REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MENTOURI-CONSTANTINE
FACULTE DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'ELECTRONIQUE

THESE

Présentée pour l'obtention du diplôme de

DOCTORAT EN SCIENCES

Électronique

OPTION

Contrôle des systèmes

Par

Ghania Debbache

Contribution à la commande neuronale adaptative des systèmes non linéaires incertains

Soutenue le 13/11/2011 devant le jury composé de:

Dr. Mohamed Khamadja	Prof., Université de Constantine	Président
Dr. Abdelhak Bennia	Prof., Université de Constantine	Rapporteur
Dr. Abdelfatah Charef	Prof., Université de Constantine	Examinateur
Dr. Ahmed Louchene	MCA, Université de Batna	Examinateur
Dr. Nacereddine Nait Said	Prof., Université de Batna	Examinateur

Remerciements

Tout d'abord, je tiens à remercier mon promoteur Dr. Abdelhak Bennia pour sa sagesse, sa générosité et sa patience infinie qui m'ont aidé à finaliser ce travail de thèse.

Je suis reconnaissante envers Dr. Noureddine Goléa pour son aide et ses conseils techniques.

Je tiens à exprimer ma sincère gratitude aux membres du Jury pour avoir pris le temps de lire et d'évaluer ce travail.

Résumé

Les méthodes de commande non linéaire se basent sur l'existence d'un modèle analytique du système. Cependant, pour des systèmes non linéaires complexes, le modèle peut être inexploitable, imprécis ou tout simplement inexistant. D'un autre côté, la commande adaptative présente un outil puissant pour la commande des systèmes linéaires et pour la réduction de l'incertitude paramétrique. Les réseaux de neurones sont apparus, récemment, comme des approximateurs ajustables et capables de reproduire le comportement complexe des systèmes non linéaires. Dans ce travail de thèse, nous combinerons la commande adaptative et les réseaux de neurones pour profiter de leurs caractéristiquement communes dans la solution des problèmes de la commande des systèmes non linéaires incertains. Ce travail de thèse décrit trois contributions à la commande neuronale des systèmes non linéaires. La première contribution étend la commande adaptative à modèle de référence aux systèmes non linéaires incertains. La deuxième contribution, élimine la nécessité de mesurer touts les états du système par l'introduction d'un observateur pour estimer les états non mesurables. La troisième contribution présente une commande neuronale adaptative indirecte, applicable à une large classe de systèmes non linéaires multivariables. Les résultats théoriques obtenus sont vérifiés par simulation sur des systèmes non linéaires instables et chaotiques.

Mots clés: Systèmes non linéaires, Commande adaptative, Réseaux de neurones, Approximation universelle, MRAC, Observateur, incertitudes, Stabilité.

Table des matières

Notations

Acronymes

MLP : réseau de neurones perceptron multicouches.
RBF : fonction radiale de base
SISO : mono-entrée mono-sortie
MIMO : multi-entrées multi-sorties
LTI : linéaire invariant dans le temps
MRAC : commande adaptative à modèle de référence
SPR : réelle strictement positive
LMI : inégalité matricielle linéaire

Symboles

A travers cette thèse, nous emploierons les notations suivantes :
$\Re$ Ensemble des réels
a Scalaire
$\underline{a}$ Vecteur
A Matrice
$|a|$ Valeur absolue
$|\underline{a}|$ Norme-2 d'un vecteur (norme Euclidienne)
$|A|$ Norme-2 d'une matrice.
n Ordre du système.
p Nombre des entrées et des sorties pour un système MIMO.
s Variable de Laplace
$\underline{x} \in \Re^n$ Vecteur d'état
$\underline{\widehat{x}} \in \Re^n$ Vecteur d'état estimé
$u \in \Re$ Entrée scalaire de commande d'un système SISO
$\underline{u} \in \Re^p$ Vecteur des entrées de commande d'un système MIMO
$y \in \Re$ Sortie scalaire d'un système SISO
$\underline{y} \in \Re^p$ Vecteur des sorties d'un système MIMO
$\underline{x}_m \in \Re^n$ Vecteur d'état de référence
$y_m \in \Re$ Sortie scalaire de référence
$A \in \Re^{n \times n}, \underline{b} \in \Re^{n \times 1}, \underline{c} \in \Re^{1 \times n}$ Matrices d'état d'un système SISO
$A \in \Re^{n \times n}, B \in \Re^{n \times p}, C \in \Re^{p \times n}$ Matrices d'état d'un système MIMO
$A_m \in \Re^{n \times n}$ Matrice d'évolution du modèle de référence

$e \in \Re$ Erreur scalaire
$\underline{e} \in \Re^{n\times 1}$ Vecteur d'erreurs
$\widehat{\underline{e}} \in \Re^{n\times 1}$ Vecteur d'erreurs estimées
$e_y \in \Re$ Erreur de sortie
A^T Transposée
$\mathrm{tr}(A)$ Trace d'une matrice
L_∞ Norme infinie pour un signal
L_2 Norme-2 pour un signal
V Fonction de Lyapunov
$P \in \Re^{n\times n}, Q \in \Re^{n\times n}$ Matrices définies positives.
$f(\underline{x}), g(\underline{x}) \in \Re$ Fonctions non linéaires scalaires d'un système SISO
$\underline{f}(\underline{x}) \in \Re^{p\times 1}$ Vecteur de fonctions non linéaires d'un système MIMO
$G(\underline{x}) \in \Re^{p\times p}$ Matrice de fonctions non linéaires d'un système MIMO
$d(\underline{x}, t) \in \Re$ Incertitude d'un système SISO
$\underline{d}(\underline{x}, t) \in \Re^{p\times 1}$ Vecteur des incertitudes d'un système MIMO
$k, \underline{k}, K$ Gains scalaire, vecteur et matrice, respectivement.
$k^*, \underline{k}^*, K^*$ Gains optimaux
$\widetilde{k}, \underline{\widetilde{k}}, \widetilde{K}$ Erreurs sur les gains
$\varphi_i(\underline{x})$ Fonction d'activation d'un réseau de neurones
$\underline{\phi}(\underline{x})$ Vecteur de fonctions d'activation
$\Phi(\underline{x})$ Matrice de fonctions d'activation
$\underline{\theta}, \Theta$ Vecteur et matrice de paramètres ajustables, respectivement.
$\underline{\theta}^*, \Theta^*$ Paramètres ajustables optimaux
$\underline{\epsilon}(\underline{x}), \underline{\varepsilon}(\underline{x})$ erreur d'approximation d'un réseau de neurones
$\underline{\widetilde{\theta}}, \widetilde{\Theta}$ Erreurs d'estimation sur paramètres ajustables
$\Omega_{\underline{x}}$ Zone de contrainte pour les états
$\Omega_{\underline{\theta}}$ Zone de contrainte pour les paramètres

Liste des figures

Chapitre 1

Introduction

1.1 Commande adaptative non linéaire : Motivation et vue d'ensemble

La recherche dans la théorie de la commande des systèmes non linéaires a été motivée par les caractéristiques non linéaires inhérentes des systèmes physiques que nous essayons souvent de commander. Les exemples de tels systèmes sont les systèmes mécaniques et électromécaniques, les limitations sur les commandes et la saturation, les systèmes couplés et interconnectés, pour ne citer que quelques uns. Si nous ajoutons à la nature non linéaire de la dynamique le fait que la plupart des systèmes ne sont pas bien connus et donc pas exactement modélisés, il est clair que les techniques de commande linéaires fassent défaut dans leurs aspects théoriques et pratiques. Bien que les systèmes linéaires soient très bien compris et commandés, la commande linéaire n'est pas assez pour garantir la stabilité et la performance des systèmes non linéaires.

Les deux dernières décennies ont été témoins des accomplissements significatifs de la théorie des systèmes dynamiques non linéaires et des applications réussies de la commande non linéaire [1-4]. Les théories avancées des systèmes non linéaires, telles que la théorie de la géométrie différentielle [3], révèlent la structure de la dynamique interne des systèmes non linéaires. Des méthodes de conception de commande non linéaires avancées, y compris la linéarisation par bouclage (Feedback linearization) [1, 3-4], l'inversion dynamique non linéaire [5-6], et le backstepping [2] ont été développées et appliquées avec succès aux problèmes de commande [7].

Ces méthodes avancées de conception de commandes non linéaires dépendent habituellement d'un modèle analytique du système dynamique, qui est, dans beaucoup d'applications de commande, souvent imprécis ou indisponible. Beaucoup de méthodes de contrôle non linéaires, basées sur la géométrie différentielle, nécessitent la différentiation répétée d'un champ de vecteur non linéaire ; ce qui entrave souvent leur efficacité et applicabilité. Inspirés par les applications réussies de la commande adaptative linéaire, quelques structures de commandes non linéaires adaptatives, telles que la linéarisation

par bouclage adaptatif [8] et la commande backstepping adaptative [2], ont été développées. Dans ces schémas de commande adaptative non linéaire, l'incertitude du système non linéaire est identifiée en ligne. La dynamique identifiée du système est employée pour améliorer la performance du contrôleur. Ces techniques de commande adaptative non linéaire permettent également l'utilisation d'un modèle dynamique simplifié dans la conception du contrôleur, alors que la composante adaptative compense automatiquement l'erreur de modélisation introduite par la simplification. Un inconvénient majeur de ces techniques de commande non linéaire adaptative est l'hypothèse que la dynamique inconnue possède une structure connue avec des paramètres inconnus entrant linéairement dans la dynamique. La paramétrisation linéaire de la dynamique inconnue pose des obstacles sérieux pour l'utilisation des algorithmes de commande adaptative dans des applications pratiques, parce qu'il est difficile voir impossible de fixer la structure des non linéarités inconnues. Ce fait a été le facteur de motivation derrière l'utilisation d'approximateurs non linéaires pour estimer les non linéarités inconnues.

1.2 Apport des réseaux de neurones

La capacité des réseaux de neurones à approximer des fonctions uniformément continues a été prouvée en plusieurs articles (voir p. ex. [9-15]). Les réseaux de neurones, avec la capacité d'approximer une grande classe de fonctions non linéaires, fournissent une structure canonique et faisable pour la représentation des systèmes dynamiques non linéaires. Ainsi, les réseaux de neurones sont un outil puissant que ce soit pour améliorer des applications quand le modèle mathématique du système non linéaire est disponible mais imprécis, ou pour établir un modèle du système à partir des données expérimentales lorsque un modèle théorique est indisponible ou impraticable. Au début des années 90, l'identification et la commande des systèmes non linéaires par des réseaux de neurones étaient proposées d'une manière systématique pour la première fois par Narendra et Parthasarathy [16]. Les résultats de cette première recherche ont été limités à la simulation, et aucune preuve de stabilité de la boucle fermée n'a été fournie. Une première analyse rigoureuse des réseaux de neurones dans des schémas d'identification et de commande peut être trouvée dans le travail de Polycarpou et Ioannou [17]. Sanner et Slotine ont proposé pour la première fois l'utilisation des réseaux à fonctions de base radiales (RBF) pour la commande des systèmes non linéaires avec une analyse de la stabilité de la boucle de commande [18]. Sadegh a proposé une commande adaptative stable à base d'un réseau de neurones avec des fonctions d'activation sigmoïdes [19]. Les premiers résultats, avec preuve de stabilité, de la linéarisation par bouclage adaptatif, en utilisant un réseau de neurones d'une seule couche cachée, ont été obtenus par Chen et Khalil [20] pour des systèmes discret, et par Chen et Liu [21] pour des systèmes continus. Lewis et al. [22-23] ont développé de nouvelles lois d'ajustement pour une commande adaptative neuronale en se basant sur l'analyse de Lyapunov. Calise et al. [24-25] ont développé une architecture de commande basée sur l'inversion dynamique avec un réseau de neurones linéairement paramétré dans la boucle de commande pour compenser l'erreur introduite par l'inverse approximatif. La littérature sur la commande neuronale est très abondante (voir p. ex.

[26-27]), et des résultats plus récents peuvent être trouvés dans de nombreuses revues spécialisées.

Dans la plupart des approches de commande par réseaux de neurones, le réseau de neurones, qui approxime la dynamique du système, est employé dans la conception et la synthèse du contrôleur. En employant les réseaux de neurones pour représenter une grande classe des incertitudes dans les systèmes non linéaires, la restriction sur la paramétrisation de l'incertitude dans la commande adaptative non linéaire est relaxée. Des algorithmes de commande adaptative, qui n'exigent pas la connaissance de la structure du système (sauf pour l'ordre et le degré relatif) ont été rendus possibles en utilisant les réseaux de neurones artificiels dans la boucle de commande. En outre, dans la commande adaptative par réseaux de neurones, les possibilités d'apprentissage des réseaux de neurones sont augmentées par la théorie de la commande non linéaire adaptative robuste. Ainsi la stabilité et la performance d'une telle commande adaptative par réseaux de neurones sont garanties.

Un aspect important des applications de la commande par réseaux de neurones est la différence entre les résultats de la théorie d'approximation et ce qui est réalisable dans des schémas adaptatifs utilisant de tels approximateurs. D'abord et d'une manière plus importante, dans des applications hors ligne les poids du réseau de neurones sont ajustés en se basant sur l'information entrée-sortie, tandis que dans des situations de commande adaptative l'ajustement des paramètres du réseau de neurones est conduit par une erreur de poursuite, qui par sa définition ne contient aucune information sur l'entrée. En plus de cet aspect, l'ajustement adaptatif doit garantir la bornitude de tous les signaux dans la boucle de commande. La rétro-propagation de l'erreur standard n'est pas généralement la seule composante dans la loi d'ajustement des paramètres.

1.3 Objectif et contributions

Le thème principal de la recherche présentée dans cette thèse est le développement de nouvelles structures de commande adaptive neuronale pour la commande des systèmes non linéaires incertains. Cet intérêt est justifié par le fait que les systèmes physiques sont, en une très grande majorité, des systèmes non linéaires sujets à des incertitudes de modélisation et des incertitudes paramétriques, ainsi qu'à des perturbations externes. Nous considérerons, dans ce travail, les systèmes non linéaires mono-variables (SISO) et multi-variables (MIMO) et les systèmes non linéaires avec des états non mesurables.

Les principales contributions de cette thèse sont :

1. Proposition d'une approche neuronale pour la commande à modèle de référence (MRAC) des systèmes non linéaires avec états mesurables.
2. Proposition d'une approche neuronale à base d'observateur pour la commande MRAC des systèmes non linéaires avec états non mesurables.

3. Proposition d'une approche neuronale pour la commande adaptative decouplée des systèmes non linéaires Multivariables.
4. Application de la commande neuronale decouplée aux robots manipulateurs.

Ces contributions ont fait l'objet des publications suivantes :

1. Ghania Debbache, Abdelhak Bennia, Noureddine Goléa, Neural network based state feedback Control of robot manipulators, TENCON 2006. IEEE Region 10 Conference, 14-17 Nov. 2006 Hong Kong.
2. Ghania Debbache, Abdelhak Bennia, Adaptive Neural Nonlinear State Feedback Control of Dynamic Nonlinear Systems, International Conference on Electrical Engineering Design & Technologies, Hammamet Tunisia, Nov. 4-6, 2007.
3. Ghania Debbache, Abdelhak Bennia, Noureddine Goléa, Neural network based MRAC control of dynamic non linear systems, Int. J. Appl. Math. Comput. Sci., 2006, Vol. 16, No. 2, 219–232.
4. Ghania Debbache, Abdelhak Bennia, Noureddine Goléa, Neural Networks-based Adaptive State Feedback Control of Robot Manipulators, International Journal of Computers, Communications & Control, 2007, No. 4, pp. 328-339.

1.4 Organisation de la thèse

Le reste de ce manuscrit est organisé comme suit :

Le chapitre 2 présente les éléments de base nécessaires à la justification et la compréhension du reste du travail. En premier, les problèmes posés par la commande des systèmes non linéaires sont définis. Un exemple illustratif très simple démontre l'incapacité des techniques de commande linéaire à résoudre ces problèmes et à produire des performances acceptables. Ce constat justifie à lui seul l'introduction des réseaux de neurones comme un élément clé de la commande adaptative des systèmes non linéaires. La deuxième partie du chapitre 2 introduit les propriétés essentielles des réseaux de neurones qui font que leur application dans la commande des systèmes non linéaires est aussi intéressante. Nous discuterons seulement les réseaux MLP et les réseaux RBF qui sont utilisés dans ce travail. La troisième partie du chapitre 2 fixe la classe des systèmes non linéaires étudiés et énonce les conditions et propriétés requises pour ces systèmes.

Le chapitre 3 introduit une technique de commande à modèle de référence (MRAC) pour une certaine classe de systèmes non linéaires. Cette commande est une extension de la commande MRAC des systèmes linéaires. L'introduction des réseaux de neurones dans la boucle de commande permet d'obtenir un retour d'état non linéaire, qui permet à la fois de compenser les non linéarités inconnues et les perturbations externes. L'analyse de stabilité montre que la dynamique de l'erreur de poursuite est stable même en présence des erreurs d'approximation. En effet en l'absence des erreurs d'approximation (approximation idéale) nous obtenons une convergence globale de l'erreur de poursuite vers zéro.

Pour éliminer la condition de la disponibilité des états du système requise dans les développements du chapitre 3, le chapitre 4 étudie l'utilisation d'un observateur d'erreur pour estimer les états nécessaires à l'implémentation de la commande MRAC neuronale. Les états sont estimés indirectement à travers l'estimation de l'erreur de poursuite. La condition SPR (réelle strictement positive) nécessaire à l'implantation des lois d'ajustement des paramètres est réalisée à travers le filtrage de la dynamique de l'erreur d'estimation. L'analyse de Lyapunov montre la stabilité de la boucle de commande augmentée par l'observateur. Les erreurs d'estimation et de poursuite restent bornées.

Le chapitre 5 présente une architecture de commande neuronale adaptative indirecte. Cette architecture est applicable à une large classe de systèmes non linéaires multivariables. Elle permet un calcul decouplé des entrées de commande en se basant sur une estimation neuronale de la dynamique du système contrôlé. L'analyse de stabilité assure la stabilité et la robustesse de la boucle de commande.

Les méthodes proposées dans ce travail sont appliquées à des benchmarks très populaires dans le domaine de la commande, à savoir, le pendule inversé (dont le modèle est analogue à celui d'un satellite, d'un missile ou d'une grue) et le robot manipulateur à deux degrés de liberté. Les résultats de simulations, sous différentes situations d'incertitudes et de perturbations, confirment les conclusions théoriques obtenus que ce soit pour les performances de poursuite ou la robustesse de la boucle de commande par rapport aux incertitudes et perturbations.

Les résultats de ce travail de recherche sont résumés dans le chapitre 6, où des conclusions sont présentées avec des directions possibles pour de futures recherches. Le matériel supplémentaire figure dans l'annexe.

Chapitre 2

Problèmatique, outils et définitions

2.1 Introduction

Le présent chapitre introduit les éléments de base nécessaires à la justification et la compréhension du reste du travail. Dans la section 2.2, les problèmes posés par la commande des systèmes non linéaires sont définis. Un exemple illustratif très simple démontre l'incapacité des techniques de commande linéaire à résoudre ces problèmes et à produire des performances acceptables. Ce constat justifie à lui seul l'introduction des réseaux de neurones comme un élément clé de la commande adaptative des systèmes non linéaires. La section 2.3 introduit les propriétés essentielles des réseaux de neurones, qui rendent leur application dans la commande des systèmes non linéaires aussi intéressante. Nous discuterons seulement les réseaux MLP et les réseaux RBF qui sont utilisés dans ce travail. La section 2.4 fixe la classe des systèmes non linéaires étudiés et énonce les conditions et propriétés requises pour ces systèmes. Enfin, la section 2.5 conclut le chapitre. Les éléments théoriques utilisés dans ce chapitre sont regroupés dans l'annexe.

