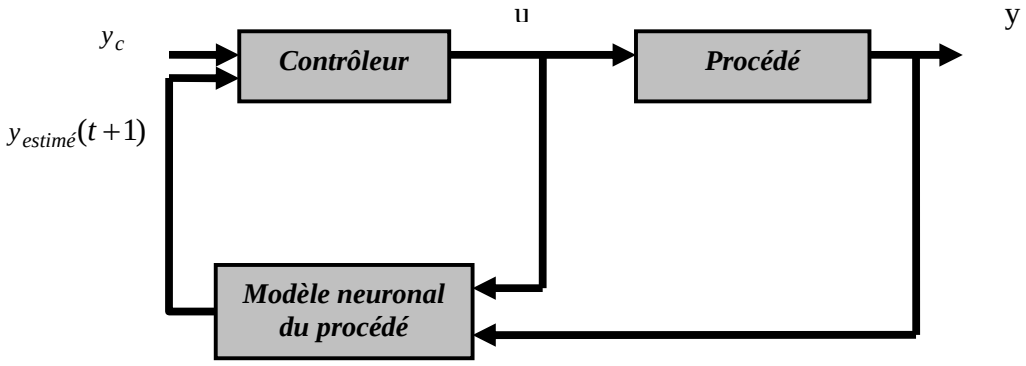


Fig.IV.2.a: Schéma de commande neuronale indirecte

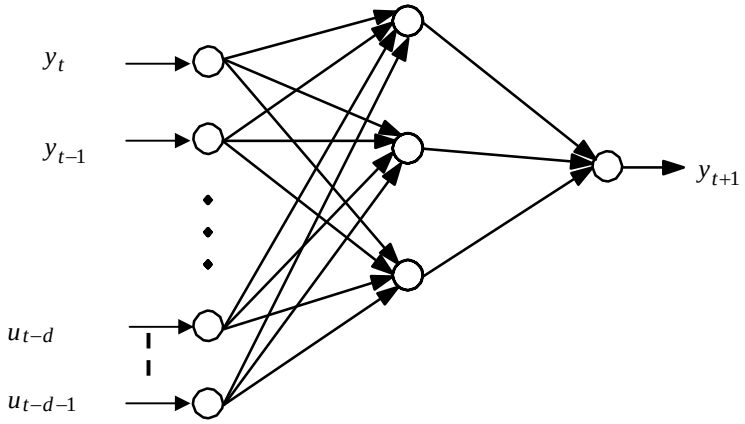


Fig.IV.2.b: modèle neuronal

IV.2.1.3. Structure de commande avec le modèle neuronal inverse

Le modèle neuronal inverse, aussi nommé régulateur neuronal, car il peut apprendre à déterminer une commande dans le but d'obtenir la sortie désirée, peut être utilisé dans différentes structures de commande. Une architecture de commande simple utilisant le modèle neuronal inverse est présentée à la Fig.IV.3. Elle consiste en un modèle neuronal inverse du procédé utilisé comme régulateur d'anticipation dans une boucle de rétroaction fermée.

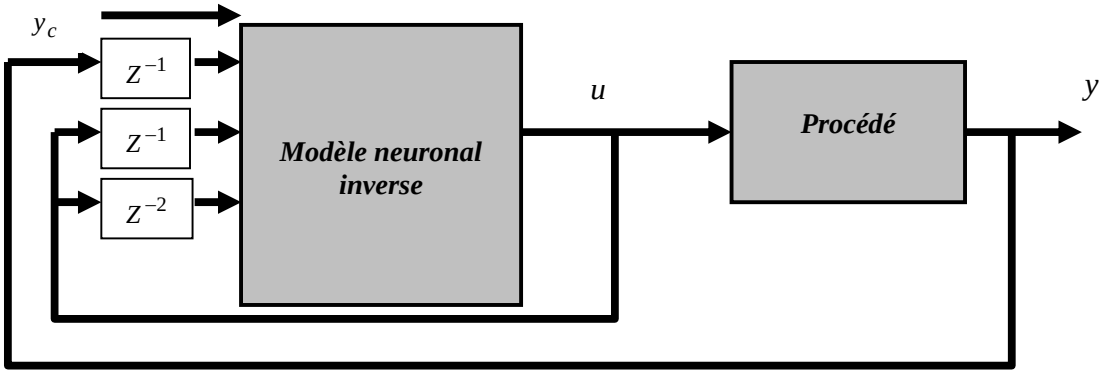


Fig.IV.3 : Structure de commande ayant comme régulateur le modèle

Pour l'apprentissage en continu du modèle neuronal inverse, la détermination de la commande que le modèle neuronal inverse aurait dû envoyer à la place de celle obtenue, n'est pas une tâche aussi évidente que dans le cas du modèle neuronal direct.
La plupart des algorithmes utilisés pour l'apprentissage du modèle neuronal inverse vont, à partir de l'erreur entre le point de consigne et la sortie du procédé (y_c - y), déterminer l'erreur de commande (u_c - u), c'est-à-dire la différence entre la commande idéale (u_c) et la commande obtenue par le modèle inverse (u).

IV.3. Commande prédictive généralisée non linéaire

IV.3.1.Introduction

Nous allons présenter dans ce qui suit, deux nouvelles stratégies de commande prédictive généralisée non linéaires, la première utilisant un modèle de prédiction neuronal du processus, pour le calcul des sorties prédites du critère de performance à minimiser, commande prédictive généralisée neuronale et la seconde est basée sur l'extraction d'un modèle linéaire à partir d'un modèle neuronal, commande prédictive généralisée linéarisée.

IV.3.2. Commande prédictive généralisée neuronale (NGPC)

IV.3.2.1. Introduction

La stratégie de la commande prédictive généralisée neuronale est basée sur la combinaison de l'avantage des réseaux de neurones de modéliser n'importe quel système non linéaire et de celui de la commande prédictive généralisée de pouvoir commander des systèmes complexes par la sélection des valeurs des paramètres de synthèse.

La structure du réseau de neurone et la méthode d'apprentissage sont définies pour l'identification du système non linéaire. Une fois le model neuronal est validé, il est utilisé pour prédire à chaque instant la sortie du système le long d'un horizon fuyant, un critère dépendant des erreurs entre la sortie prédite et le signal de référence est minimisé pour déterminer le signal de commande [Norgaard 2000], [Saidi et al 2005].

IV.3.2.2. Fonction de coût

La fonction coût à la forme quadratique suivante :

$$J = \sum_{i=N_1}^{N_2} \left[y(k+i) - r(k+i) \right]^2 + \rho \sum_{i=1}^{N_u} u^2(k+i-1) \qquad \textbf{(IV.1)}$$

Avec la contrainte:

$$\Delta u(k+i-1) = 0, \quad 1 \le N_u < i \le N_2 \qquad \textbf{(IV.2)}$$

N_u : horizon de commande;

N_1 : horizon minimum de prédiction;

N_2 : horizon maximum prédiction;

i : ordre du prédicteur;

r : trajectoire de référence;

ρ : facteur pondération;

Δ : opérateur de différentiation;

Le signal de commande u peut être sujet à des contraintes sur l'amplitude:

$$u_{\min} \le u(k+i) \le u_{\max}, \quad i = 1, \ldots, N_2 \qquad \textbf{(IV.3)}$$

La fonction coût est souvent utilisée avec le facteur de pondération ρ=0. Le paramètre le plus important dans la stratégie de la commande prédictive est l'horizon de commande N_u qui spécifie l'instant à partir duquel la sortie du contrôleur doit être maintenue constante.

La séquence de commande est obtenue par la minimisation de la fonction coût J par rapport à la commande U **.**

$$\frac{\partial J}{\partial U} = 0, \quad U = \left[u(k-d), u(k-d+1),.., \ u(k-d+N_u-1)\right]^T \qquad \textbf{(IV.4)}$$

IV.3.2.3. Prédicteur neuronal

L'utilisation des réseaux de neurones dans la modélisation et l'identification des processus non linéaire est justifiée par leurs capacités d'approximer les dynamiques des systèmes non linéaires. Pour estimer un processus non linéaire, le réseau de neurone doit subir un apprentissage jusqu'à ce que les valeurs optimales des vecteurs poids soient trouvées. Dan la plupart des applications, les réseaux feedforward sont utilisés, car les algorithmes d'apprentissage sont moins compliqués.

Le modèle neuronal incluant la plus large classe des processus non linéaire est le modèle NARMAX donné par:

$$y(k) = F[u(k-d-1), u(k-d-2),..., u(k-d-m),$$

$$y(k-1), \ y(k-2),..., \ \ y(k-n)] \qquad \textbf{(IV.5)}$$

où F[.] est une fonction non linéaire, d est le temps mort, n et m sont les degrés du modèle du système non linéaire.

Le modèle NARMAX peut être obtenu par l'ajustement des poids d'un perceptron multicouches avec des entrées décalées.

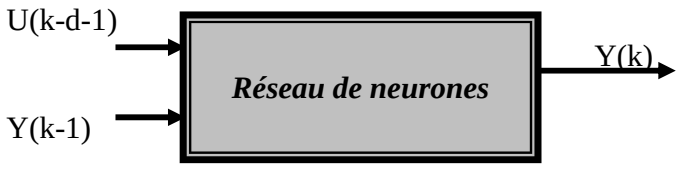


Fig. IV.4. Modèle NARMAX

La sortie du réseau de neurones est donnée par:

$$y(k)=F^{N}[U(k-d-1),Y(k-1)] \qquad \textbf{(IV.6)}$$

où F^N est la fonction de transfert entrée- sortie de la fonction non linéaire F dans (IV.5), et U(k-d-1), y(k-1) ont les vecteurs qui contiennent respectivement m et n élément décalés de u et y à partir de l'instant k-1:

$$U(k-d-1)=[u(k-d-1),u(k-d-2),...,\ u(k-d-m)]^{T}$$

$$Y(k-1)=[y(k-1),\ y(k-2),...,\ \ y(k-n)]^{T} \qquad \textbf{(IV.7)}$$

Le modèle NARMAX correspond à un réseau de neurones récurrent car certaines valeurs de ses entrées sont des valeurs passées de sa sortie.

Pour un réseau à deux couches, la sortie est définie par l'expression ci-dessous:

$$y(k)=\sum_{j=1}^{N} w_j \sigma_j(W_j^{u}U(k-d-1)+W_j^{y}Y(k-1)+b_j)+b \qquad \textbf{(IV.8)}$$

N : nombre de neurones dans la couche cachée;

σ_j : fonction d'activation du j-ème neurone de la couche cachée;

W_j^u : vecteur poids pour le j-ème neurone par rapport aux entrées stockées dans U(k-d-1);

W_j^y : vecteur poids pour le j-ème neurone par rapport aux entrées stockées dans Y(k-1);

b_j : biais pour le j-ème neurone de la couche cachée;

w_j : poids pour la couche de sortie correspondant au j-ème neurone de la couche de sortie;

b : biais pour la couche de sortie.

A partir de l'équation (IV.8), on peut écrire:

$$y(k+1)=\sum_{j=1}^{N} w_j \sigma_j (W_j^u U(k-d)+W_j^y Y(k)+b_j)+b \qquad \textbf{(IV.9)}$$

où

$$U((k-d)=[u(k-d),u(k-d-1),...,\ u(k-d+1-m)]^T$$

$$Y(k)=[y(k),\ y(k-1),...,\ y(k+1-n)]^T \qquad \textbf{(IV.10)}$$

L'expression (IV.9) représente un prédicteur à un pas. Celle d'un prédicteur à i pas peut être écrite sous la forme suivante:

$$y(k+i)=\sum_{j=1}^{N} w_j \sigma_j (W_j^u U(k-d+i-1)+W_j^y Y(k+i-1)+b_j)+b \qquad \textbf{(IV.11)}$$

où

$$U(k-d+i-1)=[u(k-d+i-1),u(k-d+i-2),...,\ u((k-d+i-m)]^T$$

$$Y(k+i-1)=[y(k+i-1),y(k+i-2),...,\ y(k+i-n)]^T \qquad \textbf{(IV.12)}$$

i : ordre du préditeur

L'algorithme NGPC est défini par les étapes suivantes:

1. Générer la trajectoire de référence;
2. Initialiser le vecteur de commande et déterminer le comportement future du système en utilisant le modèle;
3. Calculer la nouvelle commande qui minimise la fonction coût;
4. Répéter les étapes 2 et 3 jusqu'à ce que la minimisation soit achevée;
5. Appliquer la première commande au système;
6. Répéter tout le processus pour chaque pas d'échantillonnage.