2.2 Position du problème

La plupart des systèmes dynamiques rencontrés dans la pratique sont de nature non linéaire. Les méthodes de la commande linéaire peuvent parfois être appliquées aux systèmes non linéaires pour un domaine de fonctionnement bien limité, à travers la méthode de linéarisation. Cependant, le niveau de performance désirée ou les problèmes de poursuite sur un large domaine de fonctionnement nécessitent l'inclusion des non linéarités dans la conception de la commande. La non linéarité et la précision du modèle affectent directement la performance des systèmes de commande. La non linéarité peut imposer des contraintes sur la performance obtenue. Le défi de prendre en compte les non linéarités durant la conception de la commande est très compliqué lorsque la description des non linéarités est entachée d'incertitude, ou lorsqu'une partie du modèle est

inconnue, définie avec incertitude, ou change durant le fonctionnement. Pour mieux situer le problème posé par la commande des systèmes non linéaires, nous utiliserons un exemple d'un système non linéaire simple pour donner une appréciation quantitative des différents aspects du problème.

Considérons le système non linéaire scalaire :

$$\dot{x} = f(x) + u \tag{2.1}$$

où $x, u \in \Re$ sont respectivement l'état et l'entrée du système, et $f(x)$ est une fonction non linéaire.

Plusieurs approches sont possibles suivant le problème de commande en main et les hypothèses prises sur le système (2.1). Dans ce qui suit nous allons analyser le problème posé par chaque cas de figure et la solution à préconiser.

2.2.1 Problème de régulation

Considérons qu'il est souhaité que l'état x se stabilise à la consigne $r = 0$ (ou toute autre valeur constante). C'est un problème de régulation, et le système va fonctionner dans une petite région autour de ce point désiré où la non linéarité $f(x)$ est peu sollicitée et le comportement du système est quasi-linéaire.

Le développement en série de Taylor de $f(x)$ autour du point $x = 0$, produit :

$$f(x) = ax + o(x) \tag{2.2}$$

où $a = \frac{df(x)}{dx}\Big|_{x=0}$ et $o(x)$ représente les termes d'ordres supérieurs du développement. Si x est très proche du point de fonctionnement, les termes $o(x)$ sont négligeables, et le système (2.1) peut être approximé par la linéarisation :

$$\dot{x} = ax + u \tag{2.3}$$

Alors la loi de commande par retour d'état :

$$u = -kx \tag{2.4}$$

avec $k > 0$, produit en boucle fermée le système :

$$\dot{x} = -(k - a)x \tag{2.5}$$

Alors si on choisi $k \geq |a|$ (c.-à-d. qu'une majoration de a est connue), la dynamique en boucle fermée (2.5) sera localement stable.

La simulation de la figure 2.1 montre que pour des conditions initiales proches de zéro (a) la commande linéaire assure la stabilité locale, alors que pour des conditions initiales assez loin de zéro (b) la commande linéaire échoue et le système diverge.

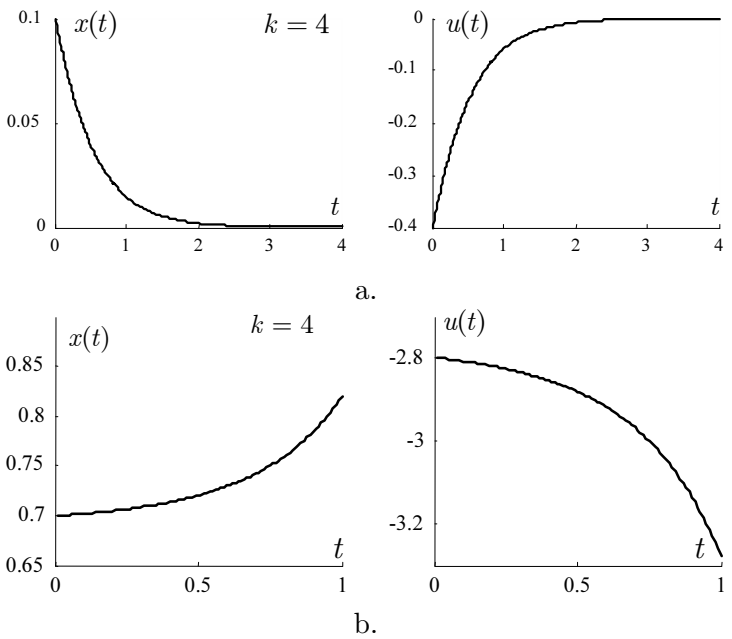


Figure 2.1 : Régulation linéaire du système non linéaire (2.1) avec $f(x) = 2xe^x$ ($a = 2$) et $u = -4x$. (a) $x(0) = 0.1$ (b) $x(0) = 0.7$.

Si la valeur de a est inconnue (mais fixe ou varie très lentement), on peut faire recours à la commande adaptative linéaire pour résoudre le problème [28]. Considérons que la valeur réelle du paramètre a est inconnue et que son estimation est $\widehat{a}$, alors la loi de commande :

$$u = -\widehat{a}x - kx \tag{2.6}$$

permet d'obtenir la dynamique en boucle fermée :

$$\dot{x} = \widetilde{a}x - kx \tag{2.7}$$

où $\widetilde{a} = a - \widehat{a}$ est l'erreur d'estimation du paramètre a.

Pour vérifier la stabilité, considérons la fonction de Lyapunov :

$$V = \frac{1}{2}x^2 + \frac{1}{2\gamma}\widetilde{a}^2 \tag{2.8}$$

où $\gamma > 0$ est un paramètre de conception.

La dérivée de (2.8) le long de la dynamique d'erreur (2.7) produit :

$$\dot{V} = -kx^2 - \frac{1}{\gamma}\widetilde{a}\left(\dot{\widehat{a}} - \gamma x^2\right) \tag{2.9}$$

où nous avons utilisé le fait que : $\dot{\widetilde{a}} = \dot{a} - \dot{\widehat{a}} = -\dot{\widehat{a}}$.

Si on choisi la loi d'ajustement de l'estimation du paramètre comme :

$$\dot{\widehat{a}} = \gamma x^2 \tag{2.10}$$

alors nous aurons : $\dot{V} = -kx^2 \leq 0$, ce qui implique une stabilité asymptotique locale.

La simulation de la figure 2.2 montre que la commande adaptative améliore la stabilité du système non linéaire (comme illustré sur la figure 2.2.a, la commande adaptative stabilise un domaine plus large que la commande linéaire statique). Cependant, pour des conditions initiales assez éloignées, la commande linéaire adaptative échoue, comme le montre la figure 2.2.b.

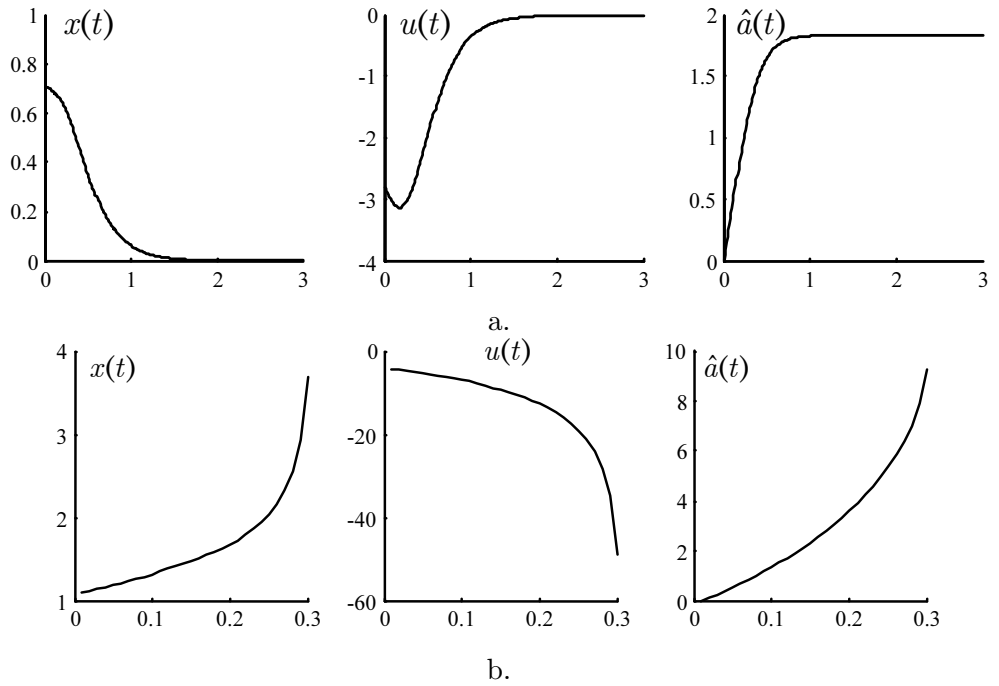


Figure 2.2 : Régulation linéaire adaptative du système non linéaire (2.1) avec $f(x) = 2xe^x$ $(a = 2)$, $\dot{\widehat{a}} = 0.1x^2$ et $u = -\widehat{a}x - 4x$. (a) $x(0) = 0.7$ (b) $x(0) = 1.1$.

2.2.2 Problème de poursuite

Dans ce cas de figure, la consigne $r(t)$ est variable sur une large plage de fonctionnement, et la non linéarité $f(x)$ sera donc très sollicitée, puisque le point de fonctionnement varie très rapidement avec $r(t)$.

Les approches pour résoudre ce problème diffèrent suivant les informations fournies au sujet de la non linéarité $f(x)$.

1. $f(x)$ est inconnue, et l'on désire utiliser une commande linéaire. Si on reprend la linéarisation (2.3), alors l'erreur de poursuite $e(t) = r(t) - x(t)$ est donnée par :

$$\dot{e} = -ax - u + \dot{r} \tag{2.11}$$

La loi de commande :

$$u = -a_0 r + \dot{r}(t) + ke$$

où a_0 est une certaine estimation de a, permet d'obtenir la dynamique de poursuite en boucle fermée :

$$\dot{e} = -(k-a)\,e(t) + (a_0 - a)\,r \tag{2.12}$$

Si $k \geq |a|$ la dynamique de l'erreur est stable, mais ne converge pas vers zéro. En régime permanent, l'erreur est proportionnelle à $r(t)$ pondérée par l'erreur sur la valeur du paramètre $(a_0 - a)$.
La commande linéaire présente, comme il est montré sur la figure 2.3, une erreur de poursuite qui augmente avec l'amplitude de la consigne $r(t)$, et sa performance diminue jusqu'à tendre vers l'instabilité.

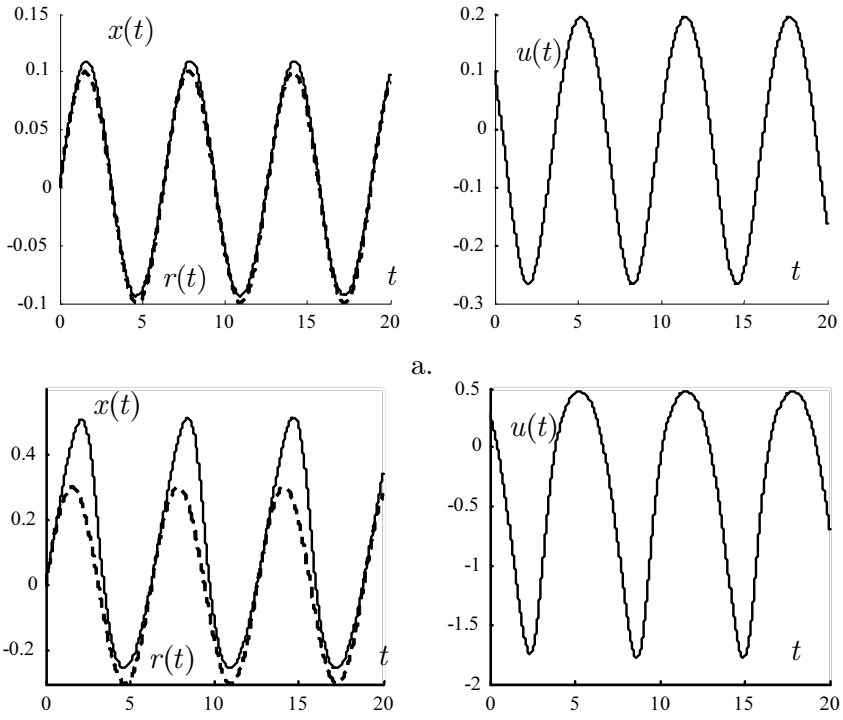


Figure 2.3 : Poursuite linéaire du système non linéaire (2.1) avec $f(x) = 2xe^x$, $r(t) = A\sin t$ et $u = -2r + \dot{r} + 4(r - x)$. (a) $A = 0.1$ (b) $A = 0.3$.

On peut aussi améliorer la performance de la commande linéaire en adoptant la commande adaptative suivante :

$$u = -\widehat{a}x + \dot{r}(t) + ke(t) \tag{2.13}$$

La dynamique d'erreur en boucle fermée est alors :

$$\dot{e} = -ke - \widetilde{a}x \tag{2.14}$$

Pour vérifier la stabilité, considérons la fonction de Lyapunov :

$$V = \frac{1}{2}e^2 + \frac{1}{2\gamma}\widetilde{a}^2 \tag{2.15}$$

La dérivée de (2.15) le long de la dynamique d'erreur (2.14) produit :

$$\dot{V} = -ke^2 - \frac{1}{\gamma}\widetilde{a}\left(\dot{\widehat{a}} + \gamma xe\right) \tag{2.16}$$

Si on choisit la loi d'ajustement de l'estimation du paramètre comme :

$$\dot{\widehat{a}} = -\gamma xe \tag{2.17}$$

alors nous aurons : $\dot{V} = -ke^2 \leq 0$, ce qui implique une stabilité asymptotique locale.

La commande linéaire adaptative améliore sensiblement la précision de la poursuite (voir figure 2.4.a). De la même manière que la commande statique, elle perd de performance avec l'augmentation de l'amplitude de la consigne $r(t)$ (figure 2.4.b).

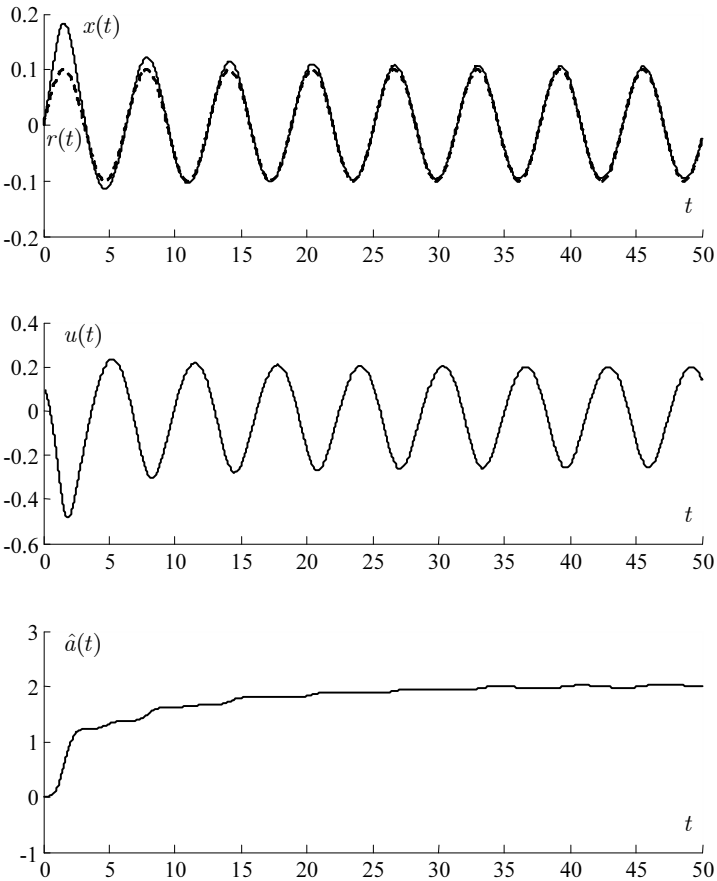


Figure 2.4.a : Poursuite linéaire adaptative du système non linéaire (2.1) avec $f(x) = 2xe^x$, $r(t) = A\sin t$ et $u = -\widehat{a}(t)x + \dot{r} + 4(r-x)$, $\dot{\widehat{a}}(t) = -0.017x(r-x)$, $A = 0.1$.

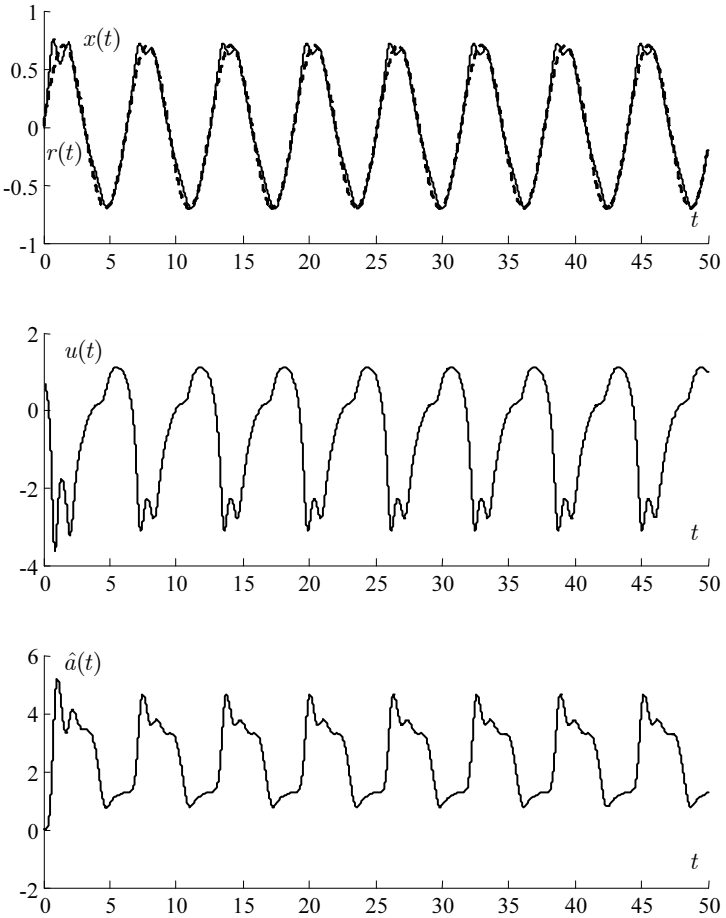


Figure 2.4.b : Poursuite linéaire adaptative du système non linéaire (2.1) avec $f(x) = 2xe^x$, $r(t) = A\sin t$ et $u = -\widehat{a}(t)x + \dot{r} + 4(r-x)$, $\dot{\widehat{a}}(t) = -0.017x(r-x)$, $A = 0.7$.

2. La structure de $f(x)$ est parfaitement connue. Ceci revient à avoir $f(x) = bf_0(x)$ avec b un paramètre inconnu (mais constant) et $f_0(x)$ une certaine fonction non linéaire connue. Ainsi, le problème se réduit à un problème de commande adaptative linéaire vis à vis des paramètres inconnus. Cette approche est à la base de toutes les techniques de commande adaptatives développées en utilisant le modèle non linéaire du système commandé [1-4].

Si la valeur estimée de b est $\widehat{b}$, alors la loi de commande :

$$u = -\widehat{b}f_0(x) + \dot{r}(t) + ke(t) \tag{2.18}$$

permet d'obtenir la dynamique d'erreur en boucle fermée :

$$\dot{e}(t) = -ke(t) - \widetilde{b}f_0(x) \tag{2.19}$$

où $\widetilde{b} = b - \widehat{b}$ est l'erreur d'estimation du paramètre b.

Pour vérifier la stabilité, considérons la fonction de Lyapunov :

$$V = \frac{1}{2}e^2 + \frac{1}{2\gamma}\widetilde{b}^2 \tag{2.20}$$

La dérivée de (2.20) le long de la dynamique d'erreur (2.19) produit :

$$\dot{V} = -ke^2 - \frac{1}{\gamma}\widetilde{b}\left(\dot{\widehat{b}} + \gamma f_0(x)e(t)\right) \tag{2.21}$$

où nous avons utilisé le fait que : $\dot{\widetilde{b}} = -\dot{\widehat{b}}$.

Si on choisi la loi d'ajustement de l'estimation du paramètre comme :

$$\dot{\widehat{b}} = -\gamma f_0(x)e(t) \tag{2.22}$$

alors nous aurons : $\dot{V} = -ke^2 \leq 0$, ce qui implique la stabilité asymptotique globale.

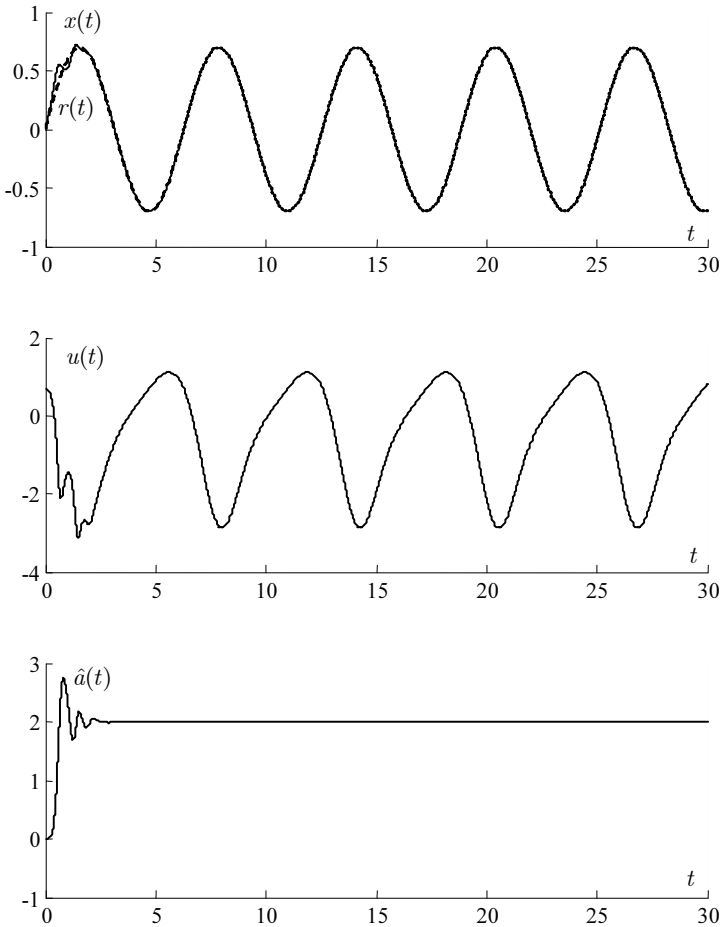


Figure 2.5.a : Poursuite adaptative non linéaire du système (2.1) avec $f(x) = bf_0(x)$, $f_0(x) = xe^x$, $b = 2$, $r(t) = A\sin t$, $A = 0.7$, $u = -\widehat{b}f_0(x) + \dot{r}(t) + 4e(t)$, $\dot{\widehat{b}} = -0.017f_0(x)e(t)$.

La commande non linéaire adaptative présente une performance assez remarquable pour la non linéarité connue, et l'erreur de poursuite est pratiquement nulle en régime établi (voir figure 2.5.a). Cette performance se détériore sensiblement à l'introduction d'un incertitude inconnue (perturbation ou défaut de fonctionnement), et le mécanisme d'adaptation ne peut répondre correctement à cette situation nouvelle (figure 2.5.b).

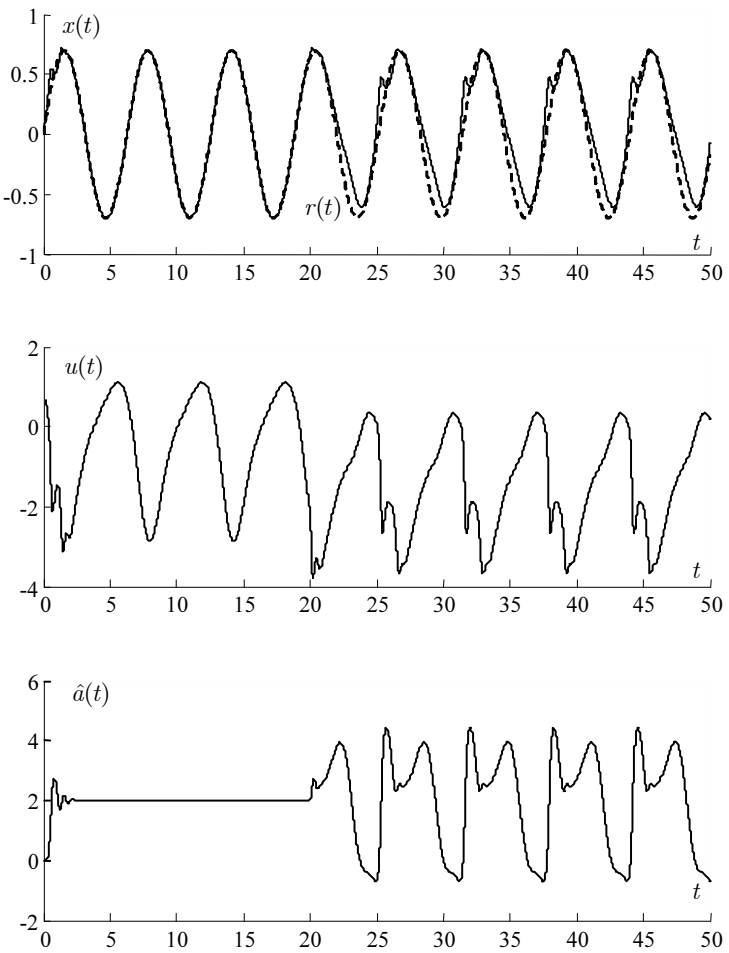


Figure 2.5.b : Poursuite adaptative non linéaire du système (2.1) avec $f_0(x) = xe^x$, $b = 2$, $r(t) = A\sin t$, $A = 0.7$, $u = -\widehat{b}f_0(x) + \dot{r}(t) + 4e(t)$, $\dot{\widehat{b}} = -0.017 f_0(x)e(t)$, $f(x) = bf_0(x)$ pour $t \leq 20\,\text{sec}$ $f(x) = bf_0(x) + \frac{1}{1+x^2}$ pour $t > 20\,\text{sec}$.

Il en sort, de cette analyse, les remarques suivantes :

1. Les techniques de la commande linéaire ne peuvent résoudre le problème de la commande des systèmes non linéaires que dans un voisinage très limité autour du point de fonctionnement. L'évolution des états du système dans un domaine assez éloigné du point de fonctionnement peut engendrer l'instabilité du système comme montré par les simulations des figures 2.1 et 2.3.

2. La commande adaptative linéaire peut améliorer sensiblement les performances du système contrôlé. Cependant, sur une large plage de variation, les performances deviennent non satisfaisantes comme montré sur les figures 2.2 et 2.4.
3. La commande adaptative non linéaire peut se montrer très efficace à condition que le modèle non linéaire traduit avec précision la dynamique du système commandé, comme montré sur la figure 2.5.a. L'apparition, durant le fonctionnement, de défauts ou de perturbations induit des modifications dans la dynamique du système qui ne sont pas prises en compte par le modèle, et peut ainsi dégrader les performances de la commande comme illustré sur la figure 2.5.b.

L'idée fondamentale, qui surgit de ces constatations, est que nous avons besoin d'une nouvelle commande avec les caractéristiques suivantes :

1. Cette commande doit être non linéaire, puisque la linéarisation ne peut prendre en compte le caractère complexe du comportement des systèmes non linéaires.
2. Cette commande doit être adaptative pour faire face aux changements de comportement dûs aux défauts de fonctionnement et aux perturbations.
3. Cette commande doit être, le plus possible, indépendante du modèle du système (free model control), et ceci pour éviter l'imprécision inhérente à toute modélisation.
4. Et enfin, cette commande doit avoir la capacité d'apprendre le comportement du système en observant seulement les données recueillies durant le fonctionnement du processus. Ce qui lui permet de s'adapter aux modifications possibles du comportement du système suite à des changements de consigne, de défauts ou de perturbations.

Dans ce cadre intervient, l'utilisation des réseaux de neurones comme un instrument très efficace pour la modélisation du comportement non linéaire. Cette capacité vient, comme nous le verrons par la suite, de plusieurs caractéristiques intrinsèques des réseaux de neurones.

2.3 Réseaux de neurones

Les réseaux de neurones artificiels utilisent un nombre d'éléments simples de calcul (neurones) interconnectés pour accomplir des tâches compliquées de classification et d'approximation. L'habilité à ajuster les paramètres du réseau de neurones (poids et biais) rend possible l'apprentissage des nouvelles informations au sujet du processus considéré à partir des données. Les réseaux de neurones possèdent la caractéristique désirable qu'un peu d'information au sujet du processus est nécessaire pour le succès de l'application du réseau au problème en main. En d'autres termes, les réseaux de neurones sont considérés comme des techniques "boîte noire". Cette approche conduit à des solutions en un temps relativement cours, puisque les modèles des systèmes, nécessaires pour beaucoup de méthodes conventionnelles, ne sont pas requis.

Vue que la fonction principale que remplissent les réseaux de neurones dans un schéma de commande adaptative est l'émulation ou l'approximation d'un certain comportement non linéaire, nous les appellerons par la suite "approximateurs". Dans ce qui suivera, on va mettre l'accent sur leurs propriétés et leurs avantages.

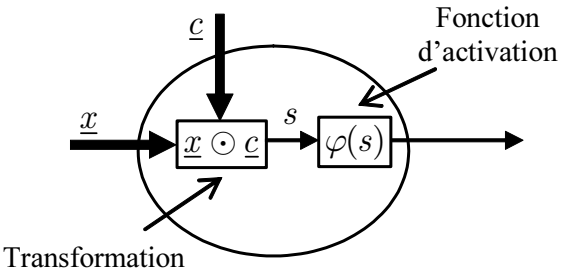


Figure 2.6 : Structure d'un neurone formel.

2.3.1 Modèle mathématique d'un neurone

Le neurone formel (figure 2.6) transforme un vecteur d'entrée $\underline{x} \in \Re^n$ en un scalaire noté s [29]. La transformation de l'entrée dépend d'un certain vecteur de poids $\underline{c}^T = \begin{bmatrix} c_1 & \dots & c_n \end{bmatrix}$ sélectionnés suivant une certaine information donnée au préalable ou ajustés pour satisfaire un certain critère de performance. La transformation qualifie la relation entre $\underline{x}$ et $\underline{c}$. Cette transformation est notée par :

$$s = \underline{c} \odot \underline{x} \tag{2.23}$$

Plusieurs transformations ont été utilisées par le passé, les plus populaires sont le produit scalaire et la distance Euclidienne.

Le produit scalaire donné par :

$$s = \sum_{i=1}^{n} c_i x_i = \underline{c}^T \underline{x} \tag{2.24}$$

est une mesure de la similarité entre les orientations des vecteurs $\underline{x}$ et $\underline{c}$. Ainsi si $\underline{x}$ et $\underline{c}$ sont orthogonaux alors $s = 0$, et plus ils sont colinéaires plus la similarité augmente.

La distance Euclidienne est donnée par :

$$s = |\underline{x} - \underline{c}| = \sqrt{\sum_{i=1}^{n} (x_i - c_i)^2} = \sqrt{(\underline{x} - \underline{c})^T (\underline{x} - \underline{c})} \tag{2.25}$$

Cette transformation produit $s \geq 0$ (puisque c'est une norme).

Ajout d'un biais

En plus des entrées variables, le neurone peut avoir un biais constant comme entrée supplémentaire, tel que le vecteur d'entrée devient $\underline{x}^T = \begin{bmatrix} 1 & x_1 & \dots & x_n \end{bmatrix} \in \Re^{n+1}$ et le vecteur des poids devient $\underline{c}^T = \begin{bmatrix} c_0 & c_1 & \dots & c_n \end{bmatrix} \in \Re^{n+1}$.

Les transformations décrites précédemment restent valables dans ce cas aussi.

Fonctions d'activation

Après la transformation, le neurone produit une sortie en utilisant une fonction d'activation. La fonction d'activation produit une valeur réelle souhaitable pour un autre neurone, ou utilisable par un système externe.

Définition 2.1 : Une fonction $\varphi(s) : \Re \rightarrow \Re$ est dite fonction d'activation si elle est continue par morceau.

Définition 2.2 : Une fonction d'activation $\varphi(s) : \Re \rightarrow \Re$ est dite bornée si $|\varphi(s)| < \infty$ pour $s \rightarrow \infty$.

Quelques unes des fonctions d'activation, utilisées dans les réseaux de neurones, sont définies dans ce qui suit.

Linéaire : C'est la plus simple des fonctions d'activation, elle est définie par :

$$\varphi(s) = s \tag{2.26}$$

Elle est généralement utilisée pour générer la sortie d'un réseau multicouches.

Tangente hyperbolique : C'est une approximation continue de la saturation, elle est définie par :

$$\varphi(s) = \frac{1 - \exp(-as)}{1 + \exp(-as)} \tag{2.27}$$

pour $a > 0$.

Sigmoïde : C'est une fonction d'activation fréquemment utilisée, elle est définie par :

$$\varphi(s) = \frac{1}{1 + \exp(-as)} \tag{2.28}$$

Gaussienne : C'est une fonction d'activation très populaire, elle est définie par :

$$\varphi(s) = \exp\left(-\frac{s^2}{\sigma^2}\right) \tag{2.29}$$

avec $\sigma \in \Re$.

Multi-Quadratique : Elle est définie par :

$$\varphi(s) = \left(s^2 + \sigma^2\right)^{\alpha}, 0 < \alpha \leq 1 \tag{2.30}$$

Multi-Quadratique inverse : Elle est définie par :

$$\varphi(s) = \left(s^2 + \sigma^2\right)^{-\alpha}, \alpha > 0 \tag{2.31}$$

Les fonctions d'activation précédentes sont représentées sur la figure 2.7. Il existe beaucoup d'autres fonctions d'activation moins populaires, mais qui peuvent remplir certaines tâches bien spécifiques.

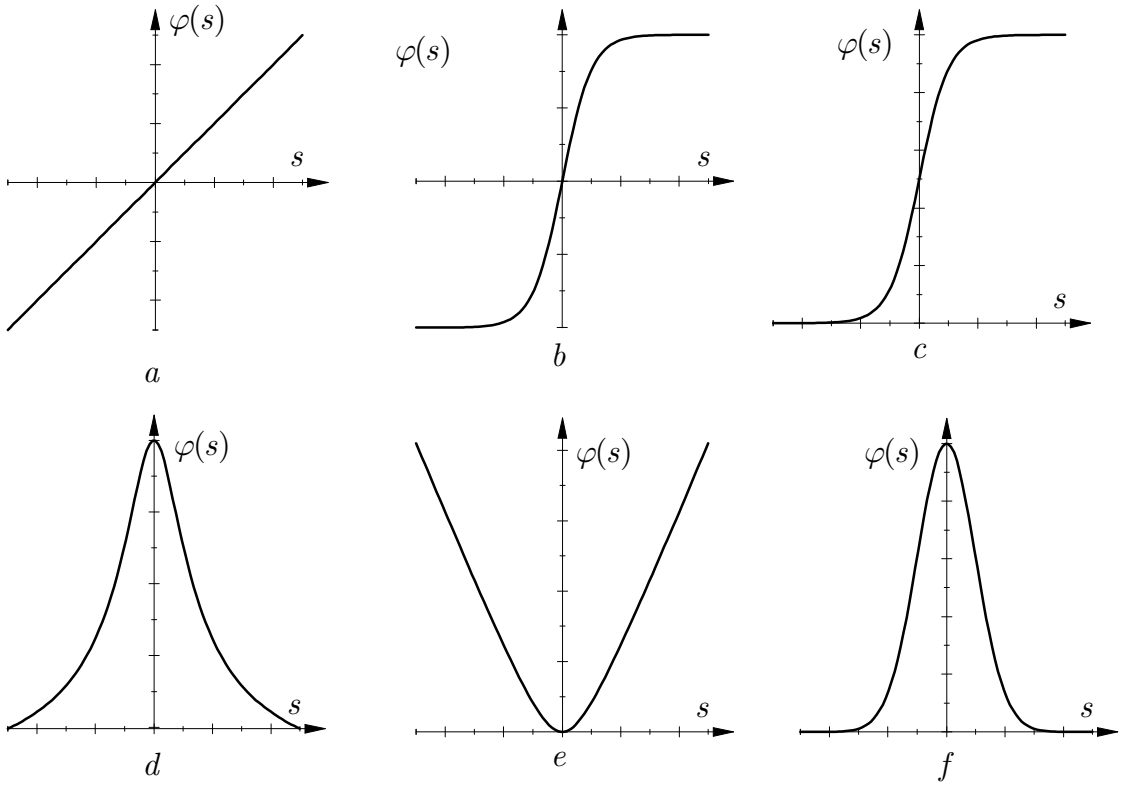


Figure 2.7 : Fonctions d'activation utilisées dans les neurones formels. (a) linéaire (b) tangente hyperbolique (c) sigmoïde (d) gaussienne, (e) quadratique, (f) quadratique inverse.

2.3.2 Réseau perceptron multicouches (MLP)

Le MLP a une long histoire, et il est de loin le réseau le plus appliqué dans le domaine du contrôle. Le MLP est une collection de neurones qui sont arrangés en couches (voir figure 2.8), tel que la sortie de chaque neurone d'une couche est passée aux entrées des neurones de la couche suivante. Une couche d'entrée est le vecteur d'entrée $\underline{x}$, alors que la couche de sortie est la connexion entre le réseau et l'environnement extérieur. Une couche cachée est une collection de neurones qui se situe entre la couche d'entrée et la couche de sortie.

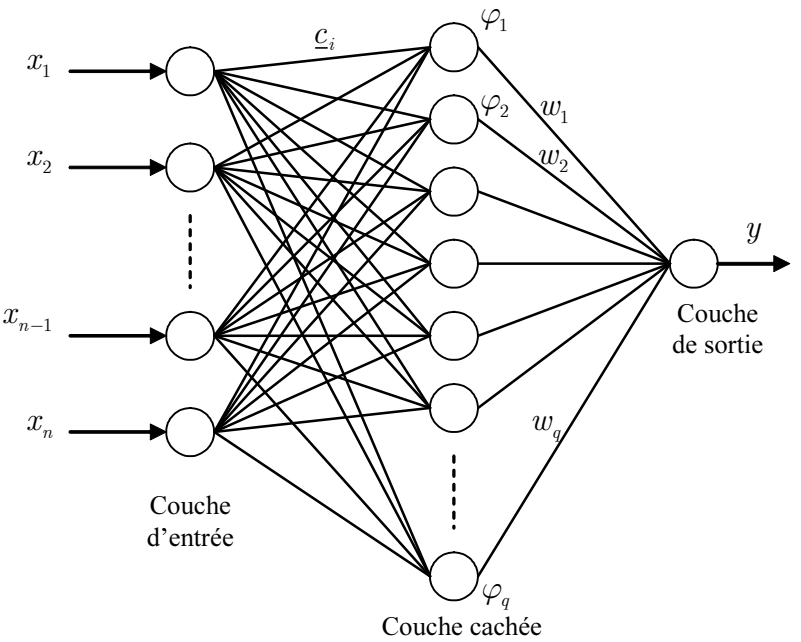


Figure 2.8 : Réseau de neurones avec une seule couche cachée.

Les réseaux MLP utilisent comme transformation le produit scalaire. Les couches d'entrée et de sortie utilisent, comme fonctions d'activation la fonction linéaire ; alors que la couche cachée utilise des fonctions d'activation non linéaires, comme la tangente hyperbolique ou la sigmoïde.

Considérons le réseau MLP avec une seule couche cachée (avec q neurones) de la figure 2.8, alors, la transformation entrée-sortie du réseau est donnée par :

$$\begin{aligned} y &= \sum_{i=1}^{q} w_i \varphi_i \left(\sum_{j=1}^{n} c_{ij} x_j \right) \\ &= \sum_{i=1}^{q} w_i \varphi_i \left(\underline{c}_i^T \underline{x} \right) \end{aligned} \tag{2.32}$$

avec $\underline{c}_i^T = \left[\begin{array}{cccc} c_{i1} & c_{i2} & \ldots & c_{in} \end{array} \right] \in \Re^{1 \times n}$.

Des réseaux MLP multicouches (avec plusieurs couches cachées) peuvent être construits, cependant, dans les applications considérées dans cette thèse une structure avec une seule couche cachée est suffisante, et nous verrons par la suite pourquoi.