Dans cette partie, l'approche Quasi-Newton basée sur l'algorithme de Broyden-Fletcher-Goldfarb-Shanno (BFGS) est utilisée voir l'annexe (meilleure convergence par rapport aux autres méthodes).

IV.3.2.4. Exemple de simulation

Cet exemple illustratif choisi, présente une dynamique nettement non linéaire excluant une approche linéaire classique, et met en lumière l'apport des réseaux de neurones pour une approche non linéaire. L'objectif de cette simulation est de tester cette technique sur un système stable en boucle ouverte et comportant des non linéarités. Ce système est décrit par l'équation différentielle suivante :

$$\dot{y}\dot{} + \dot{y} + y^3 + y = u$$

La méthode d'intégration de Runge Kutta est utilisée pour résoudre cette équation avec une fréquence d'échantillonnage de 0.2 [s]. L'entrée appliquée au système est une séquence d'impulsions d'amplitude et de durée aléatoires.

Après la procédure de simulation, l'ensemble des données entrées – sorties est divisé en deux, les cinq cents premiers échantillons sont utilisés pour l'apprentissage (Fig.IV.5).

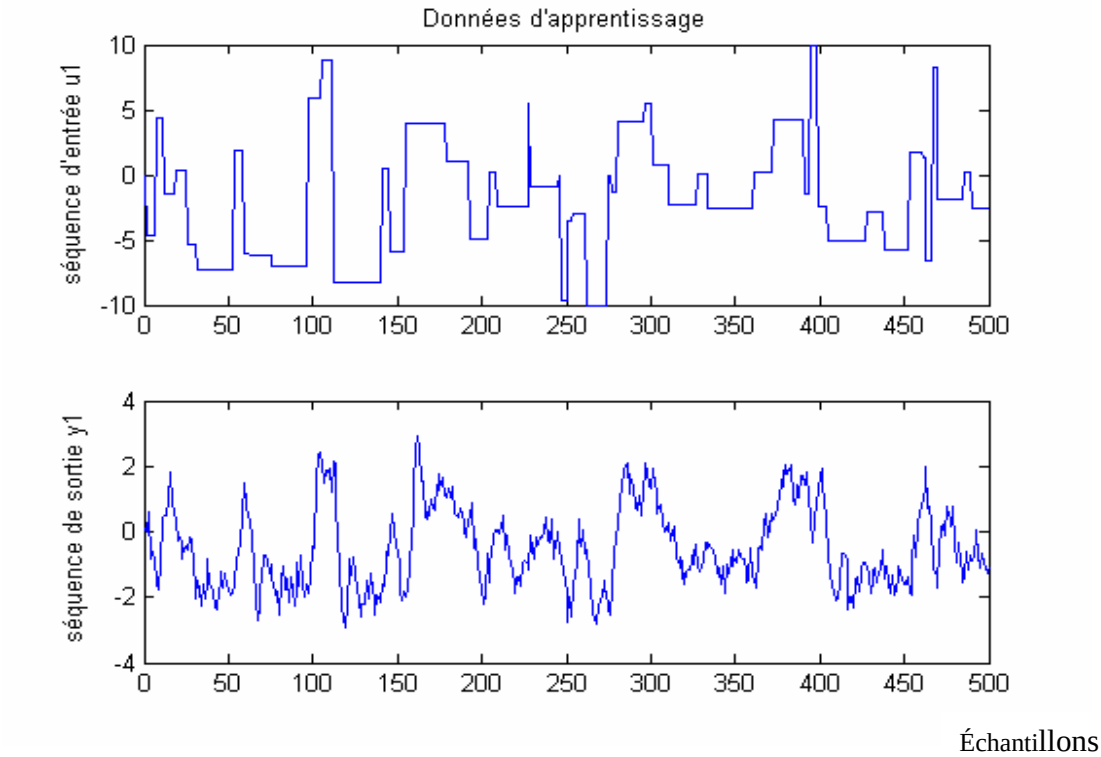


Fig.IV.5 : Données entrée-sortie pour l'apprentissage

Nous avons choisi une structure d'un réseau de neurones à deux couches, une couche cachée de cinq unités de fonction d'activation tanh et une couche de sortie linéaire (Fig.IV.6).

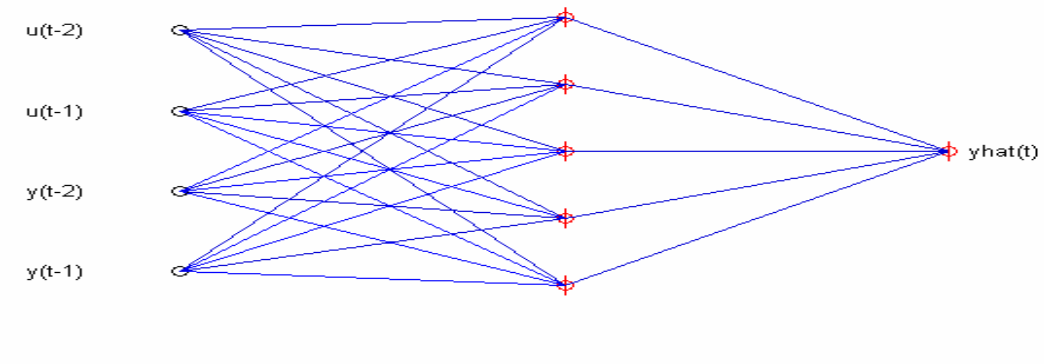


Fig.IV.6: Structure du réseau de neurones

Ensuite, la méthode de rétropropagation est utilisée pour l'apprentissage de ce modèle neuronal. Ce modèle neuronal est identifié et validé (Fig.IV.7).

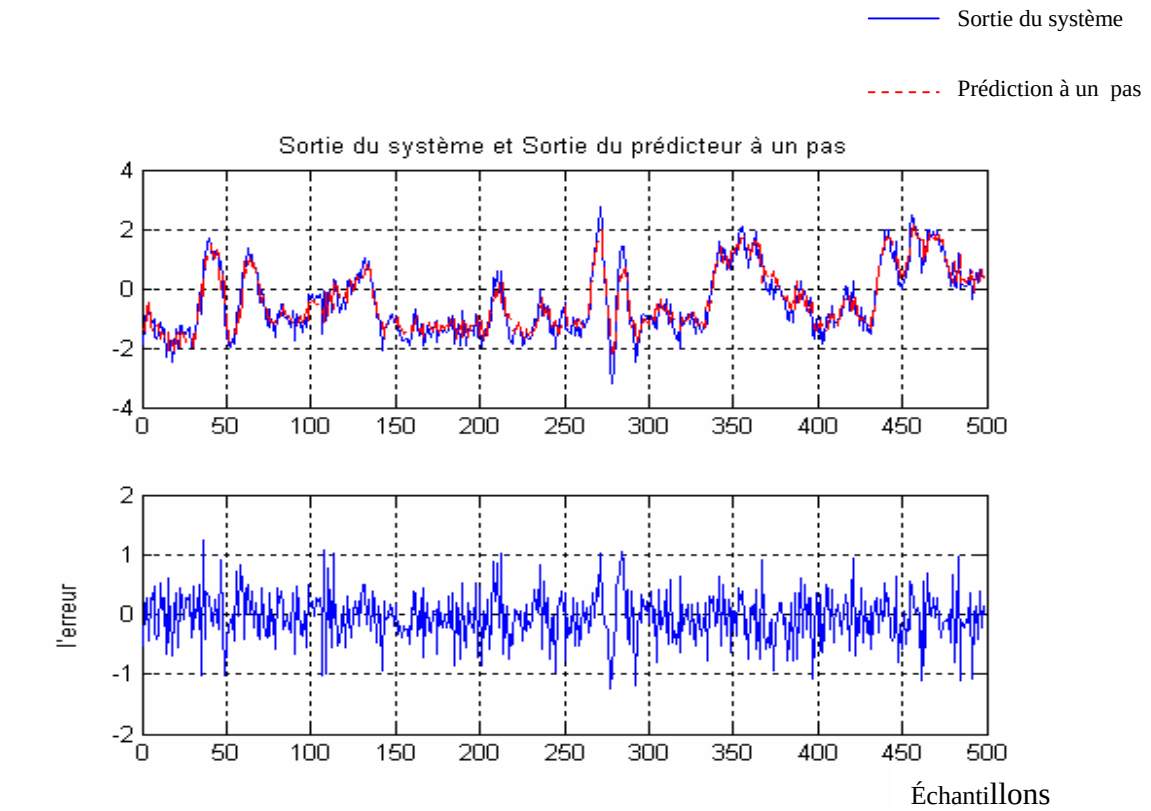


Fig.IV.7: Validation du modèle neuronal

Une fois le modèle est validé, il est utilisée pour générer les prédictions de la sortie. La méthode Quasi-Newton utilisant l'algorithme de B.F.G.S est employée pour la minimisation du critère, dépendant des erreurs entre la trajectoire de référence et la sortie prédite.

Après une procédure d'essais (Fig.IV.8, 9, 10) les valeurs des paramètres de synthèse ont été sélectionné comme suit : N1=d=1, N2=7, NU=2, ρ=0.03.

Le système arrive à suivre les variations de la référence et le signal de commande est acceptable.