Paramétrisations linéaire et non linéaire

Remarquons que, si on arrange les paramètres du réseau MLP sous la forme compacte : $\underline{c}^T = \left[\begin{array}{cccc} \underline{c}_1^T & \underline{c}_2^T & \ldots & \underline{c}_q^T \end{array} \right] \in \Re^{q \times n}$, $\underline{\theta}^T = \left[\begin{array}{cccc} w_1 & w_2 & \ldots & w_q \end{array} \right] \in \Re^q$ et $\underline{\phi}\left(\underline{c}, \underline{x}\right) = \left[\begin{array}{cccc} \varphi_1\left(\underline{c}_1^T \underline{x}\right) & \varphi_2\left(\underline{c}_2^T \underline{x}\right) & \ldots & \varphi_q\left(\underline{c}_q^T \underline{x}\right) \end{array} \right] \in \Re^q$, nous pouvons écrire la sortie du réseau

comme suit :

$$\mathcal{F}(\underline{\theta}, \underline{c}, \underline{x}) = \underline{\phi}\,(\underline{c}, \underline{x})\,\underline{\theta} \tag{2.33}$$

Il est clair que le réseau MLP opère une transformation non linéaire (la sortie est une fonction non linéaire des entrées) paramétrée par les paramètres de la couche cachée $\underline{c}$ qui interviennent d'une façon non linéaire, et les paramètres de la couche de sortie $\underline{\theta}$ qui interviennent d'une façon linéaire. En conclusion, la paramétrisation (2.33) est non linéaire.

Si maintenant les paramètres de la couche cachée $\underline{c}$ sont fixés au préalable (dans l'étape de la construction du réseau et avant son utilisation dans la boucle de commande), alors, le réseau MLP est linéaire par rapport aux paramètres ajustables $\underline{\theta}$; c'est une paramétrisation linéaire, et la transformation (2.33) peut être écrite, en cachant les paramètres fixes, comme :

$$\mathcal{F}(\underline{\theta}, \underline{x}) = \underline{\phi}\,(\underline{x})\,\underline{\theta} \tag{2.34}$$

2.3.3 Réseaux à fonctions de base radiales (RBF)

Les réseaux RBF ont été introduits initialement pour résoudre les problèmes d'interpolation des données multidimensionnelles [30]. Le réseau RBF est un réseau à une seule couche cachée, et prend la même forme représentée sur la figure 2.8. La transformation utilisée dans les réseaux RBF est la distance Euclidienne. Les fonctions d'activations utilisées comme fonction de base sont la gaussienne, la multi-quadratique et la multi-quadratique inverse. La transformation entrée-sortie du réseau RBF est donnée par :

$$y = \sum_{i=1}^{q} w_i \varphi_i\left(|\underline{x} - \underline{c}_i|\,, \sigma_i\right) \tag{2.35}$$

Le premier argument de $\varphi_i\,(.)$ est la distance radiale de l'entrée $\underline{x}$ par rapport au centre $\underline{c}_i$. Lorsque $\varphi_i\,(.)$ est sélectionnée pour être gaussienne ou multi-quadratique inverse, alors la RBF résultante possède des caractéristiques locales spécifiées par le paramètre σ_i. Dans une approche plus générale, différentes valeurs de σ_i peuvent êtres considérées pour chaque élément de la RBF. Dans les réseaux RBF standards, toutes les RBFs sont basées sur la même fonction d'activation.

Paramétrisations linéaire et non linéaire

Toutes les propriétés énoncées, concernant la paramétrisation du MLP, peuvent être transposées sur les réseaux RBF. Si on arrange les paramètres du réseau RBF sous la forme compacte :

$$\begin{aligned} \underline{c}^T &= \left[\begin{array}{cccc} \underline{c}_1^T & \underline{c}_2^T & \ldots & \underline{c}_q^T \end{array}\right] \in \Re^{n\times q}, \underline{\sigma} = \left[\begin{array}{cccc} \sigma_1 & \sigma_2 & \ldots & \sigma_q \end{array}\right] \in \Re^q, \\ \underline{\theta}^T &= \left[\begin{array}{cccc} w_1 & w_2 & \ldots & w_q \end{array}\right] \in \Re^q \end{aligned}$$

et

$$\underline{\phi}\left(\underline{c},\underline{\sigma},\underline{x}\right)=\left[\begin{array}{cccc}\varphi_1\left(\left|\underline{x}-\underline{c}_1\right|,\sigma_1\right) & \varphi_2\left(\left|\underline{x}-\underline{c}_2\right|,\sigma_2\right) & \ldots & \varphi_q\left(\left|\underline{x}-\underline{c}_q\right|,\sigma_q\right)\end{array}\right]\in\Re^q,$$

nous pouvons écrire la sortie du réseau comme suit :

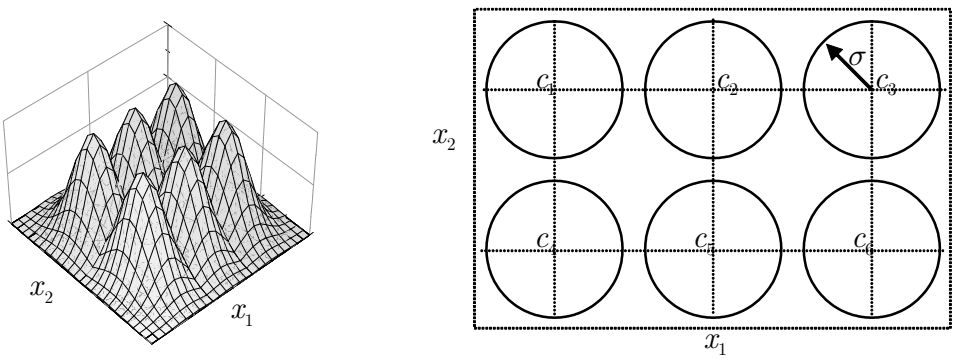
$$\mathcal{F}\left(\underline{\theta},\underline{c},\underline{\sigma},\underline{x}\right)=\underline{\phi}\left(\underline{c},\sigma,\underline{x}\right)\underline{\theta} \tag{2.36}$$

Il est clair que le réseau RBF est une fonction non linéaire de ses entrées et des paramètres des RBFs (couche cachée) $\underline{c}$ et $\underline{\sigma}$. D'où, la parametrisation (2.36) est non linéaire. Si on fixe les paramètres des RBFs au préalable, alors, le réseau MLP est linéaire par rapport aux paramètres ajustables $\underline{\theta}$, soit :

$$\mathcal{F}(\underline{\theta},\underline{x})=\underline{\phi}\left(\underline{x}\right)\underline{\theta} \tag{2.37}$$

Plusieurs méthodes existent pour choisir les paramètres des RBFs [30-33], parmi lesquelles, on peut citer le choix aléatoire des centres en utilisant les données d'entrée $\underline{x}$, et la sélection des centres par un apprentissage non supervisé. Une méthode très simple, et qu'on utilisera par la suite, consiste à distribuer les centres des RBFs d'une manière régulière afin de couvrir l'espace de fonctionnement du système considéré. Cette technique est illustrée sur la figure 2.9 pour un système d'ordre 2.

A noter que dans tous les cas (paramétrisations linéaire et non linèaire), le réseau de neurones est une fonction non linéaire par rapport à ses entrées $\underline{x}$.



a. b.

Figure 2.9 : RBFs avec distribution uniforme sur l'espace d'approximation, (a) représentation tridimensionnelle, (b) projection sur le plan $x_1 - x_2$.

2.3.4 Approximation universelle

Parmi les propriétés des réseaux de neurones, l'approximation universelle est la plus importante du point de vue identification et commande. Cette propriété est la justification de l'utilisation des réseaux de neurones dans les systèmes de commande. L'approximation universelle a fait le sujet de beaucoup de travaux de recherche et plusieurs démonstrations pour des réseaux de neurones avec différentes fonctions d'activation ont

été produites [9-15]. Le théorème suivant énonce la définition de l'approximation universelle pour les réseaux de neurones avec une couche cachée.

Théorème 2.1 [9-15] : Les réseaux de neurones, avec q neurones dans la couche cachée et l'une des fonctions d'activation décrites précédemment, peuvent approximer uniformément toute fonction continue $f(\underline{x}) : \Omega_{\underline{x}} \rightarrow \Re$ avec l'erreur de reconstruction :

$$\epsilon(\underline{x}) = f(\underline{x}) - \mathcal{F}(\underline{\theta}, \underline{c}, \underline{x}) \tag{2.38}$$

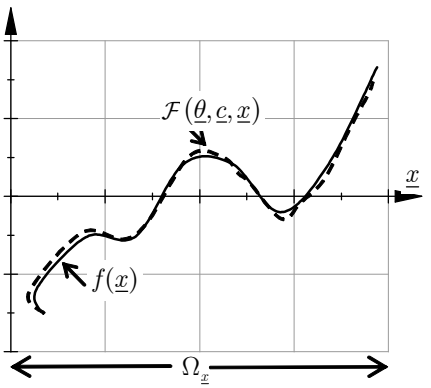


Figure 2.10 : Fonction non linéaire $f(\underline{x})$ et son réseau approximateur $\mathcal{F}(\underline{\theta}, \underline{c}, \underline{x})$ sur le domaine $\Omega_{\underline{x}}$.

L'explication du précèdent théorème nécessite les remarques suivantes :

1. Les réseaux qui vérifient le théorème précédent sont dits "approximateurs universels".
2. Comme il en sort de l'énoncé, le théorème est valide pour les réseaux MLP et RBF.
3. De plus, le théorème est valide aussi pour des réseaux avec plusieurs couches cachées ; mais une couche cachée est la structure minimale pour que la propriété soit vraie, c.-à-d., un réseau avec seulement deux couches (sans couche cachée) n'est pas un approximateur universel.
4. Le théorème est valide, avec une certaine différence qu'on expliquera par la suite, pour les paramétrisations non linéaires (2.33) et (2.36), et les paramétrisations linéaires (2.34) et (2.37).
5. L'erreur de reconstruction $\epsilon(\underline{x})$ peut être réduite en augmentant le nombre de neurones q dans la couche cachée, c.-à-d., que plus le nombre de neurones dans la couche cachée est grand plus la précision est meilleure.
6. Le théorème précédent n'est pas constructif. Ce qui revient à dire, que le théorème garantit l'existence d'un approximateur pour la fonction non linéaire continue $f(\underline{x})$, mais n'indique pas comment choisir la structure ou régler les paramètres du réseau de neurones pour atteindre une certaine précision désirée.

2.3.5 Paramètres idéals et erreur de reconstruction

A partir du théorème 2.1, il est justifié de penser qu'étant donné un approximateur $\mathcal{F}(\underline{\theta}, \underline{x})$ d'une structure déterminée, il existe un jeu de paramètres idéals $\underline{\theta}^*$ qui permet d'obtenir la meilleure approximation possible, c.-à-d., une erreur de reconstruction la plus petite possible. Ce concept est formalisé par le théorème suivant.

Théorème 2.2 [15] : Soit la fonction continue $f(\underline{x}) : \Omega_{\underline{x}} \rightarrow \Re$ et son approximateur $\mathcal{F}(\underline{\theta}, \underline{x})$ avec les entrées $\underline{x} \in \Omega_{\underline{x}}$ et les paramètres $\underline{\theta} \in \Omega_{\underline{\theta}}$, alors il existe un ensemble de paramètres idéal définis par :

$$\underline{\theta}^* = \arg \min_{\underline{\theta} \in \Omega_{\underline{\theta}}} \left(\sup_{\underline{x} \in \Omega_{\underline{x}}} |f(\underline{x}) - \mathcal{F}(\underline{\theta}, \underline{x})| \right) \tag{2.39}$$

et nous avons

$$f(\underline{x}) = \mathcal{F}(\underline{\theta}^*, \underline{x}) + \epsilon^*(\underline{x}) \tag{2.40}$$

où $\epsilon^*(\underline{x})$ est l'erreur de reconstruction minimale réalisée pour les paramètres $\underline{\theta}^*$. De plus, pour $\underline{x} \in \Omega_{\underline{x}}$ nous avons :

$$|\epsilon^*(\underline{x})| \leq \overline{\epsilon} \tag{2.41}$$

où $\overline{\epsilon}$ est une constante de majoration.

Les remarques suivantes s'imposent :

1. Les paramètres idéals $\underline{\theta}^*$ sont définis comme étant l'ensemble des paramètres qui font que l'approximateur, avec une certaine structure $\mathcal{F}(\underline{\theta}^*, \underline{x})$, soit la meilleure approximation (ou la représentation idéale) de la fonction continue $f(\underline{x})$ sur le domaine $\Omega_{\underline{x}}$.
2. Pratiquement, pour un approximateur avec une certaine structure $\mathcal{F}(\underline{\theta}^*, \underline{x})$, il existe plusieurs ensembles de paramètres $\underline{\theta}^* \in \Omega_{\underline{\theta}}$ qui peuvent réaliser la représentation idéale de $f(\underline{x})$. L'opérateur "arg" veut dire simplement qu'on prend l'un de ces jeux de paramètres.
3. La relation (2.41) indique que l'erreur de reconstruction idéale $\epsilon^*(\underline{x})$ est bornée sur le domaine de reconstruction $\Omega_{\underline{x}}$. La constante de majoration $\overline{\epsilon}$ dépend de la structure et de l'ensemble des paramètres $\underline{\theta}^*$ choisis. Elle est difficile à évaluer avec exactitude, mais, dans la pratique, on peut lui fixer une valeur surestimée où l'identifier adaptativement dans la boucle de commande.
4. Le terme structure réfère au nombre de neurones q dans la couche cachée. Le choix de ce paramètre est crucial pour la précision de l'approximation. Il existe dans la littérature deux procédures pour sélectionner ce paramètre. La première procédure consiste à fixer a priori le nombre de neurones dans la couche cachée en procédant à plusieurs tests pour obtenir la structure appropriée. La deuxième méthode, qui est toujours un domaine de recherches actives, consiste à ajouter des neurones, dès que c'est nécessaires, pour améliorer les performances tout en garantissant la stabilité de la boucle adaptative.

5. Le théorème 2.2 est valide pour les paramétrisations non linéaires (2.33) et (2.36) et les paramétrisations linéaires (2.34) et (2.37).
6. Les paramètres idéals $\underline{\theta}^*$ sont un artifice mathématique nécessaire pour l'application des réseaux de neurones aux problèmes d'approximation non linéaire. Dans la pratique, il n'est pas nécessaire de connaitre les paramètres idéals $\underline{\theta}^*$.

2.3.6 Paramétrisation linéaire contre paramétrisation non linéaire

Pour une application donnée, le concepteur commence toujours par fixer la structure du réseau de neurones, à savoir le nombre de neurones q dans la couche cachée. A ce stade, le concepteur doit choisir entre une paramétrisation non linéaire $\underline{\phi}(\underline{c}, \underline{x})\,\underline{\theta}$ où tous les paramètres du réseau sont ajustables dans la boucle de commande, ou une paramétrisation linéaire $\underline{\phi}(\underline{x})\,\underline{\theta}$ où les paramètres de la couche cachée sont fixés a priori, et seulement les paramètres de la couche de sortie sont ajustés dans la boucle de commande. Dans ce qui suit nous allons discuter les avantages et les inconvénients de chacun de ces deux choix des points de vue : complexité, capacité d'approximation et convergence de l'algorithme d'apprentissage.

Capacité d'approximation

Les résultats des recherches sur l'approximation [13, 15] montrent que pour un réseau à paramétrisation non linéaire, si les paramètres sont bien ajustés, alors l'erreur d'approximation sur le domaine $\Omega_{\underline{x}}$ est majorée par :

$$|\epsilon(\underline{x})| < \frac{\delta_{NL}}{q} \tag{2.42}$$

où q est le nombre de neurones dans la couche cachée et δ_{NL} est un paramètre dont la valeur dépend de la taille du domaine $\Omega_{\underline{x}}$ (δ_{NL} augmente avec la taille de $\Omega_{\underline{x}}$) et des oscillations de la fonction $f(\underline{x})$ à approximer (δ_{NL} augmente avec les oscillations de $f(\underline{x})$). Pour un domaine fixe et pour un vecteur d'entrée $\underline{x}$ de dimension n fixe, les résultats montrent que l'augmentation de q réduit l'erreur d'approximation d'une manière significative, alors que la complexité (nombre de paramètres) augmente linéairement seulement.

Pour les réseaux à paramétrisation linéaire, les résultats [13, 15] montrent que, pour la même fonction $f(\underline{x})$, et si les paramètres sont bien ajustés, alors l'erreur d'approximation sur le domaine $\Omega_{\underline{x}}$ est majorée par :

$$|\epsilon(\underline{x})| < \frac{\delta_L}{q^{\frac{2}{n}}} \tag{2.43}$$

où le paramètre δ_L exhibe les mêmes dépendances que pour le cas non linéaire. A noter, que l'erreur dépend de la dimension n du vecteur d'entrée $\underline{x}$ ($|\epsilon(\underline{x})|$ augmente avec la dimension n).

Complexité

L'analyse de la complexité montre qu'un réseau à paramétrisation non linéaire produit la même précision qu'un réseau à paramétrisation linéaire, mais avec une complexité beaucoup plus faible. De plus, la complexité d'un réseau à paramétrisation linéaire peut augmenter dramatiquement avec l'augmentation de n comme le montre la relation (2.43). Une complexité additionnelle pour les réseaux à paramétrisation non linéaire vient du nombre de paramètres à ajuster et la complexité de l'algorithme d'ajustement.

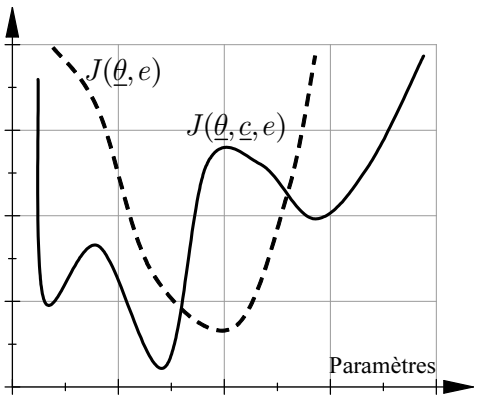


Figure 2.11 : Fonction coût convexe (−−) et fonction coût non convexe (—).

Algorithmes d'ajustement et convergence

Les algorithmes d'ajustement (ou d'apprentissage) adaptent les paramètres d'un réseaux de neurones dans le sens qui minimise une certaine fonction coût $J(.)$ (critère de performance). Pour les réseaux à paramétrisation linéaire cette fonction coût $J(\underline{\theta}, e)$ est convexe par rapport aux paramètres du réseau (voir figure 2.11). Ceci revient à dire que $J(\underline{\theta}, e)$ possède un seul minimum global, et les algorithmes d'ajustement (de type gradient) pointent toujours vers ce minimum quelque soit l'initialisation des paramètres, et les paramètres du réseau convergent vers la valeur optimale qui minimise la fonction coût [33].

Sur l'autre côté, pour les réseaux à paramétrisation non linéaire la fonction coût $J(\underline{\theta}, \underline{c}, e)$ n'est pas convexe par rapport aux paramètres du réseau (voir figure 2.11), elle présente plusieurs minimums locaux avec un bassin d'attraction différent pour chacun d'eux. Ainsi, suivant l'initialisation des paramètres du réseau, la fonction coût peut être attrapée dans l'un des minimums locaux et le minimum global ne sera jamais atteint.

2.3.7 Algorithmes d'ajustement et stabilité

Dans le contexte des applications statiques (classification, identification hors ligne, etc.), les réseaux de neurones sont entraînés par la rétro-propagation qui est un algorithme de type gradient [16]. Le réseau de neurones est entraîné, en utilisant l'information a priori, plusieurs fois jusqu'à ce qu'un objectif ou critère de performance $J(\underline{\theta}, e)$ soit

atteint. L'erreur d'entraînement est utilisée pour améliorer l'ajustement dans l'étape suivante. La question de stabilité ne se pose même pas dans ce contexte.

En général, l'algorithme d'ajustement prend la forme :

$$\dot{\underline{\theta}} = -\gamma \nabla J(\underline{\theta}, e) = -\gamma \psi(\underline{x}) e \tag{2.44}$$

où $\nabla J(\underline{\theta}, e) = \frac{\partial J(\underline{\theta}, e)}{\partial \underline{\theta}}$ et $\gamma > 0$.

L'utilisation des réseaux de neurones dans des contextes dynamiques impliquant une boucle de commande soulève une foule de problèmes qui doivent être résolus avant de pouvoir être utilisés avec confiance. Dans les boucles de commande, les réseaux de neurones sont utilisés principalement en tant que contrôleurs de systèmes dynamiques pour faire face aux non linéarités et incertitudes inconnues, ce qui rend l'ensemble du système sous étude à la fois non linéaire et adaptatif. Dans ce contexte la rétro-propagation (2.44), à elle seule, ne peut garantir la convergence des paramètres ajustables du réseau de neurones. Ainsi, on peut assister à une dérive des paramètres (parameters drift), phénomène très connu dans la littérature de la commande adaptative [28]. Ce phénomène se produit suite à la présence d'erreur de modélisation, de perturbation externe ou bruit de mesure. La divergence des paramètres du réseau de neurones peut provoquer des résultats catastrophiques tels que la perte de stabilité et de contrôle.

Pour remédier à ce problème, plusieurs modifications ont été proposées à la loi d'ajustement de base (2.44), pour assurer la bornitude des paramètres ajustables. Ces modifications incluent la σ-modification, la e-modification, la projection et la zone morte [28].

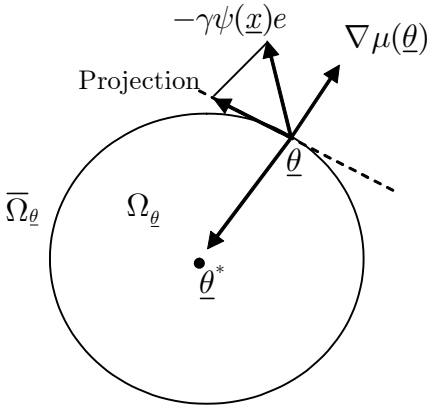


Figure 2.12 : Schématisation de l'algorithme de projection.

Un des moyens les plus simples et efficaces pour éviter la dérive des paramètres est de restreindre les estimations des paramètres dans une région bornée et convexe $\Omega_{\underline{\theta}}$, qui est choisie de sorte que $\underline{\theta}^* \in \Omega_{\underline{\theta}}$ (figure 2.12). En outre, les conditions initiales $\underline{\theta}(0)$ sont choisies de sorte que $\underline{\theta}(0) \in \Omega_{\underline{\theta}}$. La modification de projection met en œuvre cette idée comme suit : si l'estimation du paramètre $\underline{\theta}(t)$ est à l'intérieur de la région souhaitée $\Omega_{\underline{\theta}}$ ou sur le contour de cette région $\overline{\Omega}_{\underline{\theta}}$, avec sa dérivée pointant vers l'intérieur de la région, la loi d'adaptation (2.44) est mise en œuvre. Maintenant, si l'estimation du paramètre

$\underline{\theta}(t)$ est sur le contour de cette région $\overline{\Omega}_{\underline{\theta}}$ avec sa dérivée pointant vers l'extérieur de la région, alors sa dérivée est projetée sur l'hyperplan tangent à $\Omega_{\underline{\theta}}$. Par conséquent, la projection conserve le vecteur d'estimation de paramètres dans la région convexe $\Omega_{\underline{\theta}}$ tout le temps.

Enfin, l'algorithme de projection est défini comme suit. Soit la région convexe fermée $\Omega_{\underline{\theta}}$ définie par :

$$\Omega_{\underline{\theta}} = \{\underline{\theta} | \mu(\underline{\theta}) \leq 0\} \tag{2.45}$$

avec $\mu(\underline{\theta})$ est une fonction lisse. Alors l'algorithme de projection est donné par :

$$\dot{\underline{\theta}} = \begin{cases} -\gamma\psi(\underline{x})e \text{ si } |\underline{\theta}| \in \Omega_{\underline{\theta}} \text{ ou } \left(|\underline{\theta}| \in \overline{\Omega}_{\underline{\theta}} \text{ et } \nabla\mu^T(\underline{\theta})\gamma\psi(\underline{x})e \geq 0\right) \\ -\gamma\psi(\underline{x})e + \frac{\nabla\mu(\underline{\theta})\nabla\mu^T(\underline{\theta})}{\nabla\mu^T(\underline{\theta})\nabla\mu(\underline{\theta})}\gamma\psi(\underline{x})e \text{ si } |\underline{\theta}| \in \overline{\Omega}_{\underline{\theta}} \text{ et } \nabla\mu^T(\underline{\theta})\gamma\psi(\underline{x})e < 0 \end{cases} \tag{2.46}$$

où $\nabla\mu(\underline{\theta}) = \frac{\partial\mu(\underline{\theta})}{\partial\underline{\theta}}$.

Si on choisit de contraindre les paramètres estimés tel que $|\underline{\theta}| \leq \theta_{\max}$, alors nous avons $\mu(\underline{\theta}) = |\underline{\theta}|^2 - \theta_{\max}^2$ et $\nabla\mu(\underline{\theta}) = 2\underline{\theta}$. L'algorithme de projection peut être écrit comme :

$$\dot{\underline{\theta}} = \begin{cases} -\gamma\psi(\underline{x})e & \text{si } |\underline{\theta}| < \theta_{\max} \text{ ou } \left(|\underline{\theta}| = \theta_{\max} \text{ et } \gamma\underline{\theta}^T\psi(\underline{x})e \geq 0\right) \\ -\gamma\psi(\underline{x})e + \frac{\underline{\theta}\underline{\theta}^T}{\underline{\theta}^T\underline{\theta}}\gamma\psi(\underline{x})e & \text{si } |\underline{\theta}| = \theta_{\max} \text{ et } \gamma\underline{\theta}^T\psi(\underline{x})e < 0 \end{cases} \tag{2.47}$$

Cet algorithme garantit que $|\underline{\theta}| \leq \theta_{\max}, \forall t$ si $|\underline{\theta}(0)| \leq \theta_{\max}$ [28].

2.4 Systèmes non linéaires

En dépit des progrès de la théorie de la stabilité pendant plus d'un demi siècle, notre connaissance de la stabilité des systèmes non linéaires est, en général, très limitée. Cela rend la stabilité des systèmes non linéaires adaptatifs contenant des réseaux de neurones un problème vraiment formidable. Ainsi, tout en discutant de l'utilisation des réseaux de neurones pour le contrôle des systèmes non linéaires, il nous incombe de préciser la catégorie des systèmes considérés, les informations préalables à notre disposition concernant le système, les perturbations externes qui peuvent être présentes, la région d'intérêt dans l'espace d'état, et les conditions sous lesquelles les résultats obtenus sont valides.

2.4.1 Systèmes non linéaires SISO

Nous considérerons, en premier lieu, la classe des systèmes non linéaires SISO définie par l'équation différentielle non linéaire :

$$\begin{aligned} y^{(n)} &= f\left(y, \dot{y}, ..., y^{(n-2)}, y^{(n-1)}\right) + g\left(y, \dot{y}, ..., y^{(n-2)}, y^{(n-1)}\right) u \\ &\quad + d(y, \dot{y}, ..., y^{(n-2)}, y^{(n-1)}, t) \end{aligned} \tag{2.48}$$

où $f(.)$, $g(.)$ sont des fonctions non linéaires et $d(.)$ est le terme qui regroupe les incertitudes et les perturbations externes.

Si nous introduisons le vecteur des variables d'état suivant :

$$x^T = \left[\begin{array}{ccccc} x_1 & x_2 & ... & x_{n-1} & x_n \end{array} \right] = \left[\begin{array}{ccccc} y & \dot{y} & ... & y^{(n-2)} & y^{(n-1)} \end{array} \right]$$

On peut écrire l'équation (2.48) sous forme d'état comme :

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_3 \\ &\vdots \\ \dot{x}_{n-1} &= x_n \\ \dot{x}_n &= f(\underline{x}) + g(\underline{x})\, u + d(\underline{x}, t) \qquad (2.49) \\ y &= x_1 \qquad (2.50) \end{aligned}$$

ou encore, sous une forme plus compacte comme :

$$\dot{\underline{x}} = A\underline{x} + \underline{b}\left[f(\underline{x}) + g(\underline{x})\, u + d(\underline{x}, t)\right] \qquad (2.51)$$

$$y = \underline{cx} \qquad (2.52)$$

avec

$$A = \left[\begin{array}{ccccc} 0 & 1 & 0 & ... & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ & & & 1 & 0 \\ 0 & \ldots & & 0 & 1 \\ 0 & 0 & \ldots & & 0 \end{array} \right] = \left[\begin{array}{cc} \underline{0}_{(n-1)\times 1} & I_{n-1} \\ \multicolumn{2}{c}{\underline{0}_{1\times n}} \end{array} \right] \in \Re^{n\times n}$$

$$\underline{b} = \left[\begin{array}{c} 0 \\ \vdots \\ 0 \\ 1 \end{array} \right] \in \Re^{n\times 1}, \underline{c} = \left[\begin{array}{ccccc} 1 & 0 & ... & 0 \end{array} \right] \in \Re^{1\times n}$$

et $I_n \in \Re^{n\times n}$ est la matrice identité.

La forme (2.51)-(2.52) est appelée forme canonique contrôlable (ou forme d'Isodori [3]), et peut être schématisée sous le diagramme de la figure 2.13.

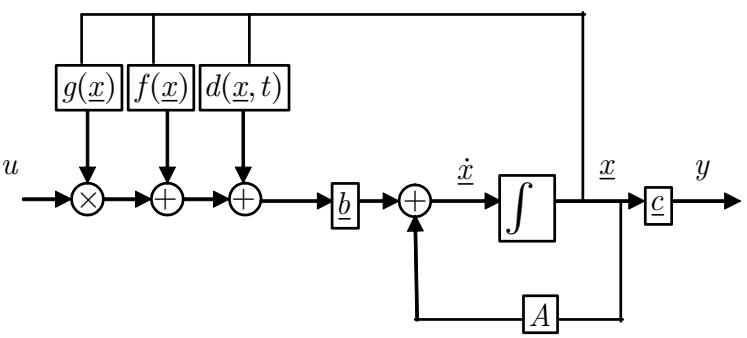


Figure 2.13 : Schématisation du système non linéaire (2.51)-(2.52).

Un grand nombre de systèmes non linéaires peuvent être représentés (ou transformés) sous la forme canonique (2.51)-(2.52).

Pour la suite de ce travail, nous considérerons que le système non linéaire (2.51)-(2.52) vérifie les hypothèses suivantes :

1. Les fonctions non linéaires $f(\underline{x})$ et $g(\underline{x})$ sont inconnues mais lisses (différentiables autant que nécessaire).
2. Le gain d'entrée $g(\underline{x})$ est strictement positif dans la région d'intérêt $\Omega_{\underline{x}}$. Le cas strictement négatif peut aussi être considéré.
3. Le terme $d(\underline{x}, t)$ est borné.

2.4.2 Systèmes non linéaires MIMO

Nous considérerons, aussi, la classe des systèmes non linéaires MIMO définie par l'ensemble d'équations différentielles non linéaires :

$$\begin{aligned} y_i^{(n_i)} &= f_i\left(y_1, \dot{y}_1, \ldots, y_1^{(n_1-1)}, y_2, \dot{y}_2, \ldots, y_2^{(n_2-1)}, \ldots, y_p, \dot{y}_p, \ldots, y_p^{(n_p-1)}\right) \\ &\quad + \sum_{j=1}^{p} g_{ij}\left(y_1, \dot{y}_1, \ldots, y_1^{(n_1-1)}, y_2, \dot{y}_2, \ldots, y_2^{(n_2-1)}, \ldots, y_p, \dot{y}_p, \ldots, y_p^{(n_p-1)}\right) u_j \\ &\quad + d_i\left(y_1, \dot{y}_1, \ldots, y_1^{(n_1-1)}, y_2, \dot{y}_2, \ldots, y_2^{(n_2-1)}, \ldots, y_p, \dot{y}_p, \ldots, y_p^{(n_p-1)}, t\right) \qquad (2.53) \\ i &= 1, ..., p \end{aligned}$$

où $f_i(.)$, $g_{ij}(.)$ $i, j = 1, ..., p$ sont des fonctions non linéaires, $d_i(.)$ $i = 1, ..., p$ sont des termes regroupant les incertitudes et les perturbations externes, et $n_1 + n_2 + ... + n_p = n$ est l'ordre du système.

Si nous introduisons le vecteur des variables d'état suivant :

$$x^T = \left[\begin{array}{cccccccccccc} y_1 & \dot{y}_1 & ... & y_1^{(n_1-1)} & y_2 & \dot{y}_2 & ... & y_2^{(n_2-1)} & ... & y_p & \dot{y}_p & ... & y_p^{(n_p-1)} \end{array}\right]$$

On peut écrire l'équation (2.53) sous forme d'état comme :

$$\dot{\underline{x}} = A\underline{x} + B\left[\underline{f}(\underline{x}) + G(\underline{x})\,\underline{u} + \underline{d}(\underline{x}, t)\right] \qquad (2.54)$$

$$\underline{y} = C\underline{x} \qquad (2.55)$$

avec

$$\underline{f}(\underline{x}) = \begin{bmatrix} f_1(\underline{x}) \\ f_2(\underline{x}) \\ \vdots \\ f_p(\underline{x}) \end{bmatrix}, G(\underline{x}) = \begin{bmatrix} g_{11}(\underline{x}) & g_{12}(\underline{x}) & \cdots & g_{1p}(\underline{x}) \\ g_{21}(\underline{x}) & g_{22}(\underline{x}) & \cdots & g_{2p}(\underline{x}) \\ \vdots & \vdots & \ddots & \vdots \\ g_{p1}(\underline{x}) & g_{p2}(\underline{x}) & \cdots & g_{pp}(\underline{x}) \end{bmatrix}, \underline{d}(\underline{x}, t) = \begin{bmatrix} d_1(\underline{x}, t) \\ d_2(\underline{x}, t) \\ \vdots \\ d_p(\underline{x}, t) \end{bmatrix}$$

et

$$A = \begin{bmatrix} A_1 & 0 & \dots & 0 \\ 0 & A_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & A_p \end{bmatrix} \in \Re^{n\times n}, B = \begin{bmatrix} \underline{b}_1 & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{b}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underline{0} \\ \underline{0} & \dots & \underline{0} & \underline{b}_p \end{bmatrix} \in \Re^{n\times p}$$

$$C = \begin{bmatrix} \underline{c}_1 & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{c}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underline{0} \\ \underline{0} & \dots & \underline{0} & \underline{c}_p \end{bmatrix} \in \Re^{p\times n}$$

où

$$A_i = \begin{bmatrix} \underline{0}_{(n_i-1)\times 1} & I_{n_i-1} \\ \underline{0}_{1\times n_i} & \end{bmatrix} \in \Re^{n_i\times n_i}, \underline{b}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \in \Re^{n_i\times 1},$$

$$\underline{c}_i = \begin{bmatrix} 1 & 0 & ... & 0 \end{bmatrix} \in \Re^{1\times n_i}, i = 1, ..., p$$

La forme (2.54)-(2.55) peut être schématisée sous la forme du diagramme de la figure 2.14.

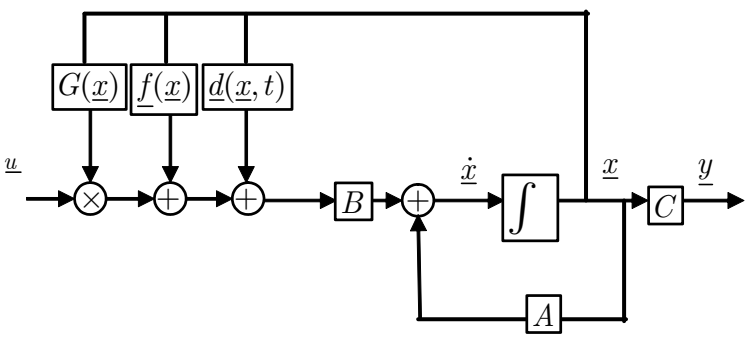


Figure 2.14 : Schématisation du système non linéaire (2.54)-(2.55).

Un grand nombre de systèmes non linéaires peuvent être représentés (ou transformés) sous la forme canonique (2.54)-(2.55).

Pour la suite de ce travail, nous considérerons que le système non linéaire (2.54)-(2.55) vérifie les hypothèses suivantes :

1. Les fonctions non linéaires $f_i(.)$, $g_{ij}(.)$ $i, j = 1, ..., p$ sont inconnues mais lisses.
2. La matrice de gain d'entrée $G(\underline{x})$ est inversible dans la région d'intérêt $\Omega_{\underline{x}}$. De plus, les éléments diagonaux $g_{ii}(.)$ $i = 1, ..., p$ sont strictement positifs.
3. Les termes $d_i(\underline{x}, t)$ $i = 1, ..., p$ sont bornés.

2.4.3 Contrôlabilité, observabilité, et stabilité

Les concepts de contrôlabilité, d'observabilité et de stabilité sont des propriétés théoriques des systèmes qui jouent des rôles importants dans les problèmes liés à la commande des systèmes.

Un système serait contrôlable si l'état initial peut être ramené vers n'importe quel état final par l'application d'une entrée de commande appropriée. Le système non linéaire SISO décrit par l'équation (2.51) est contrôlable si la matrice

$$\left[\begin{array}{ccccc} \underline{b} & A\underline{b} & A^2\underline{b} & ... & A^{n-1}\underline{b} \end{array} \right] \in \Re^{n \times n}$$

est non singulière.

Le système non linéaire MIMO décrit par l'équation (2.54) est contrôlable si la matrice

$$\left[\begin{array}{ccccc} B & AB & A^2B & ... & A^{n-1}B \end{array} \right] \in \Re^{n \times np}$$

est de rang n.

Le concept duel de la contrôlabilité est l'observabilité. Un système est dit observable si l'état initial (et par conséquent tout état suivant) du système peut être déterminé en observant la sortie du système $y(t)$ pendant un intervalle fini de temps. Pour le système non linéaire SISO décrit par l'équation (2.51)-(2.52), la condition pour l'observabilité est que la matrice

$$\left[\begin{array}{ccccc} \underline{c}^T & A^T\underline{c}^T & \left(A^2\right)^T\underline{c}^T & ... & \left(A^{n-1}\right)^T\underline{c}^T \end{array} \right] \in \Re^{n \times n}$$

soit non singulière.

Pour des systèmes non linéaires MIMO décrits par l'équation (2.54)-(2.55), la condition pour l'observabilité est que la matrice

$$\left[\begin{array}{ccccc} C^T & A^TC^T & \left(A^2\right)^TC^T & ... & \left(A^{n-1}\right)^TC^T \end{array} \right] \in \Re^{n \times np}$$

soit de rang n.

Il faut remarquer que les systèmes non linéaires (2.51)-(2.52) et (2.54)-(2.55) sous les hypothèses 2, vérifient les conditions de contrôlabilité et d'observabilité.

Contrôlabilité et stabilité

Pour des systèmes non linéaires décrits par l'équation (2.51) ou l'équation (2.54), si la paire $(A, \underline{b})$ (resp. (A, B)) est contrôlable, alors il peut être stabilisé par un retour d'état $u(\underline{x})$ (resp. $\underline{u}(\underline{x})$).

Estimation et commande

Pour des systèmes non linéaires décrits par les équations (2.51)-(2.52) ou les équations (2.54)-(2.55), si le triplet $(\underline{c}, A, \underline{b})$ (resp. (C, A, B)) est contrôlable et observable, alors il existe une entrée de commande $u(\widehat{\underline{x}})$ (resp. $\underline{u}(\widehat{\underline{x}})$) qui peut stabiliser le système, où $\widehat{\underline{x}}$ est l'estimation de l'état $\underline{x}$ du système.

2.5 Conclusion

Dans ce chapitre, nous avons essayé de cerner, à travers un exemple simple mais illustratif, les problèmes posés par la commande des systèmes non linéaires. Ensuite, nous avons introduit les éléments de base, qui serviront par la suite comme instruments pour la construction des approches de la commande adaptative neuronale proposées. Nous avons passé en revue les caractéristiques essentielles qui justifient l'utilisation des réseaux de neurones dans l'estimation et la commande des systèmes non linéaires, à savoir, l'approximation universelle, la paramétrisation idéale et les algorithmes d'apprentissage. L'exposé est peut être un peu abrégé, mais ceci est dû au fait que l'information sur les réseaux de neurones est assez abondante.