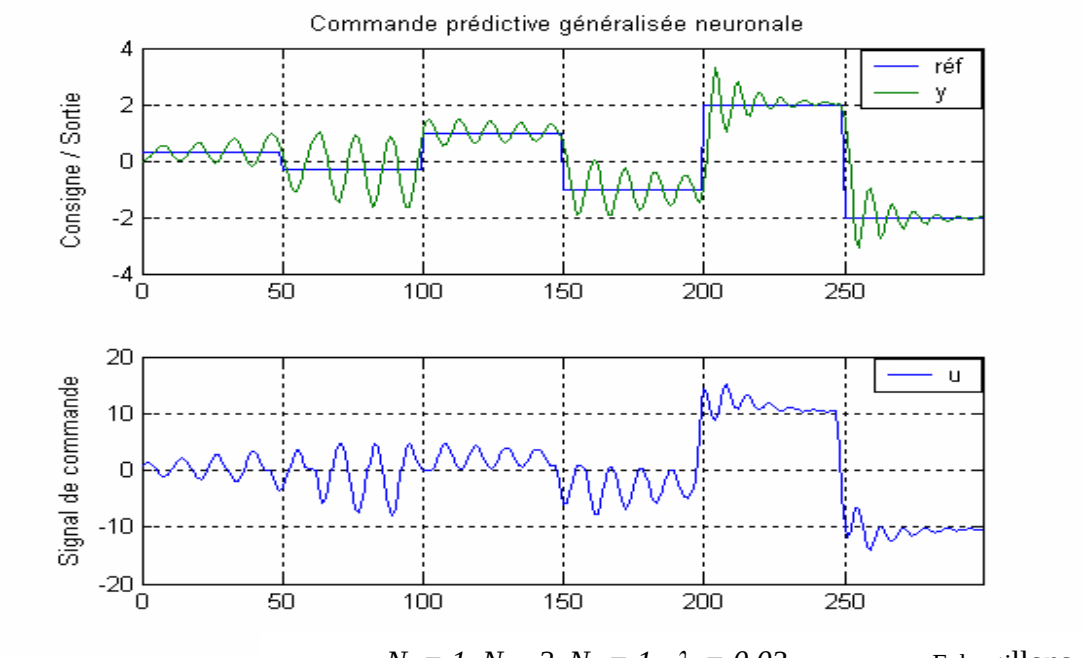
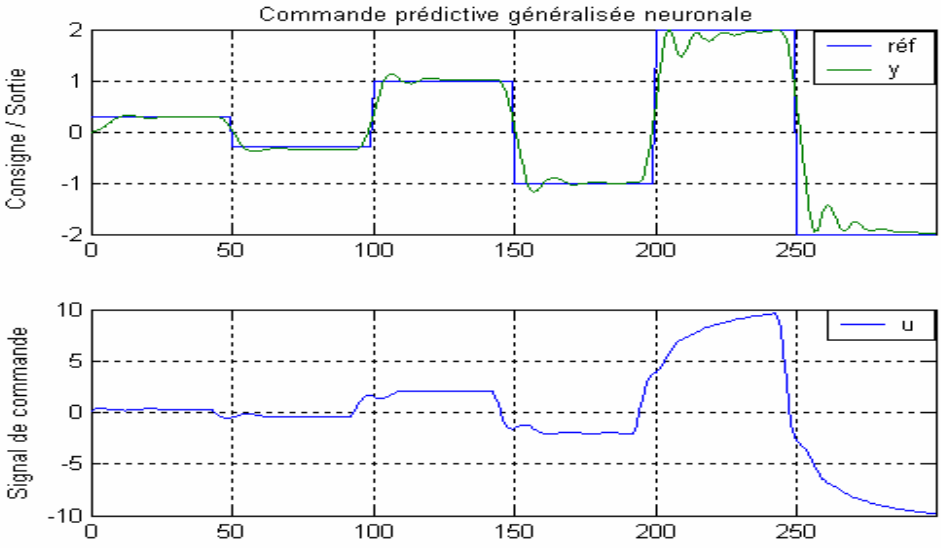
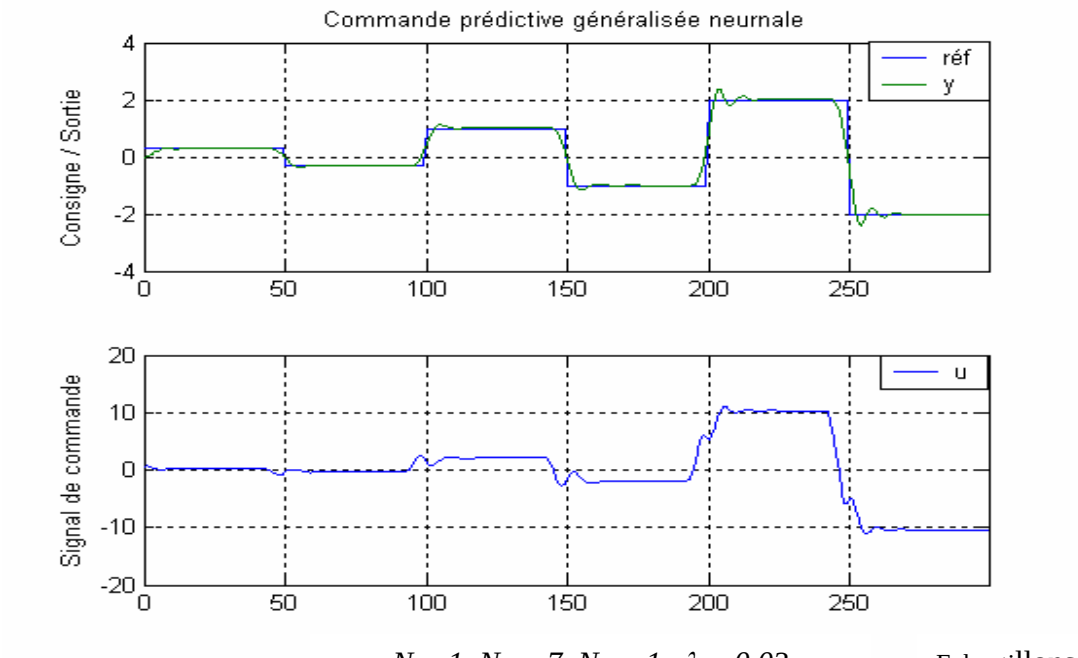


Fig.IV.8: Trajectoire de référence - signal de sortie et signal de commande



$N_1 = 1, N_2 = 7, N_u = 2, \lambda = 0.9$

Fig.IV.9: Trajectoire de référence - signal de sortie et signal de commande



$N_1 = 1, N_2 = 7, N_u = 1, \lambda = 0.03$

Fig.IV.10: Trajectoire de référence - signal de sortie et signal de commande

IV.3.2.5. Avantages et inconvénients de la NGPC

Avantages :

- Bonne pour la commande des systèmes à retard ;
- Peut stabiliser les systèmes instables en boucle ouverte ;
- Compense les bruits mesurables ;
- Prend en compte les contraintes sur les signaux d'entrée et de sortie.

Inconvénients:

- Le critère peut avoir plusieurs minima locaux ;
- Demande beaucoup de calculs.

IV.3.3. Commande prédictive généralisée linéarisée (LGPC)

IV.3.3.1. Introduction

La linéarisation des modèles non linéaires est une technique souvent utilisée pour la conception des contrôleurs pour les systèmes non linéaires. Dans le cas de la modélisation boîte blanche, c'est-à-dire le système est décrit par un ensemble d'équations différentielles non linéaires, le modèle est linéarisé autour d'un ou plusieurs points de fonctionnement (de stationnarité) suivi par l'application d'une loi de commande linéaire.

Le modèle obtenu à travers la linéarisation instantanée autour d'un point de fonctionnement est considéré valide seulement dans une certaine région autour de ce point. Le caractère des non linéarités et la taille de la région de fonctionnement peut alors déterminer si l'utilisation d'un seul modèle linéaire est suffisante.
En présence d'un ensemble de points de fonctionnement, nous obtenons plus d'un modèle linéaire. Dans ce cas, le système de commande doit contenir une banque de contrôleurs

sélectionnés par un ensemble de règles. Chaque contrôleur est conçu pour une certaine région de fonctionnement.

Si le système non linéaire est inconnu, et le modèle doit être défini à partir d'un ensemble de données expérimentales (entrées / sorties), les techniques linéaires d'identification ne peuvent être utilisées. Dans la plupart de ces cas, l'approche boîte noire utilisant les réseaux de neurones est préférée.

Dans cette la partie suivante, nous allons présenter une technique de linéarisation appelée linéarisation instantanée basée sur un modèle neuronal du système non linéaire.

IV.3.3.2. Linéarisation instantanée

L'idée de la linéarisation instantanée est d'extraire à chaque pas d'échantillonnage un modèle linéaire à partir d'un modèle neuronal non linéaire.
En supposant qu'un modèle neuronal (entrées / sorties) déterministe du système à commander est disponible.

$$y(t)=g[\varphi(t)] \qquad \textbf{(IV.13)}$$

Où le vecteur de régression est donné par :

$$\varphi(t)=[y(t-1),\ldots,\ y(t-n),u(t-d),\ldots,\ u(t-d-m)]$$

Le principe de la linéarisation instantanée est comme suit :

- Interpréter le vecteur de régression comme étant un vecteur définissant l'état du système et à l'instant t=τ linéariser g autour de l'état actuel $\varphi(\tau)$ pour obtenir le modèle approximé :

$$\tilde{y}(t) = -a_1\tilde{y}(t-1) - \ldots - a_n\tilde{y}(t-n) + b_0\tilde{u}(t-d) + \ldots + b_m\tilde{u}(t-d-m)$$

$$a_i = \frac{\partial g[\varphi(t)]}{\partial y(t-i)} \Bigg|_{\varphi(t)=\varphi(\tau)}$$

(IV.14)

$$b_i = \frac{\partial g[\varphi(t)]}{\partial u(t-d-i)} \Bigg|_{\varphi(t)=\varphi(\tau)}$$

Avec $\tilde{y}(t-i) = y(t-i) - y(\tau-i)$

$$\tilde{u}(t-i) = u(t-i) - u(\tau-i)$$

- En réarrangeant les termes, le modèle approximé peut être exprimé de la façon suivante :

$$y(t) = [1 - A(q^{-1})]y(t) + q^{-d} B(q^{-1})u(t) + \varepsilon(\tau) \qquad \textbf{(IV.15)}$$

avec $\varepsilon(\tau)$ est le biais défini par :

$$\varepsilon(\tau) = y(\tau) + a_1 y(\tau-1) + \ldots + a_n y(\tau-n) - b_0 u(\tau-d) - \ldots - b_m u(\tau-d-m)$$

Les cœfficients $\{a_i\}$ et $\{b_i\}$ ont été collectés dans les poly nômes $A(q^{-1})$ et $B(q^{-1})$:

$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_n q^{-n}$$

(IV.16)

$$B(q^{-1}) = b_0 + b_1 q^{-1} + \ldots + b_m q^{-m}$$

Le modèle approximé peut alors être interprété comme étant un modèle linéaire affecté par une perturbation constante, $\varepsilon(\tau)$, dépendant du point de fonctionnement actuel.

IV.3.3.3. Implémentation de la loi de commande

L'objectif est d'utiliser une commande prédictive généralisée linéaire pour contrôler un système ayant un comportement dynamique non linéaire (Fig.IV.5). Pour réaliser cette tache, il faut procéder comme suit :

- Exciter le système par un signal riche en information ;
- Collecter les données entrées – sorties ;
- Choisir une structure d'un réseau de neurones ;
- Identifier et valider ce réseau de neurones ;
- Utiliser ce réseau pour extraire à chaque pas d'échantillonnage un modèle linéaire ;
- Calculer la commande à partir de ce modèle linéaire.

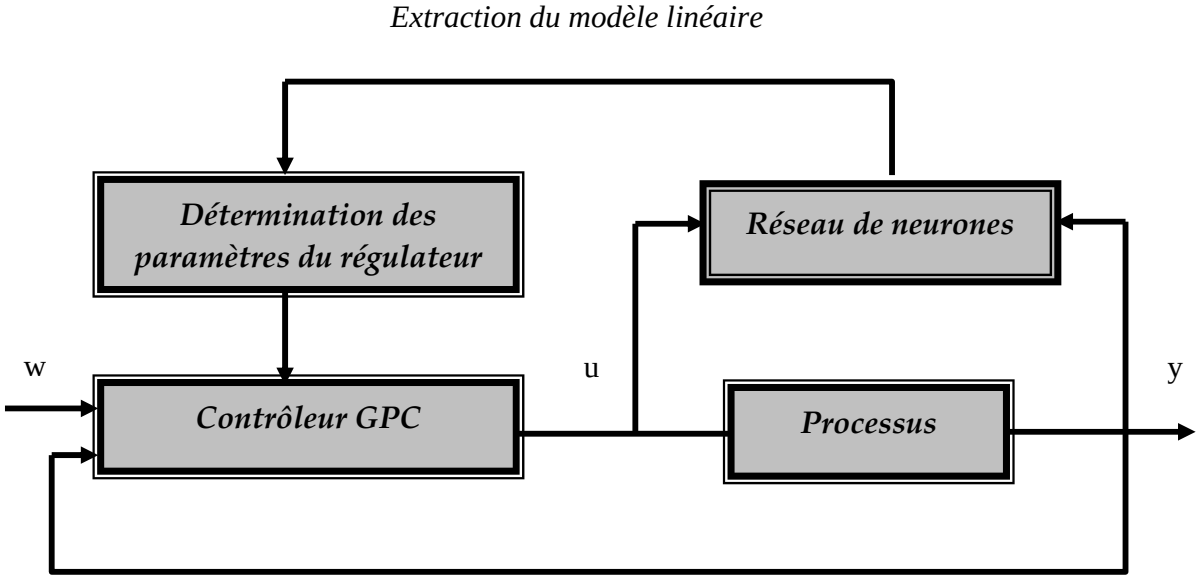


Fig.IV.11 : Schéma de la commande prédictive généralisée linéarisée

IV.3.3.4. Exemple de simulation

Nous avons repris le même exemple traité précédemment et nous avons obtenu les résultats suivants : après plusieurs opérations de sélection des valeurs des paramètres de synthèse (Fig.IV.12, 13, 14), nous avons eu une poursuite du signal de référence acceptable avec une commande douce avec les valeurs ci-après : $N_1 = 1$, $N_2 = 7$, $N_u = 1$, $\lambda = 0.03$.