Chapitre 3

Commande MRAC neuronale par retour d'état

3.1 Introduction

Le présent chapitre développe une technique de commande à modèle de référence (MRAC) neuronale pour une classe de systèmes non linéaires incertains. Cette commande est une extension de la commande MRAC des systèmes linéaires. La section 3.2 rappelle le principe de la commande MRAC des systèmes linéaires. La section 3.3 pose le problème de la commande MRAC des systèmes non linéaires, développe la loi de commande neuronale et introduit les algorithmes d'ajustement des paramètres des réseaux neurones utilisés dans la structure de commande. La section 3.4 développe l'analyse de stabilité pour la dynamique de poursuite en boucle fermée. La section 3.5 explicite quelques aspects et propriétés de la technique proposée. La section 3.6 présente deux tests de simulation pour confirmer les résultats théoriques obtenus. Enfin, la section 3.7 conclut le chapitre.

3.2 Commande MRAC des systèmes linéaires

3.2.1 Développements historiques

La commande adaptative à modèle de référence (voir figure 3.1) a été proposée pour la première fois pour les systèmes linéaires en 1958 à M.I.T. par Whitaker [35]. L'approche MRAC a été basée sur une règle heuristique de gradient, également connue sous le nom de règle de M.I.T. Grayson [36] et Butchart [37] ont développé cette première approche heuristique en une méthodologie de conception plus rigoureuse en présentant une analyse de stabilité de la boucle fermée basée sur la seconde méthode de Lyapunov (ou méthode directe). Parks prolongea le résultat de Butchart et présenta pour la première fois le principe pour la conception de la commande par retour de sortie en introduisant la condition réelle positive [38]. Dressler [39] a introduit la formulation de la commande MRAC dans l'espace d'état. Monopoli a prolongé ces résultats aux systèmes de degré

relatif supérieur à un [40], et par la suite aux systèmes multivariables [41].

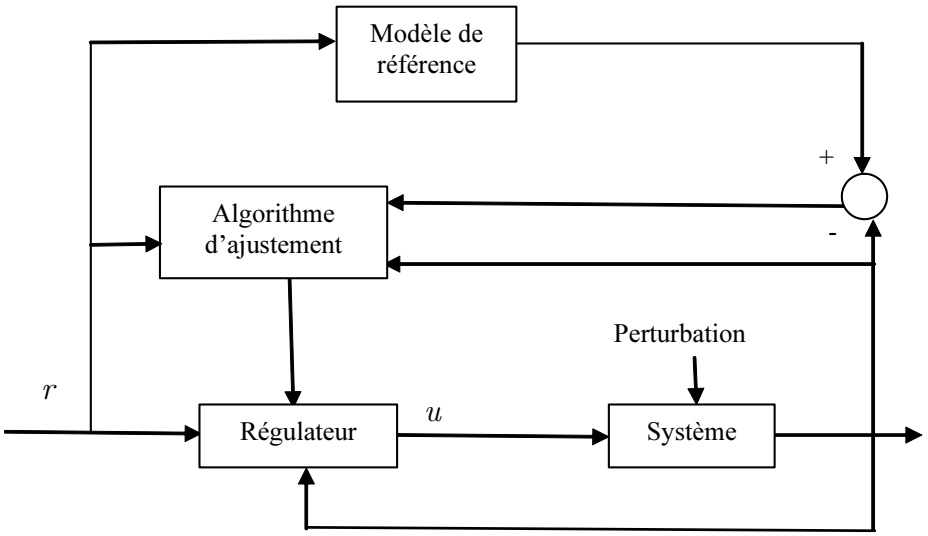


Figure 3.1 : Structure générale de la commande MRAC.

3.2.2 Conception MRAC pour les systèmes linéaires

Pour plus de clarté, nous commencerons, par énoncer le problème de la commande MRAC des systèmes linéaires (voir figure 3.2) définis par :

$$\dot{\underline{x}} = A\underline{x} + \underline{b}\,[\underline{a}\underline{x} + bu] \tag{3.1}$$

où $\underline{x}^T = \left[\begin{array}{cccc} x_1 & x_2 & .. & x_n \end{array}\right] \in \Re^{1\times n}$ est le vecteur d'état, $u \in \Re$ est l'entrée du système, A et $\underline{b}$ sont donnés par :

$$A = \left[\begin{array}{cc} \underline{0}_{(n-1)\times 1} & I_{n-1} \\ \multicolumn{2}{c}{\underline{0}_{1\times n}} \end{array}\right], \underline{b} = \left[\begin{array}{c} 0 \\ \vdots \\ 0 \\ 1 \end{array}\right]$$

où $\underline{a} = \left[\begin{array}{cccc} a_1 & a_2 & .. & a_n \end{array}\right] \in \Re^{1\times n}$ est un vecteur de paramètres inconnus. Le paramètre b est supposé positif mais inconnu. De plus, la paire $(A, \underline{b})$ est contrôlable.

Le modèle de référence est défini par l'équation d'état LTI suivante :

$$\dot{\underline{x}}_m = A_m\underline{x}_m + \underline{b}b_m r \tag{3.2}$$

où $\underline{x}_m = \left[\begin{array}{cccc} x_{m1} & x_{m2} & .. & x_{mn} \end{array}\right] \in \Re^{n\times 1}$ est le vecteur d'état de référence, r est l'entrée de référence bornée, A_m est donnée par :

$$A_m = \left[\begin{array}{cc} \underline{0}_{(n-1)\times 1} & I_{n-1} \\ \multicolumn{2}{c}{-\underline{a}_m} \end{array}\right]$$

avec $\underline{a}_m = \begin{bmatrix} a_{m1} & a_{m2} & .. & a_{mn} \end{bmatrix} \in \Re^{1\times n}$ et $b_m > 0$. La matrice A_m est Hurwitz, c.-à-d. que le polynôme

$$p(s) = s^n + a_{mn}s^{n-1} + a_{m(n-1)}s^{n-2} + ... + a_{m2}s + a_{m1}$$

a toutes ses racines dans le demi-plan gauche du plan complexe. Donc, le modèle de référence (3.2) est stable avec une dynamique imposée par le choix des racines de $p(s)$.

L'objectif de la commande MRAC est de faire suivre au système (3.1) la dynamique imposée par le modèle de référence (3.2). Ce qui revient à dire que l'erreur de poursuite $\underline{e} = \underline{x}_m - \underline{x}$ doit tendre vers zéro et que tous les autres signaux de la boucle doivent restés bornés, et ceci en dépit de la méconnaissance des paramètres du système (3.1).

Pour commencer, la soustraction de (3.1) de (3.2) conduit à la dynamique d'erreur de poursuite suivante :

$$\dot{\underline{e}} = A_m\underline{x}_m - A\underline{x} + \underline{b}\left[b_m r - \underline{a}\underline{x} - bu\right] \tag{3.3}$$

En additionnant et soustrayant le terme $\underline{a}_m\underline{x}$ à la droite de (3.3), elle peut être arrangée comme :

$$\dot{\underline{e}} = A_m\underline{e} + \underline{b}\left[b_m r - (\underline{a}_m + \underline{a})\,\underline{x} - bu\right] \tag{3.4}$$

Maintenant, on se propose comme loi de commande le retour d'état défini par :

$$u = \underline{k}_1\underline{x} + k_2 r \tag{3.5}$$

avec $\underline{k}_1 = \begin{bmatrix} k_{11} & k_{12} & .. & k_{1n} \end{bmatrix} \in \Re^{1\times n}$. Alors, la dynamique de l'erreur de poursuite en boucle fermée sera donnée par :

$$\dot{\underline{e}} = A_m\underline{e} + \underline{b}b\left[\left(\frac{b_m}{b} - k_2\right)r - \left[\frac{\underline{a}_m + \underline{a}}{b} + \underline{k}_1\right]\underline{x}\right] \tag{3.6}$$

Il est clair, que si on peut annuler le deuxième terme à droite de (3.6), alors il ne reste que la dynamique

$$\dot{\underline{e}} = A_m\underline{e}$$

qui est stable est converge exponentiellement vers zéro. On peut atteindre cet objectif, si on choisit les gains du retour d'état comme :

$$\underline{k}_1^* = -\frac{\underline{a}_m + \underline{a}}{b} \tag{3.7}$$

$$k_2^* = \frac{b_m}{b} \tag{3.8}$$

Les conditions (3.7)-(3.8) sont appelées conditions de poursuite idéale. Malheureusement, comme $\underline{a}$ et b sont inconnus, ces conditions ne peuvent être implantées directement. Pour

contourner ce problème, nous ferons appel à des lois d'ajustement adaptatives pour estimer les valeurs de $\underline{k}_1^*$ et k_2^*.

En utilisant les relations (3.7)-(3.8), on peut écrire (3.6) comme :

$$\dot{\underline{e}} = A_m \underline{e} + \underline{b} b \left[\widetilde{k}_2 r + \underline{\widetilde{k}}_1 \underline{x} \right] \tag{3.9}$$

où $\underline{\widetilde{k}}_1 = \underline{k}_1^* - \underline{k}_1$ et $\widetilde{k}_2 = k_2^* - k_2$ sont les erreurs d'estimation sur les gains de la loi de commande.

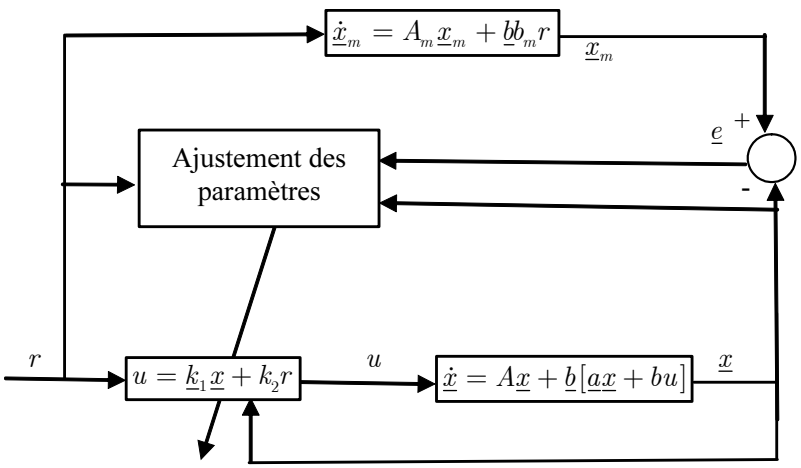


Figure 3.2 : Structure de la commande MRAC pour un système linéaire.

Pour vérifier la stabilité de la boucle fermée, et obtenir les lois d'ajustement des paramètres, on fait recours à la fonction de Lyapunov :

$$V = \frac{1}{2b} \underline{e}^T P \underline{e} + \frac{1}{2\gamma_1} \underline{\widetilde{k}}_1 \underline{\widetilde{k}}_1^T + \frac{1}{2\gamma_2} \widetilde{k}_2^2 \tag{3.10}$$

où $\gamma_1, \gamma_2 > 0$ sont des paramètres de conception, et P est une matrice définie positive donnée par la solution de l'équation de Lyapunov :

$$A_m^T P + P A_m = -Q \tag{3.11}$$

avec Q une matrice définie positive.

La dérivée de (3.10) le long de la trajectoire de (3.9) donne :

$$\dot{V} = -\frac{1}{2b} \underline{e}^T Q \underline{e} + \underline{e}^T P \underline{b} \widetilde{k}_2 r + \underline{e}^T P \underline{b} \underline{\widetilde{k}}_1 \underline{x} - \frac{1}{\gamma_1} \dot{\underline{k}}_1 \underline{\widetilde{k}}_1^T - \frac{1}{\gamma_2} \dot{k}_2 \widetilde{k}_2 \tag{3.12}$$

où nous avons utilisé le fait que : $\dot{\underline{\widetilde{k}}}_1 = -\dot{\underline{k}}_1$ et $\dot{\underline{\widetilde{k}}}_2 = -\dot{k}_2$.

L'équation (3.12) peut être arrangée comme :

$$\dot{V} = -\frac{1}{2b} \underline{e}^T Q \underline{e} - \frac{1}{\gamma_1} \left(\dot{\underline{k}}_1 - \gamma_1 \underline{b}^T P \underline{e} \underline{x}^T \right) \underline{\widetilde{k}}_1^T - \frac{1}{\gamma_2} \left(\dot{k}_2 - \gamma_2 \underline{b}^T P \underline{e} r \right) \widetilde{k}_2 \tag{3.13}$$

Le choix des lois d'ajustement des paramètres suivantes :

$$\begin{aligned} \dot{\underline{k}}_1 &= \gamma_1 \underline{b}^T P \underline{e} \underline{x}^T \\ \dot{k}_2 &= \gamma_2 \underline{b}^T P \underline{e} r \end{aligned} \tag{3.14}$$

permet d'avoir :

$$\dot{V} = -\frac{1}{2b} \underline{e}^T Q \underline{e} \tag{3.15}$$

qui est définie négative. Ce qui permet de conclure à la stabilité asymptotique de la boucle MRAC et la convergence de l'erreur de poursuite $\underline{e}$ vers zéro.

3.3 Commande MRAC des systèmes non linéaires

Comme montré dans la section précédente, la commande MRAC des systèmes linéaires ne pose pas de difficulté particulière. Pour les systèmes non linéaires, la question est tout à fait différente, surtout si ces systèmes sont de plus inconnus et sujets à des incertitudes et à des perturbations externes. Dans la présente section, nous allons développer une technique de commande MRAC basée sur des réseaux de neurones adaptatifs. Nous montrerons aussi que cette technique est stable et robuste vis à vis des incertitudes et des perturbations.

3.3.1 Position du problème

Nous considérerons la classe de systèmes non linéaires continus donnée par :

$$\dot{\underline{x}} = A\underline{x} + \underline{b}\left[f(\underline{x}) + g(\underline{x})u + d(\underline{x}, t)\right] \tag{3.16}$$

où $\underline{x}^T = \left[\begin{array}{cccc} x_1 & x_2 & .. & x_n \end{array}\right] \in \Re^n$ est le vecteur d'état, $u \in \Re$ est l'entrée du système, $f(\underline{x})$, $g(\underline{x})$ sont des fonctions non linéaires inconnues mais lisses. $d(\underline{x}, t)$ est le terme qui regroupe les incertitudes et les perturbations externes.

Le système non linéaire (3.16) satisfait les hypothèses suivantes :

P1. Le gain d'entrée $g(\underline{x})$ est strictement positif dans l'espace de commande $\Omega_{\underline{x}}$. Le cas négatif peut aussi être considéré.

P2. Le terme d'incertitudes et perturbations $d(\underline{x}, t)$ est borné.

P3. La fonction non linéaire $f(\underline{x})$ peut être décomposée comme :

$$f(\underline{x}) = \underline{a}(\underline{x})\,\underline{x} + a_0(\underline{x}) \tag{3.17}$$

avec $\underline{a}(\underline{x}) = \left[\begin{array}{cccc} a_1(\underline{x}) & a_2(\underline{x}) & \ldots & a_n(\underline{x}) \end{array}\right] \in \Re^n$ un vecteur de fonctions non linéaires inconnues, et $a_0(\underline{x})$ est une fonction non linéaire bornée.

Notons que P1 est une hypothèse usuelle dans la commande adaptative, et l'hypothèse P3 spécifie la classe des systèmes non linéaires considérée. Notons aussi que, du point de vue pratique, une large classe de systèmes réels satisfait l'hypothèse P3, p. ex., les robots manipulateurs, les machines électriques, etc.

Le modèle de référence est toujours défini par l'équation d'état (3.2).

La soustraction (3.16) de (3.2) et l'utilisation de l'hypothèse P2 conduit à la dynamique de l'erreur de poursuite suivante :

$$\dot{\underline{e}} = A_m \underline{e} + \underline{b}\left[-\left(\underline{a}_m + \underline{a}(\underline{x})\right)\underline{x} + b_m r - g(\underline{x})u - a_0(\underline{x}) - d(\underline{x},t)\right] \tag{3.18}$$

Le problème de commande peut être énoncé comme suit : Concevoir une loi de commande $u(\underline{x})$ tel que les états du système (3.16) suivent ceux du modèle de référence (3.2), avec la condition que tous les signaux en boucle fermée restent bornés.

3.3.2 Conception du retour d'état neuronal

La commande MRAC neuronale proposée est illustrée sur la figure 3.3.

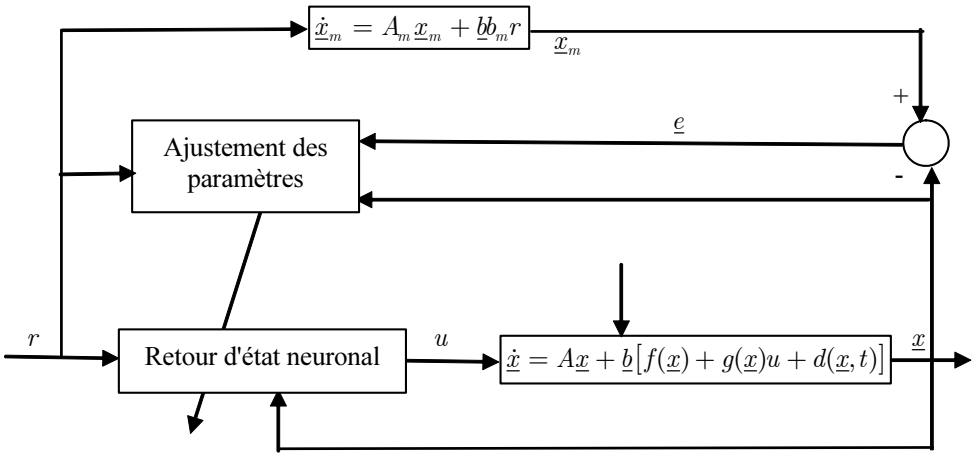


Figure 3.3 : Commande MRAC neuronale des systèmes non linéaires.

La structure du retour d'état neuronal est détaillée dans la figure 3.4.

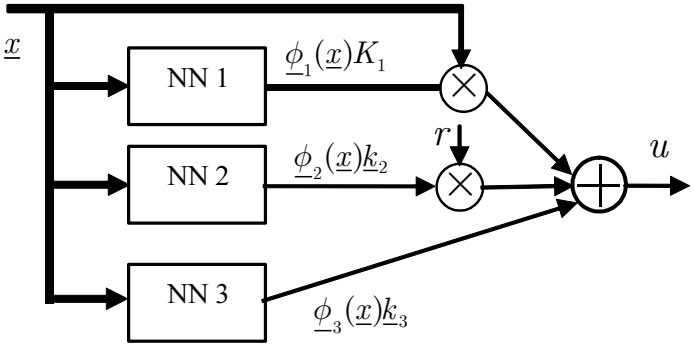


Figure 3.4 : Retour d'état neuronal.

L'architecture de commande proposée est composée de trois réseaux de neurones à une seule couche cachée avec des fonctions d'activation bornées (qui peuvent être sélectionnées parmi les fonctions d'activation décrites dans le chapitre 2). Le réseau NN1 est un réseau multi-sorties (n sorties) qui, multipliées avec le vecteur d'état, fournit le terme $\underline{\phi}_1(\underline{x})\, K_1\underline{x}$. Le réseau NN2 est un réseau mono-sortie qui, après multiplication avec la consigne r, produit le terme $\underline{\phi}_2(\underline{x})\,\underline{k}_2 r$. Enfin, le réseau NN3 est un réseau mono-sortie qui produit le terme $\underline{\phi}_3(\underline{x})\,\underline{k}_3$. La commande, qui est la somme des actions des trois réseaux de neurones adaptatifs, est donnée par :

$$u = \underline{\phi}_1(\underline{x})\, K_1\underline{x} + \underline{\phi}_2(\underline{x})\,\underline{k}_2 r + \underline{\phi}_3(\underline{x})\,\underline{k}_3 \tag{3.19}$$

où $\underline{\phi}_1(\underline{x}) \in \Re^{1\times q_1}$, $\underline{\phi}_2(\underline{x}) \in \Re^{1\times q_2}$ et $\underline{\phi}_3(\underline{x}) \in \Re^{1\times q_3}$ sont les vecteurs des sorties des neurones de la couche cachée dans les trois réseaux, respectivement, avec q_1, q_2 et q_3 les nombres de neurones dans la couche cachée de chaque réseau. Les poids $K_1 \in \Re^{q_1\times n}$, $\underline{k}_2 \in \Re^{q_2\times 1}$ et $\underline{k}_3 \in \Re^{q_3\times 1}$ sont, respectivement, les paramètres ajustables des trois réseaux. Le premier et le second termes dans (3.19) sont des versions non linéaires de la commande avec retour d'état standard (3.5), le troisième terme est ajouté pour compenser les termes $a_0(\underline{x})$ et $d(\underline{x},t)$.