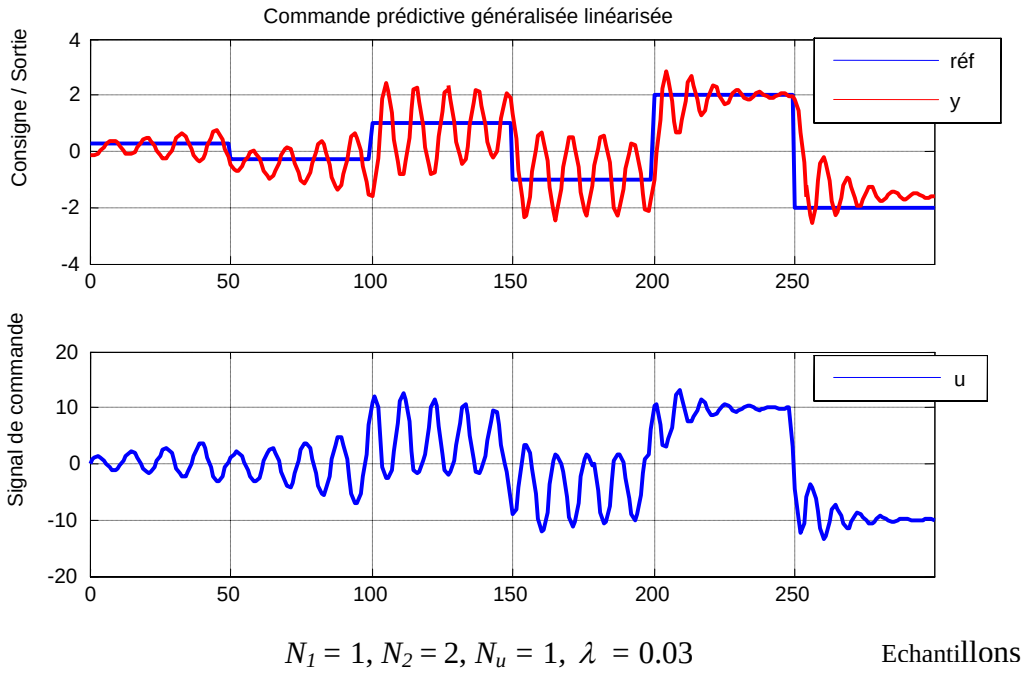


Fig.IV.12: Trajectoire de référence - signal de sortie et signal de commande

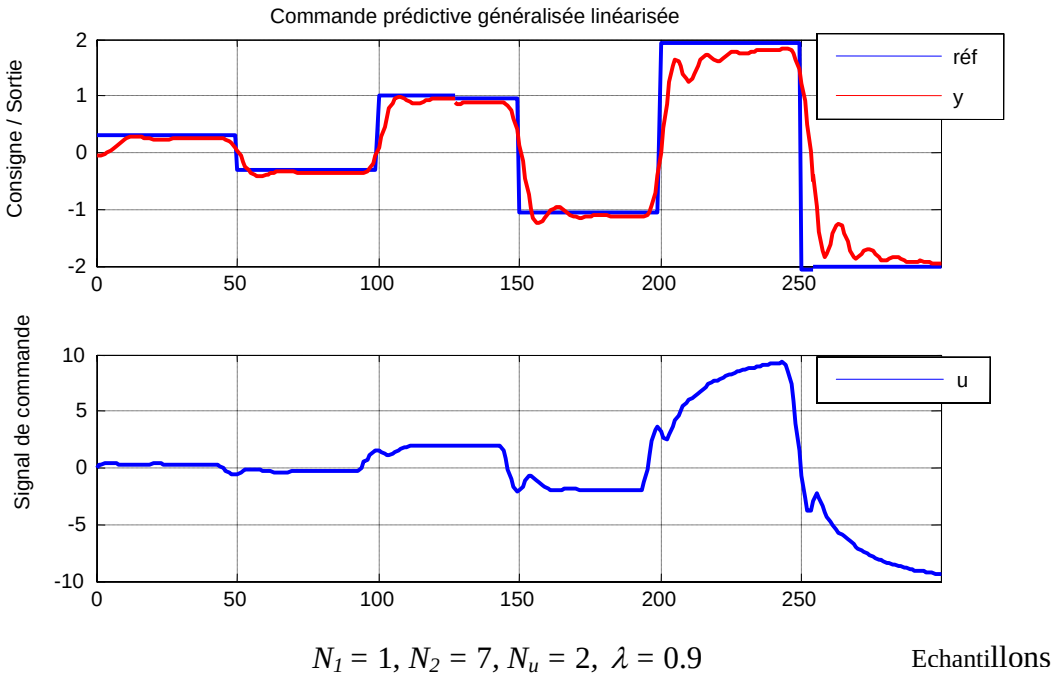


Fig.IV.13: Trajectoire de référence - signal de sortie et signal de commande

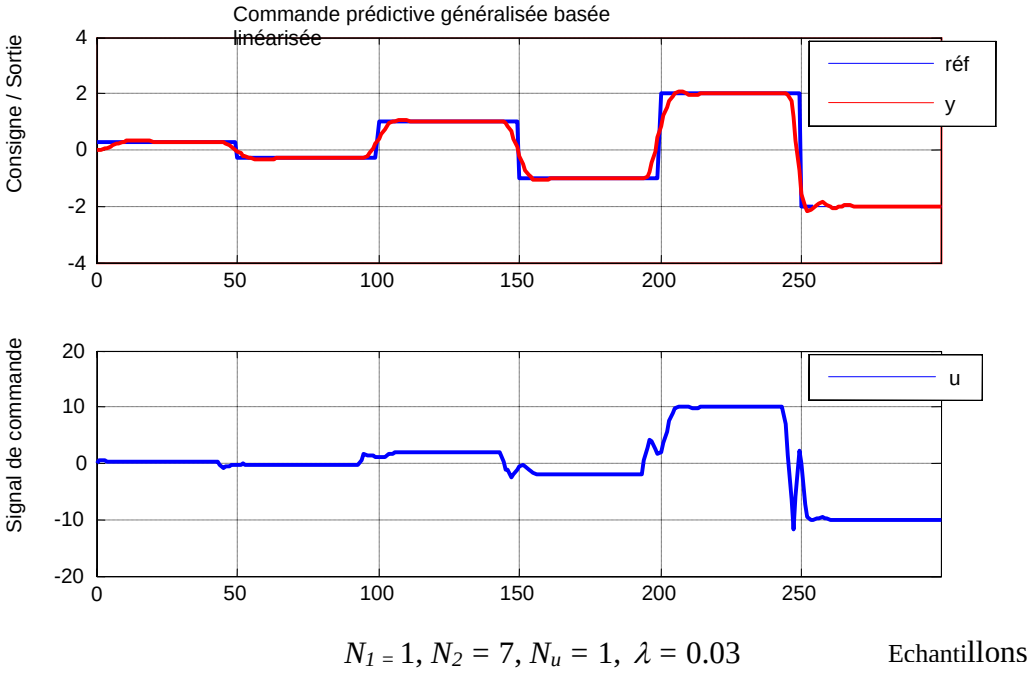


Fig.IV.14: Trajectoire de référence - signal de sortie et signal de commande

IV.3.3.5. Avantages et inconvénients de la LGPC

Avantages:

- Bonne pour la commande des systèmes à retard ;
- Peut stabiliser les systèmes instables en boucle ouverte ;
- Compense les bruits mesurables ;
- Prend en compte les contraintes sur les signaux d'entrée et de sortie.

Inconvénient:

- Liée au modèle linéaire, qui peut être valide uniquement au voisinage du point de fonctionnement.

IV.4. Conclusion

Dans cette partie, nous avons présenté deux approches de la commande prédictive pour les systèmes non linéaires modélisés par des réseaux de neurones. Dans la première approche, les prédictions sont calculées dune façon récursive à partir du prédicteur neuronal à un pas. Le calcul de la commande est un peu complexe car il doit se faire à chaque pas d'échantillonnage, par l'emploi d'un algorithme de minimisation itératif. La seconde approche est basée sur les modèles linéarisés extraits du modèle neuronal suivant le principe de la linéarisation instantanée. La stratégie de la commande prédictive généralisée linéarisée est préférée à celle de la commande prédictive généralisée neuronale car son implémentation et le calcul de sa commande sont plus simples. L'inconvénient majeur de cette stratégie de commande est que le modèle linéaire n'est valide qu'au voisinage du point de fonctionnement. Les performances de ces deux stratégies de commande ont été illustrées à travers exemple.

Chapitre V SIMULATEURS DE CONDUITE D'AUTOMOBILE

V.1. Introduction

Un simulateur de conduite d'automobile est un outil de réalité virtuelle permettant l'étude comportementale du conducteur dans diverses situations de conduite. La difficulté ou l'impossibilité de reproduire, dans le réel, les conditions de certaines situations routières accroissent l'intérêt de cet outil, utilisé pour confronter son utilisateur à des situations de conduite aussi proches que possible de la réalité. Les objectifs d'un simulateur de conduite sont :

- mettre à la disposition des nouveaux conducteurs des formations pour développer les réflexes dans les diverses situations de conduite afin de réduire au maximum le nombre des accidents de la route.

- développer des véhicules de plus en plus sophistiqués en intégrant de nouvelles techniques.

V.2. Plates–formes mobiles utilisées dans les simulateurs de conduite automobile

La restitution de mouvement pour les simulateurs de conduite automobile est assurée par la commande de plates formes- mobiles. Nous distinguons un ensemble d'architectures possibles décrites ci-dessous :

V.2.1. Plates-formes à base statique ou fixe

Dans ce type de simulateur, nous avons une liaison rigide contraignant tous les mouvements possibles. La conduite et la restitution du mouvement se basent alors uniquement sur le retour visuel, auditif et/ou autres substituts haptiques tel qu'un siège vibrant (Fig.V.1.).

Fig.V.1. : Simulateur à immersion totale développé par l'université américaine de Michigan

V.2.2. Plates-formes à structure série

Une plate-forme est dite à structure mécanique série si la cabine est portée par une série articulaire formant une chaîne cinématique ouverte à partir de la base jusqu'au point support de la cabine du simulateur (Fig.V.2).

Fig.V.2. : Simulateur à base série de l'institut suédois de recherche en transport routier

V.2.3. Plates-formes à structure parallèle

Les structures mécaniques parallèles comportent au moins une chaîne cinématique fermée. Une structure mécanique parallèle est constituée par un corps fixe, appelé "entrée" (base) et un corps mobile appelé "sortie" (terminal), connectés ensemble via un nombre de sous chaînes ouvertes et fermées (Fig.V.3).

une plate-forme parallèle est un mécanisme constitué d'une partie terminal à *n* DDL et d'une base fixe. La partie terminale de ce mécanisme est reliée à la base par plusieurs chaînes cinématiques indépendantes. Chacune de ces chaînes compte au plus deux segments articulés.

Dans le cas où la chaîne comporterait deux segments, l'articulation entre ces deux segments a un DDL. La motorisation s'effectue par *n* actionneurs à un DDL, un pour chaque chaîne.

Les premières expérimentations d'une structure parallèle en tant que manipulateur remonte à l'année 1947 par Mc Gough**.** En effet, l'auteur avait construit un mécanisme en chaîne cinématique fermée permettant de positionner et d'orienter une plate-forme mobile, qu'il employait pour tester des pneumatiques.
En 1965, Stewart a conçu un simulateur de vol basé sur une structure parallèle. Ce qui est connu aujourd'hui sous le nom de plate-forme de Stewart est en fait la plate-forme développée par Gough.

Cette structure est construite en reliant les extrémités de six sous-chaînes sérielles SPS (Sphérique – Prismatique – Sphérique) d'une part à la plate-forme mobile qui est un plateau triangulaire, et, d'autre part, à la partie fixe.
La partie mobile est mue par six actionneurs linéaires, constituant une liaison prismatique commandée. La plate-forme de Gough–Stewart est donc le précurseur des divers hexapodes et plate-formes pour restituer le mouvement de plusieurs simulateurs de conduite automobile actuels.

L'architecture parallèle semble s'être imposée comme un standard pour la restitution du mouvement des simulateurs de conduite automobile, mais aussi d'autres véhicules tels que les camions, les hélicoptères, les avions de chasse, etc.

Fig.V.3. : Plate-forme à structure parallèle de Stewart avec une demi cabine simulateur de l'université de Valence, Espagne

V.2.4. Plates-formes à structure hybride

Les plates-formes, dites à structure hybride (série et parallèle), sont généralement formées de chaînes séries connectées aux manipulateurs parallèles ou vice-versa (Fig.V.4).

Dans les simulateurs de conduite automobile recensés, la combinaison qui semble être un standard est un montage d'une structure parallèle sur une structure série linéaire.

La structure série offre un espace de travail plus étendu et permet la restitution d'un intervalle d'intensité d'accélération ou d'inertie plus importante.

L'inconvénient de cette structure est la difficulté de synthétiser les modèles géométriques et dynamiques directs et inverses.

Fig.V.4. :Plate-forme mobile hybride du simulateur de l'université de l'Iowa

.

V.3. Stratégies de contrôle des plates-formes de restitution du mouvement

Dans un simulateur de conduite automobile, le conducteur utilise particulièrement les informations rendues visuellement pour percevoir et appréhender comment le véhicule répond à ses commandes désirées.

Toutefois, la sensation du mouvement du véhicule est tout aussi importante même si le conducteur ne s'en rend pas consciemment compte. La restitution du mouvement est tout aussi essentielle si le réalisme et la cohérence du rendu sensoriel ne sont pas compromis.

D'un point de vu pratique, il est bien admis qu'aucun simulateur, aussi perfectionné soit-il, ne pourra reproduire exactement le mouvement du véhicule simulé.

En effet, les véhicules utilisent de grandes distances alors que les plates-formes de restitution sont limitées en terme d'espace de travail, et certaines configurations et transitions ne peuvent tout simplement pas être reproduites à cause des limites technologiques.

En prenant en compte l'ensemble du simulateur, il s'agit donc de reproduire au conducteur, le plus fidèlement possible, le mouvement effectué par le véhicule virtuel. Du point architectural on peut diviser ce processus en trois modules :

1- *Ce qui relève du très bas niveau* : Il concerne l'automatique et l'asservissement des actionneurs de la plate-forme mobile en tirant le meilleur compromis possible des objectifs classiques définis en terme de stabilité – précision – performances.

   Cet étage reçoit ses consignes du niveau intermédiaire décrit ci-après.

2- *Ce qui relève du niveau intermédiaire* : Il constitue le transfert entre le haut niveau simulation (expliqué au point suivant) et le bas niveau précédent.

   Ce module se base sur le modèle "robotique[1]" de la plate-forme mobile. Il prend en charge la planification des trajectoires désirées de la plate-forme en prenant en compte les contraintes liées aux butées articulaires, aux singularités éventuelles, etc.

   De plus, et c'est son rôle principal, comme les courses de la plate-forme mobile ne sont pas infinies, il se charge de la mise en œuvre des stratégies automatiques de remise de la plate-forme à la position initiale ou à une position donnée par un module de prédiction de la suite de la simulation…

3- *Ce qui relève du haut niveau* : il prend ses signaux d'entrés (i.e. les accélérations du véhicule) directement du moteur virtuel les transforme (ou projette) entre divers repères et produit les nouvelles sorties au niveau intermédiaire.

L'intérêt est porté sur deux fonctionnalités particulièrement intéressantes du niveau intermédiaire :

a) La décomposition et la génération des trajectoires désirées de la plate-forme en prenant en compte les illusions et les imperfections de la perception sensorielle. Il contrôle donc la plate-forme mais aussi le rendu visuel et éventuellement haptique en prenant en compte les contraintes articulaires et les consignes pouvant entraîner soit des configurations singulières soit des dépassements dans la course alloués ou possible des articulations mécaniques.

b) La procédure de remise au "point mort", nécessaire pour pallier la limitation de l'espace de travail des plates-formes de restitution du mouvement. Ce procédé est désigné dans le domaine par le mot anglais *washout*, et prend en compte les illusions et les seuils de perception sensoriels de l'opérateur pour faire en sorte que le conducteur ne se rend pas compte de la limitation des courses de la plate-forme tout en préservant la qualité du rendu du mouvement du véhicule réel et surtout la continuité et la cohérence des rendus vestibulaire, haptique et visuel.

V.3.1. Algorithme de restitution de mouvement

Le problème avec l'accélération longitudinale est l'importance de la vitesse qui peut être atteinte et les débuts d'accélérations qui sont présentées au conducteur, les périodes correspondant à des vitesses constantes peuvent être ignorées et relèvent de la seule vection.

Les débattements du dispositif de restitution du mouvement sont limités, surtout pour les mécanismes parallèles, les périodes du mouvement correspondant à des vitesses constantes ou aux variations faibles non perçues doivent être exploités pour remettre la

cabine à la position de référence. Il faudrait donc revenir "à zéro" sans que cela puisse être senti par le conducteur. C'est ce qui est communément appelé "*washout*". Il faut aussi revenir le plus vite possible sans éveiller l'attention de l'opérateur.
Les seuils qui permettent de réaliser le *washout* sont évalués autour de 0.1m/s², [Gun 78].

Dans un algorithme de restitution de mouvement (ou de génération de trajectoire) classique, les intensités des accélérations et des vitesses angulaires du véhicule simulé sont d'abord modifiées (changement d'échelle) puis bornées.
Les valeurs obtenues sont projetées dans le repère inertiel de la plate-forme.
Les valeurs sont alors filtrées pour éliminer les basses fréquences, qui correspondent aux mouvements de grandes amplitudes du véhicule.

Les paramètres des filtres utilisés dans l'algorithme sont déterminés pour maximiser la sensation des mouvements sélectionnés pour la restitution tout en éliminant ceux qui engendrent des singularités ou des dépassements de l'espace atteignable de la plate-forme.
En effet, le déplacement de la plate-forme revient alors toujours à zéro après une application d'une force spécifique, et force ce qu'on a convenu d'appeler le *washout*.
Notons à ce propos, que le mot "filtre *washout*" désigne par abus de langage un algorithme de restitution de mouvement (en anglais *motion cueing*) qui, en en jargon robotique, n'est autre qu'une génération de trajectoire.

Dans le simulateur de Renault, un algorithme de restitution de mouvement a été adopté avec les particularités suivantes, [Reymond et al 2000], (Fig.V.5.) :

- le modèle du véhicule donne les accélérations angulaires et linéaires ;

- les accélérations sont d'abord filtrées, puis intégrées pour être encore filtrés par le *washout* ;

- les sorties du *washout* sont utilisées pour le contrôle de la plate-forme.

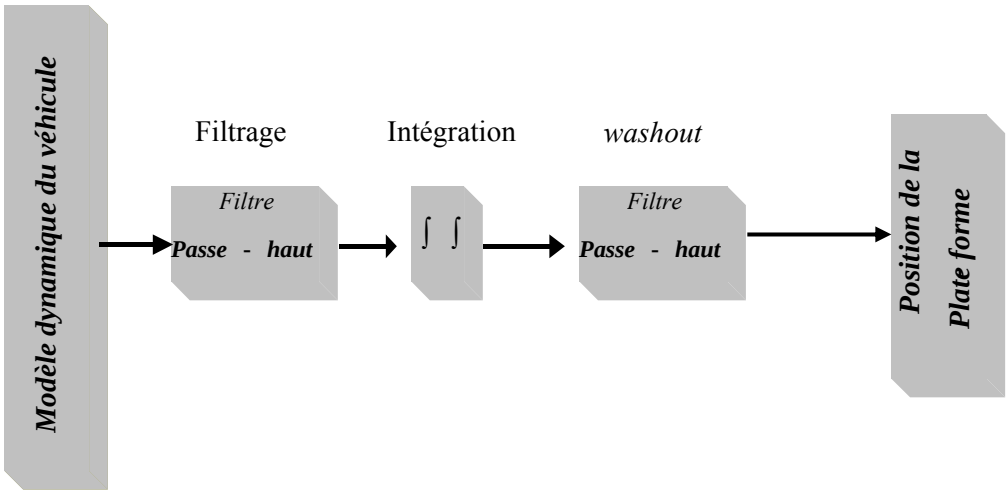


Fig.V.5. : algorithme de restitution du mouvement longitudinal

V.4. Description d'un simulateur à deux degrés de liberté

Le simulateur de conduite conçu par les équipes de recherche de l'institut de recherche sur le transport et sa sécurité (INRETS) de paris et le laboratoire des systèmes complexes (LSC) de l'université d'evry – France-, est considéré comme étant deux systèmes indépendants liés mécaniquement, le siège rotatif et la plateforme mobile (Fig.V.6). Chacun d'eux est commandé par un seul actionneur, un moteur électrique à courant continu.

Les deux actionneurs sont utilisés pour la restitution du mouvement longitudinal et le mouvement du siège.

Le mouvement longitudinal aura la fonction de simuler une conduite en fil en restituant des accélérations ou des décélérations sur des courtes courses (exemple : au lieu un freinage sec sur une cinquantaine de mètres, avec des commandes qu'on souhaite développer, cela devra être fait sur quelques dizaines de centimètres).

Le but ici reste la perception du mouvement et non pas le mouvement lui-même.

Le siège du simulateur (dossier seul ou avec assise) est aussi motorisé pour dupliquer avec les mouvements longitudinaux de la base. Dans la suite de ce travail, nous nous intéressons au mouvement longitudinal assuré par la plate-forme.

Fig.V.6.: Architecture du simulateur de conduite de l'INRETS-LSC

V.4.1 Modélisation de la plate-forme

La plateforme mobile supporte la cabine comportant le siège, le tableau de bord et le conducteur (Fig.V.7). Le mouvement longitudinal de la cabine est assuré par un mécanisme de transmission vis écrou à roulements, commandé par un moteur à courant continu.

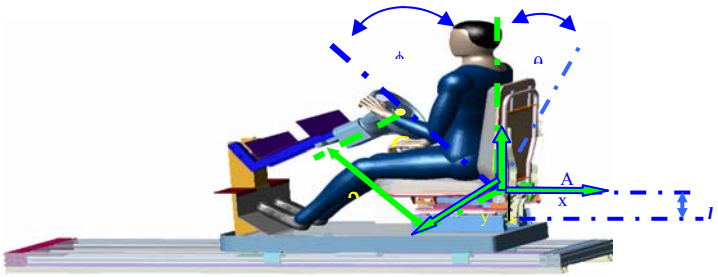


Fig.V.7.: Schéma de la plateforme mobile

Les équations du système sont comme suit :

Equation électrique de l'actionneur :

$$u - e = R_1 + L_1 \frac{di}{dt} \qquad \textbf{(V.1)}$$

Où u est la tension appliquée en volt, e est la force contre électromotrice en volt, R_1 est la résistance de l'actionneur en ohms, L_1 est l'inductance de l'armature en henry et i est le courant électrique en ampères.

Equation mécanique de l'actionneur de traction de la cabine :

$$T_{a1} = J_{a1} \frac{d\omega_{a1}}{dt} + f_{a1}\omega_{a1} + \frac{T_{al1}}{N_1} \qquad \textbf{(V.2)}$$

Où les indices a et l sont respectivement pour l'actionneur et la charge, T est le couple en N.m, J est l'inertie en Kg.m^2, ω_{al} est la vitesse angulaire en rad/sec, f est le frottement de l'armature en N.m.sec/rad, et N_1 le facteur de réduction. Le couple T_{al} et la force contre électromotrice e sont donnés par :

$$T_{a1} = k_{t1} i\,, \qquad e = k_{e1}\omega_{a1} \qquad \textbf{(V.3)}$$

i : courant électrique;

ω_{a1} : vitesse angulaire;

$k_{t1,}\, k_{e1}$: constantes

L'ensemble cabine se déplace sur des voies guidées sous l'action d'une force externe F_{x1} en newton suivant l'axe des $\overrightarrow{X}$. L'équation du mouvement est définie comme suit :

$$M \frac{d\dot{x}}{dt} + f_{x1}\dot{x} = F_{x1} \qquad \textbf{(V.4)}$$

M, masse totale de l'ensemble cabine mass (kg) = $m_c + m_t$ où m_c est la masse de la cabine et m_t est la masse estimée du conducteur.