A ce stade, l'introduction de la loi de commande (3.19) dans (3.18) produit :

$$\begin{aligned}\dot{\underline{e}} =& A_m\underline{e} - \underline{b}\Big[\Big(\underline{a}_m + \underline{a}(\underline{x}) + g(\underline{x})\underline{\phi}_1(\underline{x})\, K_1\Big)\underline{x} + \Big(g(\underline{x})\underline{\phi}_2(\underline{x})\,\underline{k}_2 - b_m\Big) r \\ &+ \Big(g(\underline{x})\underline{\phi}_3(\underline{x})\,\underline{k}_3 + (a_0(\underline{x}) + d(\underline{x},t))\Big)\Big]\end{aligned} \tag{3.20}$$

A partir des résultats de l'approximation universelle (voir chapitre 2), il existe des paramètres optimaux K_1^*, $\underline{k}_2^*$ et $\underline{k}_3^*$ tel que :

$$[\underline{a}_m + \underline{a}(\underline{x})] + g(\underline{x})\underline{\phi}_1(\underline{x})K_1^* = \underline{\epsilon}_1(\underline{x}) \tag{3.21}$$

$$g(\underline{x})\underline{\phi}_2(x)\,\underline{k}_2^* - b_m = \epsilon_2(\underline{x}) \tag{3.22}$$

$$(a_0(\underline{x}) + d(\underline{x},t)) + g(\underline{x})\underline{\phi}_3(\underline{x})\underline{k}_3^* = \epsilon_3(\underline{x}) \tag{3.23}$$

où $\underline{\epsilon}_1(\underline{x}) \in \Re^{1\times n}$ et $\epsilon_2(\underline{x}), \epsilon_3(\underline{x}) \in \Re$ sont les erreurs d'approximation idéales pour les trois réseaux de neurones dans (3.19), et les paramètres optimaux K_1^*, $\underline{k}_2^*$ et $\underline{k}_3^*$ sont définis par :

$$K_1^* = \arg\min_{K_1\in\Omega_1}\left\{\sup_{\underline{x}\in\Omega_{\underline{x}}}\left|[\underline{a}_m + \underline{a}(\underline{x})] + g(\underline{x})\underline{\phi}_1(\underline{x})K_1^*\right|\right\}$$

$$\underline{k}_2^* = \arg\min_{\underline{k}_2\in\Omega_2}\left\{\sup_{\underline{x}\in\Omega_{\underline{x}}}\left|g(\underline{x})\underline{\phi}_2(x)\,\underline{k}_2^* - b_m\right|\right\}$$

$$\underline{k}_3^* = \arg\min_{\underline{k}_3\in\Omega_3}\left\{\sup_{\underline{x}\in\Omega_{\underline{x}}}\left|(a_0(\underline{x}) + d(\underline{x},t)) + g(\underline{x})\underline{\phi}_3(\underline{x})\underline{k}_3^*\right|\right\}$$

où Ω_1, Ω_2, Ω_3 et $\Omega_{\underline{x}}$ sont des ensembles de contraintes pour K_1, $\underline{k}_2$, $\underline{k}_3$ et $\underline{x}$, respectivement.

Les propriétés (3.21)-(3.23) peuvent être vue comme une version non linéaire des propriétés (3.7)-(3.8) énoncées pour la commande MRAC des systèmes linéaires.

Le contrôleur adaptatif neuronal (3.19) vérifie les hypothèses suivantes :

C1. Les fonctions d'activation du réseau de neurones NN2 $\underline{\phi}_2(\underline{x})$ sont telles que $\left|\dot{\underline{\phi}}_2(\underline{x})\right| < \varphi_0$, où φ_0 est une constante positive connue, et $\underline{\phi}_2(\underline{x})\,\underline{k}_2^* > 0$.

C2. Les fonctions d'activation des réseaux de neurones sont telles que $\left|\underline{\phi}_i(\underline{x})\right| < 1, i = 1,2,3$.

C3. Les majorations suivantes : $\Omega_1 = \{K_1 | \|K_1\| < \kappa_1\}$, $\Omega_2 = \{\underline{k}_2 | |\underline{k}_2| < \kappa_2\}$ et $\Omega_3 = \{\underline{k}_3 | |\underline{k}_3| < \kappa_3\}$, avec $\kappa_1, \kappa_2, \kappa_3 > 0$, sont données.

C4. Les erreurs d'approximation sont bornées par $|\underline{\epsilon}_1(\underline{x})| \leq \epsilon_1$, $|\epsilon_2(\underline{x})| \leq \epsilon_2$ et $|\epsilon_3(\underline{x})| \leq \epsilon_3$, pour certaines constantes positives ϵ_1, ϵ_1 et ϵ_3.

L'hypothèse C1 indique que les fonctions d'activation du réseau NN2 $\underline{\phi}_2(\underline{x})$ devraient être choisies comme continues (différentiables). Notons que $\dot{\underline{\phi}}_2(\underline{x})$ n'est pas nécessairement globalement bornée, mais elle aura une borne constante dans $\Omega_{\underline{x}}$ due à la continuité de $\underline{\phi}_2(\underline{x})$. L'hypothèse C2 est vérifiée pour les fonctions d'activation sigmoïde, hyperbolique et RBF gaussienne. L'hypothèse C3 est émise pour assurer la bornitude des paramètres ajustables du contrôleur neuronal. L'hypothèse C4 découle de la propriété d'approximation universelle des réseaux de neurones.

Ainsi, l'utilisation des propriétés (3.21)-(3.23) dans (3.20) fournit :

$$\begin{aligned}\dot{\underline{e}} =& A_m\underline{e} + g(\underline{x})\underline{b}\left[\underline{\phi}_1(\underline{x})\,\widetilde{K}_1\underline{x} + \underline{\phi}_2(\underline{x})\,\widetilde{\underline{k}}_2 r + \underline{\phi}_3(\underline{x})\,\widetilde{\underline{k}}_3\right] \\ & - \underline{b}\left[\underline{\epsilon}_1(\underline{x})\,\underline{x} + \epsilon_2(\underline{x})\,r + \epsilon_3(\underline{x})\right] \end{aligned} \tag{3.24}$$

où $\widetilde{K}_1 = K_1^* - K_1$, $\widetilde{\underline{k}}_2 = \underline{k}_2^* - \underline{k}_2$ et $\widetilde{\underline{k}}_3 = \underline{k}_3^* - \underline{k}_3$ sont les erreurs d'estimation des paramètres des réseaux de neurones.

En exploitant la propriété (3.22), nous avons :

$$g(\underline{x}) = \frac{b_m + \epsilon_2(\underline{x})}{\underline{\phi}_2(x)\,\underline{k}_2^*} \tag{3.25}$$

Ce qui permet d'arranger l'équation (3.24) comme :

$$\dot{\underline{e}} = A_m\underline{e} + \frac{b_m}{\underline{\phi}_2(x)\,\underline{k}_2^*}\underline{b}\left[\underline{\phi}_1(\underline{x})\,\widetilde{K}_1\underline{x} + \underline{\phi}_2(\underline{x})\,\widetilde{\underline{k}}_2 r + \underline{\phi}_3(\underline{x})\,\widetilde{\underline{k}}_3\right] + \frac{1}{\underline{\phi}_2(x)\,\underline{k}_2^*}\underline{b}\left[\underline{\alpha}_1\underline{e} + \xi\right] \tag{3.26}$$

avec

$$\begin{aligned} \xi &= -\underline{\alpha}_1\underline{x}_m + \alpha_2 r + \alpha_3 \\ \underline{\alpha}_1 &= \underline{\epsilon}_1(\underline{x})\,\underline{\phi}_2(\underline{x})\underline{k}_2^* - \epsilon_2(\underline{x})\,\underline{\phi}_1(\underline{x})\,\widetilde{K}_1 \\ \alpha_2 &= -\epsilon_2(\underline{x})\,\underline{\phi}_2(\underline{x})\underline{k}_2 \\ \alpha_3 &= \epsilon_2(\underline{x})\,\underline{\phi}_3(\underline{x})\widetilde{\underline{k}}_3 - \epsilon_3(\underline{x})\,\underline{\phi}_2(\underline{x})\underline{k}_2^* \end{aligned}$$

Il est clair que la dynamique de l'erreur de poursuite (3.26) est gouvernée par trois termes : le premier terme est exponentiellement stable, le deuxième terme qui dépend des erreurs sur les paramètres du contrôleur neuronal adaptatif, et le troisième terme qui dépend des erreurs d'approximation réalisées par les réseaux de neurones.

3.3.3 Lois d'ajustement des paramètres

La bornitude des paramètres ajustables est, comme nous le verrons par la suite, une condition nécessaire pour l'établissement de la stabilité de la boucle de commande. Comme déjà introduit au chapitre 2, on peut utiliser les algorithmes du gradient avec projection pour assurer la bornitude des paramètres ajustés. A cet effet, les paramètres du contrôleur neuronal sont ajustés en utilisant l'algorithme suivant :

$$\dot{K}_1 = \gamma_1 b_m \underline{e}^T P \underline{b} \left(\underline{\phi}_1^T(\underline{x}) \underline{x}^T - I_1 \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x}) \underline{x}^T \right] \left(\frac{1 + \|K_1\|}{\kappa_1} \right)^2 K_1 \right) \tag{3.27}$$

$$\dot{\underline{k}}_2 = \gamma_2 b_m \underline{e}^T P \underline{b} \left(\underline{\phi}_2^T(\underline{x}) - I_2 \frac{\underline{k}_2 \underline{k}_2^T \underline{\phi}_2^T(\underline{x})}{|\underline{k}_2|^2} \right) r \tag{3.28}$$

$$\dot{\underline{k}}_3 = \gamma_3 b_m \underline{e}^T P \underline{b} \left(\underline{\phi}_3^T(\underline{x}) - I_3 \frac{\underline{k}_3 \underline{k}_3^T \underline{\phi}_3^T(\underline{x})}{|\underline{k}_3|^2} \right) \tag{3.29}$$

où

$$I_1 = \begin{cases} 0, \text{ si } \|K_1\| < \kappa_1 \text{ ou } \left(\|K_1\| = \kappa_1 \text{ et } \underline{e}^T P \underline{b} \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x}) \underline{x}^T \right] \leq 0 \right) \\ 1, \text{ si } \|K_1\| = \kappa_1 \text{ et } \underline{e}^T P \underline{b} \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x}) \underline{x}^T \right] > 0 \end{cases}$$

$$I_2 = \begin{cases} 0, \text{ si } |\underline{k}_2| < \kappa_2 \text{ ou } \left(|\underline{k}_2| = \kappa_2 \text{ et } \underline{e}^T P \underline{b} \underline{\phi}_2(\underline{x}) \underline{k}_2 r \leq 0 \right) \\ 1, \text{ si } |\underline{k}_2| = \kappa_2 \text{ et } \underline{e}^T P \underline{b} \underline{\phi}_2(\underline{x}) \underline{k}_2 r > 0 \end{cases}$$

$$I_3 = \begin{cases} 0, \text{ si } |\underline{k}_3| < \kappa_3 \text{ ou } \left(|\underline{k}_3| = \kappa_3 \text{ et } \underline{e}^T P \underline{b} \underline{\phi}_3(\underline{x}) \underline{k}_3 \leq 0 \right) \\ 1, \text{ si } |\underline{k}_3| = \kappa_3 \text{ et } \underline{e}^T P \underline{b} \underline{\phi}_3(\underline{x}) \underline{k}_3 > 0 \end{cases}$$

et $\gamma_1, \gamma_2, \gamma_3 > 0$ sont les pas d'ajustement sélectionnés par le concepteur, et $P > 0$ est une matrice définie positive qu'on déterminera par la suite.

3.4 Analyse de stabilité

Une approche de commande ne peut être considérée comme valide que si sa stabilité est prouvée. Dans cette section nous allons démontrer que la commande MRAC neuronale développée dans la section précédente est stable. Pour démontrer la stabilité de la commande MRAC neuronale, nous allons procéder en deux étapes : En premier lieu nous démontrerons que les lois d'ajustement (3.27)-(3.29) gardent les paramètres ajustables bornés ; et en deuxième lieu, nous démontrerons la stabilité de la dynamique de l'erreur en boucle fermée.

La bornitude des paramètres est énoncée par le théorème suivant.

Théorème 3.1 :

Les lois d'ajustement (3.27)-(3.29) assurent que : $\|K_1\| \leq \kappa_1$, $|\underline{k}_2| \leq \kappa_2$ et $|\underline{k}_3| \leq \kappa_3$ $\forall t \geq 0$, si les valeurs initiales des paramètres $K_1(0)$, $\underline{k}_2(0)$ et $\underline{k}_3(0)$ sont sélectionnées correctement.

Preuve :

Pour prouver que $\|K_1\| \leq \kappa_1$, considérons la fonction de Lyapunov :

$$V_1 = \frac{1}{2}\|K_1\|^2 = \frac{1}{2}\mathrm{tr}\left[K_1^T K_1\right] \tag{3.30}$$

alors

$$\dot{V}_1 = \frac{1}{2}\frac{d\|K_1\|^2}{dt} = \mathrm{tr}\left[K_1^T \dot{K}_1\right] \tag{3.31}$$

Si $I_1 = 0$, nous avons $\|K_1\| < \kappa_1$ ou $\|K_1\| = \kappa_1$ et

$$\dot{V}_1 = \gamma_1 b_m \underline{e}^T P \underline{b} \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x})\underline{x}^T\right] \leq 0 \tag{3.32}$$

c.-à-d., que nous avons toujours $\|K_1\| \leq \kappa_1$.

Si $I_1 = 1$, nous avons $\|K_1\| = \kappa_1$ et

$$\begin{aligned}
\dot{V}_1 =& \gamma_1 b_m \underline{e}^T P \underline{b} \mathrm{tr}\left[K_1^T \left(\underline{\phi}_1^T(\underline{x})\underline{x}^T - \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x})\underline{x}^T\right]\left(\frac{1+\|K_1\|}{\kappa_1}\right)^2 K_1\right)\right] \\
=& \gamma_1 b_m \underline{e}^T P \underline{b}\left(\mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x})\underline{x}^T\right] - \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x})\underline{x}^T\right]\left(\frac{1+\|K_1\|}{\kappa_1}\right)^2 \mathrm{tr}\left[K_1^T K_1\right]\right) \\
=& \gamma_1 b_m \underline{e}^T P \underline{b} \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x})\underline{x}^T\right]\left(1 - \left(\frac{1+\|K_1\|}{\kappa_1}\right)^2 \mathrm{tr}\left[K_1^T K_1\right]\right)
\end{aligned} \tag{3.33}$$

Puisque $\underline{e}^T P \underline{b} \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{x})\underline{x}^T\right] > 0$ et $\|K_1\| = \kappa_1$, nous obtenons $\dot{V}_1 \leq 0$, ce qui implique que $\|K_1\| \leq \kappa_1$. Alors, nous obtenons $\|K_1\| \leq \kappa_1$, $\forall t \geq 0$.□

Une analyse similaire peut être utilisée pour montrer que $|\underline{k}_2| \leq \kappa_2$ et $|\underline{k}_3| \leq \kappa_3$ $\forall t \geq 0$.

Pour la démonstration de la stabilité de la dynamique de l'erreur en boucle fermée, considérons la fonction de Lyapunov :

$$V = \frac{1}{2}\underline{\phi}_2(\underline{x})\underline{k}_2^* \underline{e}^T P \underline{e} + \frac{1}{2\gamma_1}\mathrm{tr}\left[\widetilde{K}_1^T \widetilde{K}_1\right] + \frac{1}{2\gamma_2}\widetilde{\underline{k}}_2^T \widetilde{\underline{k}}_2 + \frac{1}{2\gamma_3}\widetilde{\underline{k}}_3^T \widetilde{\underline{k}}_3 \tag{3.34}$$

La dérivée de (3.34) le long de la trajectoire de (3.26) donne :

$$\begin{aligned}
\dot{V} =& \frac{1}{2}\underline{\phi}_2(\underline{x})\underline{k}_2^* \underline{e}^T\left(A_m^T P + P A_m\right)\underline{e} + \frac{1}{2}\dot{\underline{\phi}}_2(\underline{x})\underline{k}_2^* \underline{e}^T P \underline{e} + \underline{e}^T P \underline{b}\alpha_1 \underline{e} + \underline{e}^T P \underline{b}\xi \\
&+ b_m \underline{e}^T P \underline{b}\left[\underline{\phi}_1(\underline{x})\widetilde{K}_1 \underline{x} + \underline{\phi}_2(\underline{x})\widetilde{\underline{k}}_2 r + \underline{\phi}_3(\underline{x})\widetilde{\underline{k}}_3\right] \\
&+ \frac{1}{\gamma_1}\mathrm{tr}\left[\widetilde{K}_1^T \dot{\widetilde{K}}_1\right] + \frac{1}{\gamma_2}\widetilde{\underline{k}}_2^T \dot{\widetilde{\underline{k}}}_2 + \frac{1}{\gamma_3}\widetilde{\underline{k}}_3^T \dot{\widetilde{\underline{k}}}_3
\end{aligned} \tag{3.35}$$

En utilisant le fait que $\dot{\widetilde{K}}_1 = -\dot{K}_1, \dot{\widetilde{\underline{k}}}_2 = -\dot{\underline{k}}_2$ et $\dot{\widetilde{\underline{k}}}_3 = -\dot{\underline{k}}_3$, (3.35) peut être arrangée comme :

$$\begin{aligned}\dot{V} = &\frac{1}{2}\underline{\phi}_2(\underline{x})\underline{k}_2^*\underline{e}^T\left(A_m^TP + PA_m\right)\underline{e} + \frac{1}{2}\dot{\underline{\phi}}_2(\underline{x})\underline{k}_2^*\underline{e}^TP\underline{e}\\ &+ \underline{e}^TP\underline{b}\underline{\alpha}_1\underline{e} + \underline{e}^TP\underline{b}\xi + \frac{1}{\gamma_1}\mathrm{tr}\left[\widetilde{K}_1^T\left(\gamma_1 b_m\underline{e}^TP\underline{b}\underline{\phi}_1^T(\underline{x})\underline{x}^T - \dot{K}_1\right)\right]\\ &+ \frac{1}{\gamma_2}\widetilde{\underline{k}}_2^T\left(\gamma_2 b_m\underline{e}^TP\underline{b}\underline{\phi}_2^T(\underline{x})r - \dot{\underline{k}}_2\right) + \frac{1}{\gamma_3}\widetilde{\underline{k}}_3^T\left(\gamma_3 b_m\underline{e}^TP\underline{b}\underline{\phi}_3^T(\underline{x}) - \dot{\underline{k}}_3\right)\end{aligned} \tag{3.36}$$

Alors, l'introduction des lois d'ajustement (3.27)-(3.29) dans (3.36) donne :

$$\begin{aligned}\dot{V} = &\frac{1}{2}\underline{\phi}_2(\underline{x})\underline{k}_2^*\underline{e}^T\left(A_m^TP + PA_m\right)\underline{e} + \frac{1}{2}\dot{\underline{\phi}}_2(\underline{x})\underline{k}_2^*\underline{e}^TP\underline{e} + \underline{e}^TP\underline{b}\underline{\alpha}_1\underline{e} + \underline{e}^TP\underline{b}\xi\\ &+ I_1\mathrm{tr}\left[\widetilde{K}_1^Tb_m\underline{e}^TP\underline{b}\mathrm{tr}\left[K_1^T\underline{\phi}_1^T(\underline{x})\underline{x}^T\right]\left(\frac{1+\|K_1\|}{\kappa_1}\right)^2K_1\right]\\ &+ I_2\widetilde{\underline{k}}_2^T\frac{b_m\underline{e}^TP\underline{b}\underline{k}_2\underline{k}_2^T\underline{\phi}_2^T(\underline{x})}{|\underline{k}_2|^2}r + I_3\widetilde{\underline{k}}_3^T\frac{b_m\underline{e}^TP\underline{b}\underline{k}_3\underline{k}_3^T\underline{\phi}_3^T(\underline{x})}{|\underline{k}_3|^2}\end{aligned} \tag{3.37}$$

qui peut, à son tour, être arrangée comme :

$$\begin{aligned}\dot{V} = &\frac{1}{2}\underline{\phi}_2(\underline{x})\underline{k}_2^*\underline{e}^T\left(A_m^TP + PA_m\right)\underline{e} + \frac{1}{2}\dot{\underline{\phi}}_2(\underline{x})\underline{k}_2^*\underline{e}^TP\underline{e} + \underline{e}^TP\underline{b}\underline{\alpha}_1\underline{e} + \underline{e}^TP\underline{b}\xi\\ &+ I_1b_m\underline{e}^TP\underline{b}\mathrm{tr}\left[\widetilde{K}_1^TK_1\right]\mathrm{tr}\left[K_1^T\underline{\phi}_1^T(\underline{x})\underline{x}^T\right]\left(\frac{1+\|K_1\|}{\kappa_1}\right)^2\\ &+ I_2\widetilde{\underline{k}}_2^T\underline{k}_2\frac{b_m\underline{e}^TP\underline{b}\underline{k}_2^T\underline{\phi}_2^T(\underline{x})}{|\underline{k}_2|^2}r + I_3\widetilde{\underline{k}}_3^T\underline{k}_3\frac{b_m\underline{e}^TP\underline{b}\underline{k}_3^T\underline{\phi}_3^T(\underline{x})}{|\underline{k}_3|^2}\end{aligned} \tag{3.38}$$

Nous prouverons maintenant que les trois derniers termes dans (3.38) sont toujours ≤ 0.