Le mécanisme de traction est commandé par le couple externe T_{sl} donné par :

$$T_{s1} = J_{s1}\frac{d\omega_s}{dt} + f_{s1}\omega_{s1} + T_{sl1} \qquad \textbf{(V.5)}$$

Où J_{sl} est l'inertie du mécanisme de traction, f_{sl} est la force de frottement et T_{sl1} est le couple induit par la charge.

$$T_{sl1} = \frac{p_1}{2\pi\eta}F_{X1} \qquad \textbf{(V.6)}$$

L'équation (V.5) peut être écrite de la façon suivante:

$$T_{s1} = J_{s1}\frac{d\omega_{s1}}{dt} + f_{s1}\omega_{s1} + \frac{p_1}{2\pi\eta}(M\frac{d\dot{x}}{dt} + f_{x1}\dot{x}) \qquad \textbf{(V.7)}$$

Finalement, on obtient l'équation du mouvement de la cabine ci-dessous:

$$k_{t1}i = (\frac{2\pi N_1}{p_1}J_{a1} + \frac{2\pi}{p_1 N_1}J_{s1} + \frac{p_1}{2\pi\eta N_1}M)\frac{d\dot{x}}{dt} + (\frac{2\pi N_1}{p_1}f_{a1} + \frac{2\pi}{p_1 N_1}f_{s1} + \frac{p_1}{2\pi\eta N_1}f_{x1})\dot{x}$$

Alors :

$$u = R_1 i + L_1\frac{di}{dt} + \frac{2\pi N_1 k_{e1}}{p_1}\dot{x} \qquad \textbf{(V.11)}$$

En utilisant la transformée de Laplace, on obtient la fonction de transfert :

$$\frac{X}{U} = \frac{1}{s}\frac{k_{t1}}{(J_1 s + f_1)(L_1 s + R_1) + \frac{2\pi}{p_1}N k_{e1}k_{t1}} \qquad \textbf{(V.12)}$$

X : position de la cabine

U : signal de commande (tension électrique)

V.4.2. Restitution du mouvement:

Pour donner au conducteur l'illusion de la sensation des effets inertiels du simulateur, la plateforme est équipée d'un algorithme classique washout.

Le washout ou le système de restitution de mouvement fait penser le conducteur qu'il effectue des mouvements continus alors que l'espace de déplacement est limité.
La commande assure de faibles déplacements et un retour à la position neutre, durant les phases continues du signal d'accélération pour préparer la plateforme à un autre éventuel mouvement.
Cette commande est générée par un contrôleur prédictif et appliquée au moteur à courant continu responsable du mouvement de translation [Saidi et al 2006].

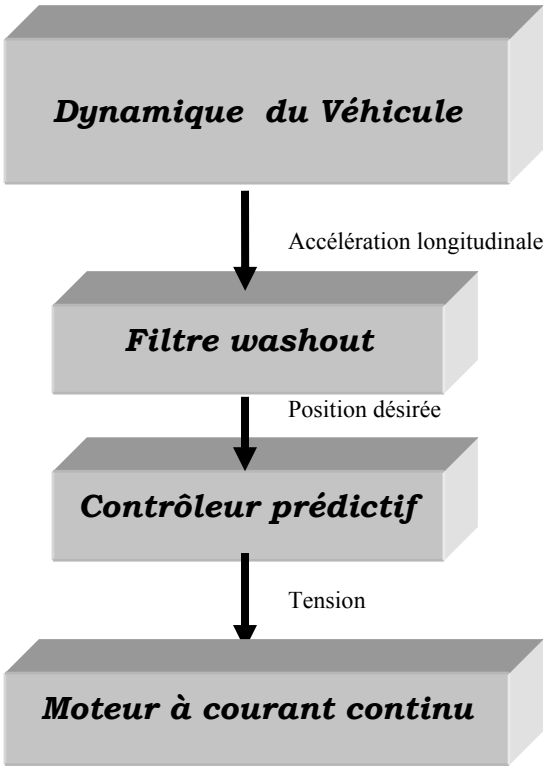


Fig.V.8. : Architecture de la commande de la plateforme mobile

V.4.2.1. Extraction de la position désirée

L'accélération longitudinale est récupérée à partir des signaux appliqués par le conducteur sur le véhicule réel (Fig.V.9).

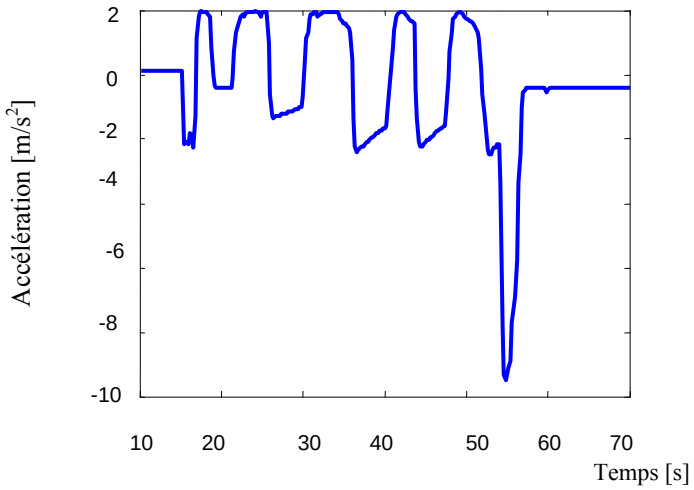


Fig.V.9. : Signal d'accélération longitudinale

Cette accélération est passée par un algorithme de washout classique qui comporte deux filtres passe haut. Le premier filtre passe haut permet d'enlever les composantes basses fréquences de l'accélération (composantes continues) car les composantes basses fréquences sont celles qui durent longtemps et provoquent de grands déplacements.

Les constantes de temps de la fonction de transfert de ce filtre sont choisies d'une manière à ne pas dépasser l'enveloppe de travail de la plateforme (1.2 [m]).

Après ce filtrage passe haut, le signal est intégré deux fois pour avoir la position correspondante.

Cette position est passée par un deuxième filtre qui permet de ramener la plateforme à sa position neutre à chaque fois où l'accélération est constante et d'une manière non perceptible par le conducteur, afin de la préparer un autre éventuel déplacement.

A la sortie de ce filtre, on obtient la position consigne (ou référence) à envoyer à la plate forme.

V.4.2.2. Élaboration de la loi de commande prédictive

La loi de commande prédictive abordée au chapitre I, a été implémentée pour assurer le mouvement longitudinal de la plateforme. Comme l'espace de déplacement de cette dernière est restreint entre - 0.6 m et + 0.6 m et afin d'éviter la collision avec les butées et remplacer les contacts de fins de course utilisés qui présentent l'inconvénient de provoquer l'arrêt totale du système (coupure de l'alimentation), la version sous contrainte de la GPC est utilisée.

Des tests de sélection des valeurs des paramètres de synthèse ont permis de retenir les valeurs suivantes : N_1=1, N_2=7, N_u=4, λ=0.001.

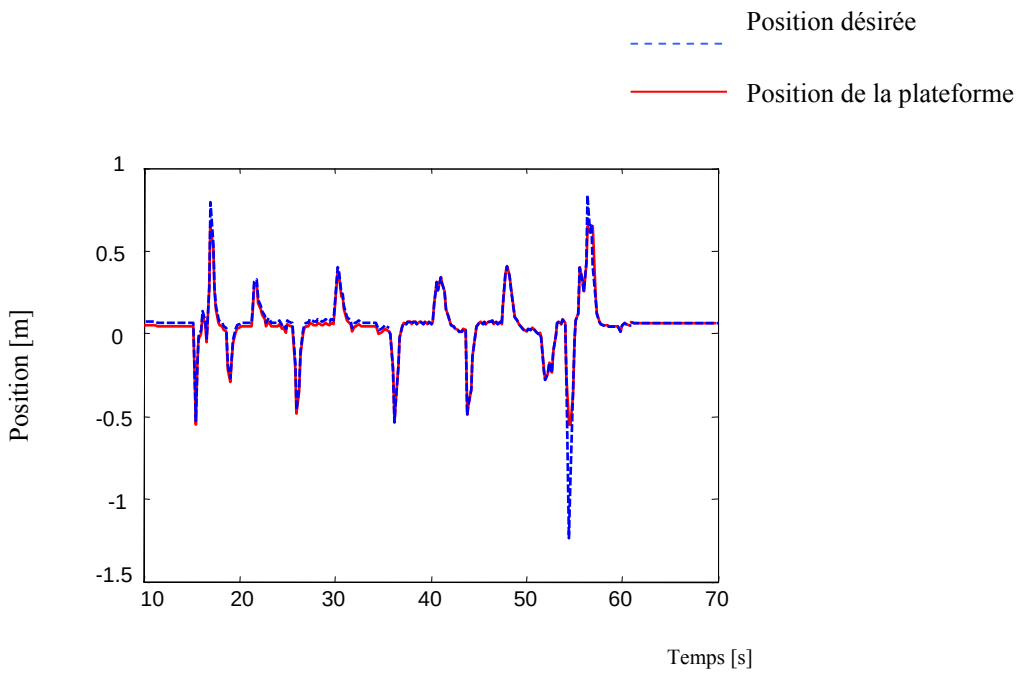


Fig.V.10. : Position désirée et position de la plateforme

Les performances de cette loi de commande sont montrées dans les courbes de la Fig.V.10. D'après ces courbes, il est bien montré que cette loi de commande permet de restituer le mouvement longitudinal en respectant les contraintes sur l'espace de déplacement et en faisant revenir la plate forme à sa position neutre, durant les phases constantes du signal d'accélération.

V.5. Conclusion

Dans ce chapitre, nous avons présenté les différentes structures des plates-formes mobiles, utilisées dans les simulateurs de conduite. Une plateforme mobile à deux degrés de libertés conçue conjointement par l'INRETS et LSC a été présentée. Cette plateforme est utilisée pour tester les stratégies de restitution de mouvement dans un simulateur de conduite. Nous nous sommes spécialement intéressé aux stratégies de restitution du mouvement longitudinal. L'objectif ici est de créer une stratégie optimale pour les situations de conduite en fil.

Après avoir détailler le modèle dynamique de la plateforme, nous avons décrit les étapes de la restitution du mouvement longitudinal, à savoir, extraction du signal de la position désirée, utilisation du filtre washout, commander l'actionneur. Enfin, nous avons vérifié les performances de cette plate-forme mobile en employant un contrôleur prédictif.

Conclusion générale

Le travail dont nous avons rendu compte dans le présent mémoire porte sur une stratégie de commande à horizon fuyant, la commande prédictive généralisée GPC, qui par sa simplicité a fait l'objet d'un grand intérêt dans le domaine industriel. La version linéaire de cette loi de commande a été présentée : sa philosophie, ses concepts et sa mise en œuvre. Les performances réalisées par cette stratégie de commande ont été prouvées à travers un exemple de servomécanisme.

Nous avons aussi exposé les éléments essentiels qui permettent de comprendre pourquoi, et dans quels cas, il est avantageux de mettre en oeuvre des réseaux de neurones. Les propriétés intéressantes des réseaux de neurones qui résident dans les points suivants :

- *outils statistiques, qui permettent d'ajuster des fonctions non linéaires très générales à des ensembles de points ;*
- *approximateurs parcimonieux ;*
- *permettent de modéliser des phénomènes statiques (réseaux non bouclés) et dynamiques (réseaux bouclés) ;*

ont fait des réseaux de neurones de meilleurs candidats pour le développement de la version non linéaire de la GPC. En effet, deux méthodologies de mise en œuvre de la GPC non linéaire ont été introduites : l'une fondée sur l'apprentissage d'un contrôleur prédictif neuronal, la seconde requérant l'apprentissage d'un modèle neuronal pour l'extraction à chaque pas d'échantillonnage d'un modèle linéaire du système autour d'un (ou plusieurs) point (s) de fonctionnement . Les performances de ces deux techniques de commande ont été montrées à travers un exemple illustratif, et les avantages et inconvénients de chaque technique ont été énoncés.

La puissance de cette loi de commande prédictive est mise en valeur par une application réelle « la commande prédictive d'une plate forme mobile d'un simulateur de conduite d'automobile ». Une description et une modélisation de la plateforme ont été introduites, et une implémentation du contrôleur prédictif pour la commande du mouvement longitudinal de cette plateforme a été réalisée. Les résultats obtenus montrent l'efficacité de cette loi de commande dans ce genre d'application. En effet, nous avons pu intégrer les contraintes sur la position de la plate forme dans notre contrôleur et supprimer les contacts de fin de courses utilisées dans l'architecture de commande classique. Les contacts de fin de courses présentent l'inconvénient de provoquer la coupure de l'alimentation et l'arrêt total de la plateforme.

Dans les travaux à venir, nous proposons d'intégrer la commande du dossier du siège pour réaliser la perception des opérations d'accélération, décélération et de freinage, et à équiper le dessous du siège par deux vérins pour assurer les situations des virages et aussi à introduire un système vibratoire sous le siège afin de restituer le mouvement dû aux irrégularités de la chaussée.
La nature non linéaire du modèle du siège, nous encourage à exploiter les techniques de commande prédictive à base de réseaux de neurones qui ont été abordées dans ce modeste travail.

Références bibliographiques

Ackley D., Hinton G. and Sejnowski T., A learning algorithm for Boltzmann machines, *Cognitive Science, Vol. 9,* 1985, pp.147-169.

Alberto B., **Edoardo** M., Fulfilling Hard Constraints in Uncertain Linear Systems by Reference Managing, *Dipartimento di Sistemi e Informatica, Universita di Firenze, Via di S. Marta 3, 50139 Firenze, Italy{*bemporad|mosca*}*@dsi.ing.unifi.it

Astrôm K.J. and Wittenmark B., Computer Controlled Systems. Theory and Design, *Prentice Hall Inc., 2nd Edition, Englewood Cliffs, New Jersey,* 1989.

Battiti R. , First and Second Order Methods for Learning: Between Steepest Descent Methods and Newton's Methods, *Neural Computation, Vol. 4, No.2,* 1992, pp.141-166.

Baum E.B. and Haussler D., What size net gives valid generalization, *Neural Computing, Vol. 1(1),* 1989, pp.151-160.

Billings S.A., H.B., Jamaluddin H.B. and Chen S., Properties of Neural Networks With Applications to Modelling non-linear Dynamical Systems, *Int. Journal of Control, Vol. 55, No 1,* 1992, pp.193-224.

Bishop C., Neural Networks for Pattern Recognitions, *Clarendon Press-Oxford New-York,* 1995.

Boucher P., La commande prédictive, *Editions Technip,* 1996.

Bruenelli R., Training nets through stochastic minimization. *Neural Networks, Vol. 7(9),* 1994, pp.1405-1412.

Bryson A.E. Jr. and Yu C.H., Applied Optimal Control. Optimization, Estimation and Control, *by Ginn and Company Waltham, Massachusetts,* 1969.

Cebuhar W.A. and Costanza V., Non Linear Control of CSTR's , *Chem. Eng. Science, 39,* 1984, pp.1715-1722.

Cybenko G., Approximation by superposition of a sigmoid function. *Math. of Control, Signals and Systems, Vol. 2(4),* 1989, pp.303-314.

Camacho E.F., and Bordons C., Model Predictive Control, *Springer-Velag,* 1999, London

Clarke D.W., Mohtadi C. and Tuffs P.S., Generalized Predictive Control, *Part 1 and Part 2, Automatica, 2 3*, 1987, pp.137-160.

Davalo E., Naïm P., Des réseaux de neurones, *Eyrolles*, 1989

De keyser R., Van cauwenberghe A., Extended Prediction Self Adaptive Control, *IFAC Symp.Ident.Syst.Param.Est., York,* 1985.

Dreyfus G., Idan Y., The Canonical Form of Non-linear Discrete-Time Models, *Neural Computation Vol. 10, No. 1,* 1998, pp.133-164.

Fausset L., Fundamentals of Neural Networks. Architecture - Algorithms and Applications, *1st Edition, Printice Hall, Inc.*, 1994.

Fletcher R., Practical methods of optimisation, *John Wiley and Sons Ltd, second edition, New-York, USA,* 1987.

Friedland B., Control system design. An introduction to state space methods, *McGraw-Hill Company, New York,* 1987.

Hagan M.T., Menhaj M., Training Feed-forward Networks with the Marquardt Algorithm, *IEEE Transactions on Neural Networks, 5*(6), 1994, pp.989-993.

Hambrecht A. et Robin P., Intégration d'une régulation adaptative neuronale dans un environnement industriel d'automatismes , *REE, Vol. n° 1,* janvier 1998.

Hassibi B. and Stork D. G., Second order derivatives for network pruning : Optimal brain surgeon. *In S.J. Hanson, J.D. Cawnan, and C.L. Giles, Editors, Advances in Neural Information Processing Systems 5*, Morgan Kaumann, San Mateo, CA, 1993, pp. 164-171.

Haykin S., Neural networks. A comprehensive foundation, *Macmillan College Publishing Company*, 1994

Hebb D., The Organization of Behavior, *New York: Wiley*, 1949.

Hecht-Nielsen R., Neurocomputing, *Addison-Wesley Publishing Company*, 1990

Hertz J., Krogh A. and Palmer R.G., Introduction to the theory of neural computation, *Computation and neural systems series*. Addison-Wesley, New-York, NY, 1991.

Hopfield J.J., Neural networks and physical systems with emergent collective computational abilities, *Proceedings of the National Academy of Sciences, Vol. 79*, 1982, pp.2554-2558.

Hornik K., Stinchcombe M. and White H., Multilayer Feedforward Networks are Universal Approximators, *Neural Networks, Vol. 2,* 1989, pp.359-366.

Hornik K., Stinchcombe M. and White H., Universal approximation of an unknown mapping and its derivatives using multiplayer feedforward networks, *Neural Networks, Vol. 3 .*, 1990, pp.551-560.

Hsu Y.L., Liang H.P., and Tsai S.J., An Improvement Response for CSC No.1 HSM, *IEEE Transactions on industry Applications, Vol. 36, No 3*, May/June 2000, pp.854-860.

Jacobs R.A., Increased rates of convergence through learning rate adaptative, *Neural Networks, Vol. 1*, 1988, pp.295-307.

Kemeny A., Simulation et perception du movement, *DSC 99 'driving simulation conference'*, Paris, France, 1999, pp33-p55.

Kheddar A., Garrec P.h., Architectures de plates– formes mobiles pour simulateurs de conduite automobile,2002, *CRIIF*

Konno Y., Shioya M., and Ueyama T., Development and Application of Dynamic System Simulator, *Nippon Steel Technical Report, No. 67*, October 1995, pp.63-68.

Lengellé R. and Denoeux T. Training MLPs layer by layer using an objective function for internal representations. *Neural Networks*, 9, 1996, pp.83-97.

Le Cun Y., Une procédure d'apprentissage pour réseau à seuil asymérique. *In Cognitiva 85: A la Frontière de l'Intelligence Artificielle des Sciences de la connaissance des Neurosciences*, Paris: CESTA, Paris, 1985, pp.599-604.

Le Cun Y., Boser B., Denker J.S., Hendersen D., Howard R.E., Hubard W., Jackel L.D., Backpropagation applied to handwritten zip code recognition. *Neural Computation, Vol. 1*, 1989, pp.541-551.

Leontaritis I.J. and Billings S.A., Input-output parametric models for non-linear systems–Part 1: Deterministic non-linear systems; Part 2: Stochastic non-linear systems, *International Journal of Control, Vol. 41*, 1985, pp.164-168.

Levenberg K., A Method for the Solution of Certain Non-linear Problems in Least Squares, *Quarterly Journal of Applied Mathematics II (2)*, 1944, pp.164-168.

Linden A. and Kindermann K., Inversion of multilayered nets, *Int. Joint. Conf. On Neural Networks (Wachington D.C.)*, June 1989, pp.425-430.

Ljung L., System Identification ; Theory for the User, *Prentice Hall, Englewood Cliffs, New Jersey*, 1987.

Marie M. et Mokhtari M. , Application de MATLAB© Ver. 5 et SIMULINK Ver. 2, Chapitre IV, *Springer-Verlag, France*, 1998.

Marquardt D., An Algorithm for Least-Squares Estimation of Nonlinear Parameters, *SIAM J. Appl. Math. 11*, 1963, pp.164-168.

McCulloch W. and W. Pitts, A logical calculus of the ideas immanent in nervous activity, *Bulletin of Mathematical Biophysics, Vol. 5*, 1943, pp.115-133.

Minoux M., Programmation Mathématique : Théorie et Algorithmes, *Ed. Dunod,* 1983.

Minsky M., et Papert S., Perceptrons, *MIT Press, Cambridge, MA*, 1969.

Mohellebi H., Espié S., Arioui H., Amouri A. and Kheddar A., Low cost motion platform for driving simulator, 5th international conference on machine automation, *ICMA'04*, 2004, *Osaka, Japan*

Narendra K.S. and Parthasarathy K., Identification and Control of Dynamical Systems using Neural Networks, *IEEE Trans. on Neural Networks, Vol. NO. 1*, 1990, pp.4-27.

Nash J.C., Compact Numerical Methods for Computers : Linear Algebra and Function Minimization, *Adam-Hilger Ltd, Bristol,* 1980.

Neelakantan R. and Guiver J., Applying Neural Networks, *Hydrocarbon Processing , Gulf Publishing Company, Houston,* September 1998, pp.91-96.

Nehaoua L., Amouri A. and Arioui H., *Classic and Adaptive Washout Comparison for a Low Cost Driving Simulator* 13th Mediterranean Conference on Control and Automation (MED), IEEE, 27-29 Juin 2005.

Nerrand O., Réseaux de neurones pour le filtrage adaptatif, l'identification et la commande de processus, *Thèse de Doctorat de l'Université Pierre et Marie Curie, Paris VI*, 1992.

Norgaard M., Neural networks for modelling and control of dynamic systems, *springer-Verlag London, 2001*.

Oussar Y., Réseaux d'ondelettes et réseaux de neurones 0pour la modélisation statique et dynamique de processus, *Thèse de Doctorat de l'Université Pierre et Marie Curie, Paris VI*, 1998.

Parker D.B., Learning-logic, *Technical Report TR-47, Center for Computational Research in Economics and Management Sci., MIT,* April, 1985.

Plaut D.C., Nowlan S.J. and Hinton G.E., Experiments on learning by back-propagation, *Technical Report CMU-CS-86-126, Carnegie Mellon University - Pittsburgh, PA 15213,* 1990.

Psaltis D., Siders A. and Yamamura A., A Multilayered Neural Network Controller, *IEEE Control System Magazine,* April 1988, pp.17-21.

Ramond G., Contribution à la commande predictive généralisée adaptative et application, *Université paris XI, U.F.R.Scientifique d'orsay* 2001.

Reymond G., Kemeny A, Droulez J., Berthoz A., Contribution of motion platform to kinesthetic restitution in a driving simulator, 1999, *DSC2000,* Paris, France, pp 33-55

Reymond G., Kemeny A., Motion cueing in the Reneault Driving Simulator, *Vehicle System Dynamics,* 2000, pp.249-259

Reymond G., Contribution respective des stimuli visuels, vestibulaires et proprioceptifs dans la perception du mouvement du conducteur, 2000, P*aris VI University thesis (in French*).

Rivals I., Modélisation et commande par réseaux de neurones : application au pilotage d'un véhicule autonome, *Thèse de Doctorat de l'Université Pierre et Marie Curie, Paris VI,* 1995.

Robbins H. and Monro S., A stochastic approximation method, *Annals of Math. Stat., Vol. 22,* 1951, pp.400-407.

Rosenblatt F., The Perceptron: a Perceiving and Recognizing Automaton, *Project PARA, Report 85-460-1, Cornell Aeronautical Lab.*, 1957.

Rosenblatt F., The Perceptron: a probabilistic model for information storage and organization in the brain, *Psychological Review, Vol. 65,* 1958, pp.386-408.

Saidi M.L., Kermiche S., Abbassi H.A., Arbaoui F, Neural generalized predictive control (study and simulation) », *Conférence nationale sur l'ingénierie de l'electronique, CNIE'04,* Oran, novembre 2004.

Saidi M.L., Kermiche S., Debbeh A., Arbaoui F., Abbassi H.A., Neural networks in predictive control. *Conférence Internationale sur la productique, CIP'05, Tlemcen, Décembre 2005.*

Saidi M.L., Debbeh A., Arioui H., Kermiche S., Abbassi H.A., Predictive control of motion platform in driving simulator, *Asian journal of information technology, ISSN 1682-3915, Vol.5, Number 2, 2006,* pp. 133-138

Seigler I., Kemeny A., 2001, Etude sur la pertinence de la restitution physique du mouvement en simulation de conduite en fonction des caractéristiques physiologiques et psychophysiques de la perception du mouvement propre, 2001.

Soeterboek R., Predictive control. A unified Approach, *Prentice-Hall,* 1992

Sorensen O., Neural Networks in Control Applications, *Ph.D. Thesis. Aalborg University, Department of Control Engineering,* 1994.

Taylor J.G., The promise of neural networks, *Springer-Verlag*, 1993

Thomas P. and Bloch G., Robust Pruning for Multilayer Perceptrons. *In Proceedings of IMACS/IEEE Multiconference on Computational Engineering in Systems Applications CESA'98, Vol. 4, Nabeul-Hammamet, Tunisia,* April 1-4, 1998, pp.17-22.

Tollenaere T., SuperSAB: fast adaptive back propagation with good scaling properties, *Neural Networks, Vol. 3,* 1990, pp.561-573.

Vermeulen W., Bodin A., and Van Der Zwaag S., Prediction of the Measured Temperature after the last Finishing Stand Using Artificial Neural Networks, *Steel Research 68*(1), 1997,pp.20-26.

Urbani D., Méthodes Statistiques de Sélection d'Architectures Neuronales: Application à la Conception de Modèles de Processus Dynamiques, *Thèse de Doctorat de l'Université Pierre et Marie Curie, Paris VI,* 1995.

Walter E. et Pronzato L., Identification de modèles paramétriques à partir de données expérimentales, *Editions Masson, Paris,* 1994.

Warwick K., An overview of Neural Networks in Control Applications, *In "Neural Networks for Robotic Control, Theory and Applications ", Edited by Zalzala A.M.S. and Morris A.S., First published by Ellis Horwood Ltd.,* 1996, pp.1-25.

Weerasooriya S., Sharkawi M.A., Identification and Control of a DC Motor Using Back-Propagation Neural Networks , *IEEE Trans. On Energy Conversion, 6*(4), 1991, pp.363-369.

Werbos P., Beyond regression: new tools for prediction and analysis in the behavioural sciences, *PhD thesis, Harvard University, Cambridge, MA.,* 1974.

Widrow B. and Hoff M., Adaptive switching circuits, *WESCON Convention Record, New York: IRE, Vol. 4,* 1960, pp.96-104.

Wu J. K., Neural Networks and Simulation Methods, *1st Edition, Marcel Dekker, Inc.*, 1994.

Ydestie B., Extended Horizon Adaptive Control, *Proceeding of the IFAC 9th World Congress, Paper 14.4 / E-4, Budapest,* 1984.

Zietsman J.H., Kumar S., Meech J.A., Samarasekera I.V., and Brimacombe J.K., Taper Design In Continuous Billet Casting Using Artificial Neural Networks, *Ironmaking and Steelmaking, 25*(6), 1998, pp.476-483.

Annexe

A.1. La méthode du gradient simple

A.1.1. Présentation de la méthode

La méthode du gradient simple consiste à la mise en oeuvre de la formule de mise à jour des paramètres suivante :

$$\theta^k = \theta^{k-1} - \mu_k \nabla J\left(\theta^{k-1}\right) \qquad \textbf{(A.1)}$$

La direction de descente est donc simplement l'opposée de celle du gradient ; c'est en effet la direction suivant laquelle la fonction de coût diminue le plus rapidement.

En pratique, la méthode du gradient simple peut être efficace lorsque l'on est loin du minimum de *J*. Quand on s'en approche, la norme du gradient diminue et donc l'algorithme progresse plus lentement. A ce moment, on peut utiliser une méthode de gradient plus efficace.

Un réglage du pas de gradient μ_k est nécessaire : en effet, une petite valeur de ce paramètre ralentit la progression de l'algorithme ; en revanche une grande valeur aboutit généralement à un phénomène d'oscillation autour de la solution. Diverses heuristiques, plus ou moins efficaces, ont été proposées.

A.1.2. Techniques de réglage du pas

- ***Technique du pas constant* :** elle consiste à adopter un pas constant $\mu_k = \mu$ tout au long de l'algorithme. Elle est très simple mais peu efficace puisqu'elle ne prend pas en considération la décroissance de la norme du gradient.
- ***Technique du pas asservi* :** on peut asservir le pas à l'aide de la norme du gradient de sorte que le pas évolue en sens inverse de celle-ci. A chaque étape, le pas peut être calculé par :

$$\mu_k = \frac{\mu}{1 + \|\nabla J\|} \qquad \textbf{(A.2)}$$

où μ est un paramètre constant. Lors de l'utilisation de la technique du pas asservi, l'adoption de la valeur $\mu = 10^{-3}$ se révéle très souvent satisfaisante.
Le dénominateur est augmenté du nombre 1 afin d'éviter une instabilité numérique au moment de la division dans le cas où la norme du gradient devient très proche de zéro. Cette technique offre un bon compromis du point de vue de la simplicité et de l'efficacité.

A.2. Les méthodes de gradient du second ordre

Les méthodes que nous venons de décrire sont simples mais en général inefficaces. On a donc systématiquement recours à l'utilisation de méthodes plus performantes [Battiti, 1992]. Elles sont dites du second ordre parce qu'elles prennent en considération la dérivée seconde de la fonction de coût.

A.2.1. Algorithme de Newton

L'algorithme de Newton consiste à modifier les paramètres à chaque itération selon :

$$\theta^k = \theta^{k-1} - \left[H\left(\theta^{k-1}\right) \right]^{-1} \nabla J\left(\theta^{k-1}\right) \qquad \textbf{(A.3)}$$

La direction d(i) est une direction de descente seulement si la matrice hessienne est définie positive.

A.2.2. Algorithme de Levenberg-Marquardt

L'algorithme de Levenberg-Marquardt [Levenberg, 1944], [Marquardt, 1963] repose sur l'application de la formule de mise à jour des paramètres suivante :

$$\theta^k = \theta^{k-1} - \left[H\left(\theta^{k-1}\right) + \mu_k I \right]^{-1} \nabla J\left(\theta^{k-1}\right) \qquad \textbf{(A.4)}$$

où $H(\theta^{\kappa-1})$ est le Hessien de la fonction de coût et μ_k est le pas. Pour de petites valeurs du pas, la méthode de Levenberg-Marquardt s'approche de celle de Newton. Inversement, pour de grandes valeurs de μ_k, l'algorithme Levenberg-Marquardt est équivalent à l'application de la règle du gradient simple avec un pas de ($1/\mu_k$).
La première question relative à cet algorithme est celle de l'inversion de la matrice $\left[H\left(\theta^{k-1}\right) + \mu_k I \right]$. L'expression exacte du Hessien de la fonction *J* est :

$$H\left(\theta^k\right) = \sum_{n=1}^{N} \left(\frac{\partial e^n}{\partial \theta^k} \right) \left(\frac{\partial e^n}{\partial \theta^k} \right)^T + \sum_{n=1}^{N} \frac{\partial^2 e^n}{\partial \theta^k \partial \left(\theta^k\right)^T} e^n \qquad \textbf{(A.5)}$$

avec : $e^n = y_p^n - y^n$.

Le second terme de l'expression étant proportionnel à l'erreur, il est donc permis de le négliger en première approximation, ce qui fournit une expression approchée :

$$\tilde{H}\left(\theta^k\right) = \sum_{n=1}^{N}\left(\frac{\partial e^n}{\partial \theta^k}\right)\left(\frac{\partial e^n}{\partial \theta^k}\right)^T = \sum_{n=1}^{N}\left(\frac{\partial y^n}{\partial \theta^k}\right)\left(\frac{\partial y^n}{\partial \theta^k}\right)^T \qquad \textbf{(A.6)}$$

Dans le cas d'un modèle linéaire par rapport aux paramètres, en d'autres termes, si y est une fonction linéaire de θ, le second terme de l'expression de H est nul est l'approximation devient exacte. Plusieurs techniques sont envisageables pour l'inversion de la matrice [$\tilde{H} + \mu_k I$] [Friedland, 1987], [Gourdin et Boumahrat, 1991].

A.2.3. Algorithme de Broyden, Fletcher, Goldfarb et Shanno (BFGS)

L'algorithme de BFGS, du nom de ses inventeurs : **B**royden, **F**letcher, **G**oldfarb et **S**hanno, [Minoux, 1983] fait partie des méthodes d'optimisation dites « quasi-newtoniennes ». Ces méthodes sont une généralisation de la méthode de Newton. La méthode de Newton consiste à l'application de la règle suivante :

$$\theta^k = \theta^{k-1} - \left[H\left(\theta^{k-1}\right)\right]^{-1} \nabla J\left(\theta^{k-1}\right) \qquad \textbf{(A.7)}$$

où $H(\theta)$ est le Hessien de la fonction J calculé avec le vecteur des paramètres disponible à l'étape courante. La direction de descente est dans ce cas :

$$d_k = -\left[H\left(\theta^{k-1}\right)\right]^{-1} \nabla J\left(\theta^{k-1}\right) \qquad \textbf{(A.8)}$$

Le pas μ_k est constant et égal à 1.

Pour que le déplacement soit dans le sens contraire du gradient, il est indispensable que la matrice du Hessien soit définie positive. Sous cette condition, et si la fonction de coût est quadratique par rapport aux paramètres, la méthode de Newton converge vers l'unique solution en une seule itération.

En général, la fonction de coût n'est généralement pas quadratique. Elle peut néanmoins l'être localement, à proximité d'un minimum de ses minima. Donc, la méthode de Newton ne peut converger en une seule itération. De plus, cette méthode nécessite l'inversion de la matrice du Hessien à chaque itération, ce qui conduit à des calculs lourds.

L'algorithme de BFGS, ainsi que l'algorithme de Levenberg-Marquardt présenté dans le paragraphe suivant, sont des méthodes quasi-newtoniennes qui permettent de pallier ces inconvénients.

L'algorithme de BFGS est une règle d'ajustement des paramètres ayant l'expression suivante :

$$\theta^k = \theta^{k-1} - \mu_k M_k \nabla J\left(\theta^{k-1}\right) \qquad \textbf{(A.9)}$$

où M_k est une approximation, calculée itérativement, de l'inverse de la matrice Hessienne.

L'approximation de l'inverse du Hessien est modifiée à chaque itération suivant la règle suivante :

$$M_k = M_{k-1} + \left[1 + \left(\frac{\gamma_{k-1}^T M_{k-1} \gamma_{k-1}}{\delta_{k-1}^T \gamma_{k-1}}\right)\right] \frac{\delta_{k-1}^T \delta_{k-1}}{\delta_{k-1}^T \gamma_{k-1}} - \frac{\delta_{k-1} \gamma_{k-1}^T M_{k-1} + M_{k-1} \gamma_{k-1} \delta_{k-1}^T}{\delta_{k-1}^T \gamma_{k-1}}$$

(A.10)

avec : $\gamma_{k-1} = \nabla J\left(\theta^k\right) - \nabla J\left(\theta^{k-1}\right)$ et $\delta_{k-1} = \theta^k - \theta^{k-1}$

Nous prenons pour valeur initiale de M la matrice identité. Si, à une itération, la matrice calculée n'est pas définie positive, elle est réinitialisée à la matrice identité.
Reste la question du choix du pas μ_k. A cet effet, une méthode économique en calculs est souvent recommandée dans la littérature : la technique de [Nash, 1980]. Cette technique recherche un pas qui vérifie la condition de descente :

$$J\left(\theta^{k-1} + \mu_k d_k\right) \leq J\left(\theta^{k-1}\right) + m_1 \mu_k d_k^T \nabla J\left(\theta^{k-1}\right) \qquad \textbf{(A.11)}$$

où m_1 est un facteur choisi très inférieur à 1 (par exemple $m_1 = 10^{-3}$).

En pratique, la recherche du pas se fait de manière itérative. On initialise μ_k à une valeur positive arbitraire. On teste la condition (A.11). Si elle est vérifiée, on accepte l'ajustement des paramètres. Sinon, on multiplie le pas par un facteur inférieur à 1 (par exemple 0.2) et on teste à nouveau la condition de descente.
On répète cette procédure jusqu'à ce qu'une valeur satisfaisante du pas soit trouvée. Si au bout d'un certains nombre d'essais, le pas atteint une valeur très petite (de l'ordre de 10^{-16}, par exemple), on peut considérer alors qu'il n'est pas possible de trouver un pas satisfaisant.

Une méthode « quasi-newtonienne », n'est efficace que si elle est appliquée au voisinage d'un minimum.
D'autre part, la règle du gradient simple est efficace lorsqu'on est loin du minimum et sa convergence ralentit considérablement lorsque la norme du gradient diminue (en d'autres termes, lorsqu'on s'approche du minimum). Ces deux techniques sont donc complémentaires.

De ce fait, l'optimisation s'effectue en deux étapes : utilisation de la règle du gradient simple pour s'approcher d'un minimum, et de l'algorithme de BFGS pour l'atteindre.