Si $I_1 = I_2 = I_3 = 0$ le résultat est trivial.

Si $I_1 = 1$, alors $\|K_1\| = \kappa_1$ et $\underline{e}^TP\underline{b}\mathrm{tr}\left[K_1^T\underline{\phi}_1^T(\underline{x})\underline{x}^T\right] > 0$. D'autre part, nous avons :

$$\mathrm{tr}\left[\widetilde{K}_1^TK_1\right] = \frac{1}{2}\mathrm{tr}\left[K_1^{*T}K_1^*\right] - \frac{1}{2}\mathrm{tr}\left[K_1^TK_1\right] - \frac{1}{2}\mathrm{tr}\left[\widetilde{K}_1^T\widetilde{K}_1\right] \tag{3.39}$$

Puisque $\mathrm{tr}\left[K_1^{*T}K_1^*\right] \leq \kappa_1^2$, $\mathrm{tr}\left[K_1^TK_1\right] = \kappa_1^2$ et $\mathrm{tr}\left[\widetilde{K}_1^T\widetilde{K}_1\right] \geq 0$ nous obtenons $\mathrm{tr}\left[\widetilde{K}_1^TK_1\right] \leq 0$, ce qui implique que le cinquième terme dans (3.38) est toujours ≤ 0.

Une même analyse peut être utilisée pour montrer que les deux derniers termes dans (3.38) sont aussi ≤ 0.

Avec le résultat précédent, (3.38) peut être réécrite comme :

$$\dot{V} \leq \frac{1}{2}\underline{\phi}_2(\underline{x})\underline{k}_2^*\underline{e}^T\left(A_m^TP + PA_m\right)\underline{e} + \frac{1}{2}\dot{\underline{\phi}}_2(\underline{x})\underline{k}_2^*\underline{e}^TP\underline{e} + \underline{e}^TP\underline{b}\underline{\alpha}_1\underline{e} + \underline{e}^TP\underline{b}\xi \tag{3.40}$$

Maintenant, en utilisant C1 et le fait que $\left|\underline{\phi}_2(\underline{x})\right| \leq 1$ et $\|\underline{b}\underline{\alpha}_1\| \leq \alpha_0$ (où α_0 est une borne supérieure donnée), dans (3.40) nous aurons :

$$\dot{V} \leq \frac{1}{2}\kappa_2 \underline{e}^T \left(A_m^T P + P A_m\right) \underline{e} + \frac{1}{2}\varphi_0 \kappa_2 \underline{e}^T P \underline{e} + \alpha_0 \underline{e}^T P \underline{e} + \underline{e}^T P \underline{b} \xi \tag{3.41}$$

Finalement, (3.41) peut être arrangée comme :

$$\dot{V} \leq \frac{1}{2}\kappa_2 \underline{e}^T \left(A_m^T P + P A_m + \left(\varphi_0 + 2\frac{\alpha_0}{\kappa_2}\right) P\right) \underline{e} + \underline{e}^T P \underline{b} \xi \tag{3.42}$$

Si la matrice A_m est choisie telle que :

$$A_m^T P + P A_m + \left(\varphi_0 + 2\frac{\alpha_0}{\kappa_2}\right) P \leq -Q \tag{3.43}$$

pour une certaine matrice définie positive Q, alors (3.42) devient :

$$\dot{V} \leq -\frac{1}{2}\kappa_2 \underline{e}^T Q \underline{e} + \underline{e}^T P \underline{b} \xi \tag{3.44}$$

qui peut être bornée par :

$$\dot{V} \leq -\frac{1}{2}\kappa_2 \lambda_{\min Q} |\underline{e}|^2 + |P\underline{b}|\,|\underline{e}|\,|\xi| \tag{3.45}$$

où $\lambda_{\min Q}$ est la plus petite valeur propre de Q.

Alors, $\dot{V} \leq 0$ en dehors de la région bornée définie par :

$$\Omega_{\underline{e}} = \left\{\underline{e} \,|\, |\underline{e}| \leq \frac{2\,|P\underline{b}|}{\kappa_2 \lambda_{\min Q}} |\xi|\right\} \tag{3.46}$$

La région de convergence $\Omega_{\underline{e}}$ (3.46) est illustrée sur la figure 3.5. Notons aussi que la taille de $\Omega_{\underline{e}}$ dépend à la fois des amplitudes des erreurs d'approximation (représentées par $|\xi|$) et des paramètres de conception κ_2, P et Q.

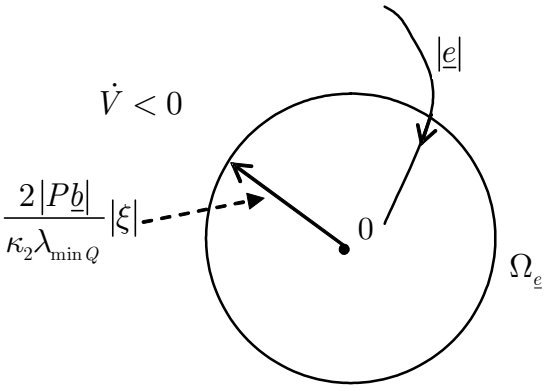


Figure 3.5 : Représentation générique de la région de convergence $\Omega_{\underline{e}}$ dans l'espace d'erreur.

Les propriétés de la commande par retour d'état neuronal adaptatif proposée sont résumées par le théorème suivant.

Théorème 3.2 :

Le système de commande composé par le système non linéaire (3.16) satisfaisant P1-P3, le modèle de référence (3.2), le retour d'état neuronal (3.19) satisfaisant C1-C4, les lois d'ajustement (3.27)-(3.29), et les matrices P, Q satisfaisant (3.43), garantit ce qui suit :

1.
$$|\underline{x}| \leq |\underline{x}_m| + c_1 |\xi| \tag{3.47}$$

2.
$$|u| \leq \kappa_1 |\underline{x}_m| + \kappa_2 |r| + \kappa_3 + c_2 |\xi| \tag{3.48}$$

3.
$$\int_0^T |\underline{e}|^2 \, dt \leq c_3 + c_4 \int_0^T |\xi|^2 \, dt \tag{3.49}$$

Preuve :

1. L'utilisation de (3.46) et du fait que $|\underline{x}| \leq |\underline{x}_m| + |\underline{e}|$ fournit :
$$|\underline{x}| \leq |\underline{x}_m| + \frac{2 |P\underline{b}|}{\kappa_2 \lambda_{\min Q}} |\xi| \tag{3.50}$$
Alors (3.47) est obtenue en posant $c_1 = \dfrac{2 |P\underline{b}|}{\kappa_2 \lambda_{\min Q}}$.

2. A partir de (3.19) et de la bornitude des paramètres ajustables (prouvée ci-dessus), la commande peut être majorée par :
$$|u| \leq \kappa_1 |\underline{x}| + \kappa_2 |r| + \kappa_3 \tag{3.51}$$
Alors, l'introduction de (3.50) dans (3.51) devient :
$$|u| \leq \kappa_1 |\underline{x}_m| + \kappa_2 |r| + \kappa_3 + \kappa_1 c_1 |\xi| \tag{3.52}$$
Alors (3.48) est prouvée avec $c_2 = \kappa_1 c_1$.

3. A partir de (3.46), nous avons $\underline{e} \in L_\infty$, et puisque K_1, $\underline{k}_2, \underline{k}_3 \in L_\infty$, alors $V \in L_\infty$. Puisque tous les termes dans (3.26) sont bornés, $\dot{\underline{e}} \in L_\infty$, et $\underline{e}$ est uniformément continue. L'utilisation du résultat (3.46) dans (3.45) produit :
$$\dot{V} \leq -\frac{1}{2} \kappa_2 \lambda_{\min Q} |\underline{e}|^2 + \frac{2 |P\underline{b}|^2}{\kappa_2 \lambda_{\min Q}} |\xi|^2 \tag{3.53}$$
L'intégration des deux membres de (3.53) donne :
$$\int_0^T |\underline{e}|^2 \, dt \leq \frac{2}{\kappa_2 \lambda_{\min Q}} V(0) + \left(\frac{2 |P\underline{b}|}{\kappa_2 \lambda_{\min Q}} \right)^2 \int_0^T |\xi|^2 \, dt \tag{3.54}$$
En posant $c_3 = \dfrac{2}{\kappa_2 \lambda_{\min Q}} V(0)$ et $c_4 = \left(\dfrac{2 |P\underline{b}|}{\kappa_2 \lambda_{\min Q}} \right)^2$, nous obtenons (3.49).□

3.5 Remarques

Pour plus de détails, les remarques suivantes s'imposent :

1. Nous notons que les réseaux de neurones utilisés peuvent être des réseaux RBF ou réseaux MLP ou une combinaison des deux types de réseaux.
2. Une architecture avec un seul réseau de neurones avec plusieurs sorties est possible, c.-à-d. $\underline{\phi}_1(\underline{x}) = \underline{\phi}_2(\underline{x}) = \underline{\phi}_3(\underline{x}) = \underline{\phi}(\underline{x})$. Ce qui simplifie la conception de la commande neuronale et l'ajustement des paramètres, ce que nous verrons dans l'exemple 2 de la section simulation. Cependant, en pratique les fonctions $f(\underline{x})$ et $g(\underline{x})$ peuvent dépendre de variables d'état différentes, et si nous avons cette information au préalable, nous pouvons favoriser une architecture multi-réseaux avec une complexité réduite.
3. A titre de comparaison, les approches MRAC neuronales proposées dans [18] et [42] traitent des systèmes non linéaires avec un gain d'entrée constant, c.-à-d. $g(\underline{x}) = cst$, ce qui limite leurs champs d'applications.
4. Comparée à l'approche MRAC neuronale proposée dans [43] , cette approche est plus simple puisque seulement le retour d'état neuronal est employé, et aucun terme de commande robuste n'est exigé pour assurer la bornitude des variables.
5. Comparée avec [44], où la borne supérieure sur $\dot{g}(\underline{x})$ est nécessaire pour atteindre l'objectif de la commande, cette structure nécessite uniquement la borne supérieure sur $\dot{\underline{\phi}}_2(\underline{x})$, qui est conçue par l'utilisateur et donc disponible pour l'évaluation.
6. Il peut être montré en utilisant le lemme de Barbalat [28] (voir annexe) que si $\xi \in L_2$, alors l'erreur de poursuite converge asymptotiquement vers zéro, c.-à-d., $\lim_{t\to\infty} \underline{e}(t) = 0$.

3.6 Résultats de simulation

La validité de la structure de commande neuronale adaptative proposée est vérifiée sur deux systèmes tests. Le premier est un système chaotique et le deuxième est le pendule inversé, très populaire dans les évaluations des techniques nouvelles de commande.

3.6.1 Exemple 1

Pour ce premier test nous considérerons l'oscillateur forcé de Van der Pol défini par :

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -x_1 - \mu(x_1^2 - 1)x_2 + d(t) + u \end{aligned} \tag{3.55}$$

avec $d(t) = 2.5\cos 3t$.

Malgré que le système (3.55) paraisse simple, son comportement est très exotique. En effet, pour la valeur $\mu = 0.5$ le système entre en auto-oscillation (cycle limite) comme illustré sur la figure 3.6.a, alors que pour la valeur $\mu = 6$ le système exhibe un comportement chaotique comme illustré sur la figure 3.6.b.

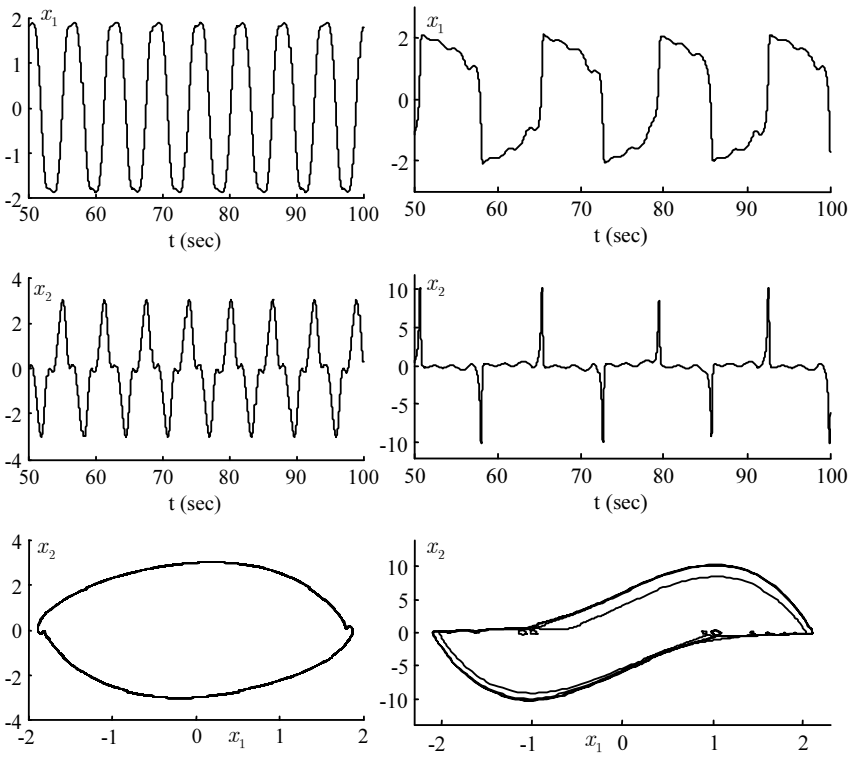


Figure 3.6 : Comportement autonome du système (3.55) ($u(t) = 0$). a) cycle limite pour $\mu = 0.5$, b) chaotique pour $\mu = 6$.

Remarquons que l'on peut écrire

$$-x_1 - \mu(x_1^2 - 1)x_2 = \underline{a}(x)\underline{x}$$

avec $\underline{x}^T = \left[\begin{array}{cc} x_1 & x_2 \end{array}\right]$ et $\underline{a}(x) = \left[\begin{array}{cc} -1 & -\mu(x_1^2 - 1) \end{array}\right]$.

Le modèle de référence est donné par :

$$A_m = \left[\begin{array}{cc} 0 & 1 \\ -4 & -4 \end{array}\right], b_m = 4$$

La loi de commande neuronale est donnée par :

$$u = \underline{\phi}_1(\underline{x})\, K_1\underline{x} + \underline{\phi}_2(\underline{x})\, \underline{k}_2 r + \underline{\phi}_3(\underline{x})\, \underline{k}_3 \tag{3.56}$$

Chaque terme dans (3.56) est réalisé avec un réseau MLP avec 3 neurones dans la couche cachée et des fonctions d'activation tangente hyperbolique. Les entrées pour chaque réseau sont les deux états plus le biais (comme montré sur la figure 3.7).

Les paramètres ajustables des 3 réseaux de neurones sont $\underline{c}_1 \in \Re^{3\times 3}$, $K_1 \in \Re^{3\times 2}$, $\underline{c}_2 \in \Re^{3\times 3}$, $\underline{k}_2 \in \Re^{3\times 1}$ et $\underline{c}_3 \in \Re^{3\times 3}$, $\underline{k}_3 \in \Re^{3\times 1}$, respectivement. Les bornes nécessaires sont fixées à $\varphi_0 = 1$, $\kappa_1 = 7$, $\kappa_2 = 2$, $\kappa_3 = 3$, $\overline{\epsilon}_1 = 0.01$, $\overline{\epsilon}_2 = \overline{\epsilon}_3 = 0.001$, d'où

$\alpha_0 = 0.09$. Les lois d'ajustement (3.27)-(3.29) sont utilisées avec $\gamma_1 = \gamma_2 = \gamma_3 = 500$ et $\gamma_{c1} = \gamma_{c2} = \gamma_{c3} = 0.01$.

La solution de l'inégalité matricielle linéaire (3.43), nous permet d'avoir :

$$P = \begin{bmatrix} 1.5092 & 0.3130224 \\ 0.3130224 & 0.2352081 \end{bmatrix} > 0$$

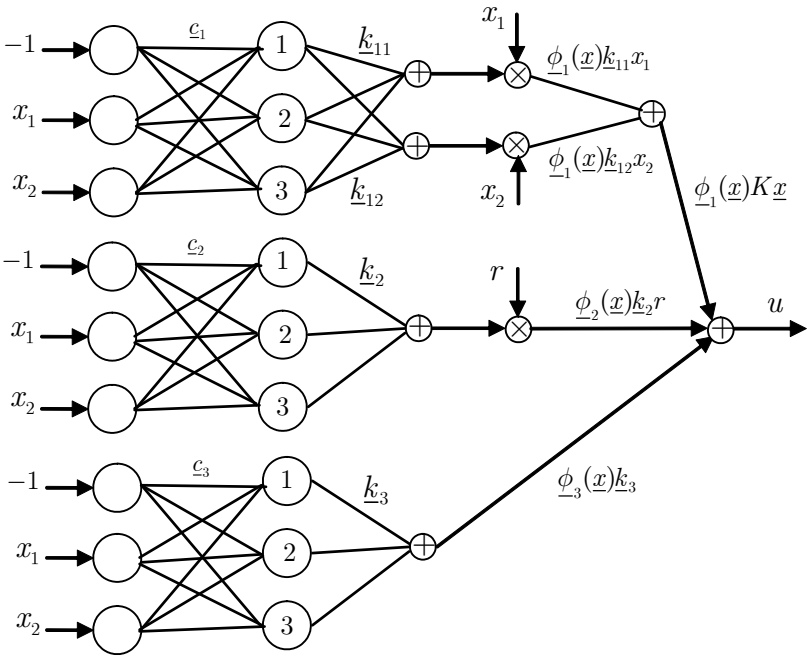


Figure 3.7 : Architecture du contrôleur neuronal à base de réseaux MLP pour l'exemple 1.

La figure 3.8 montre les performances de la commande neuronale pour (a) le comportement oscillatoire et (b) le comportement chaotique. Les simulations montrent qu'après une brève période d'apprentissage, la commande neuronale assure parfaitement le suivi du modèle de référence.

La figure 3.9 illustre les performances de la commande neuronale lors du passage d'un comportement oscillatoire vers un comportement chaotique à $t = 30\,\text{sec}$ (figure 3.9.a), et le passage d'un comportement oscillatoire vers un comportement chaotique à $t = 30\,\text{sec}$ (figure 3.9.b). Les simulations montrent toujours, qu'après un bref régime transitoire, la commande neuronale prend en compte ce changement de dynamique et assure parfaitement le suivi du modèle de référence.

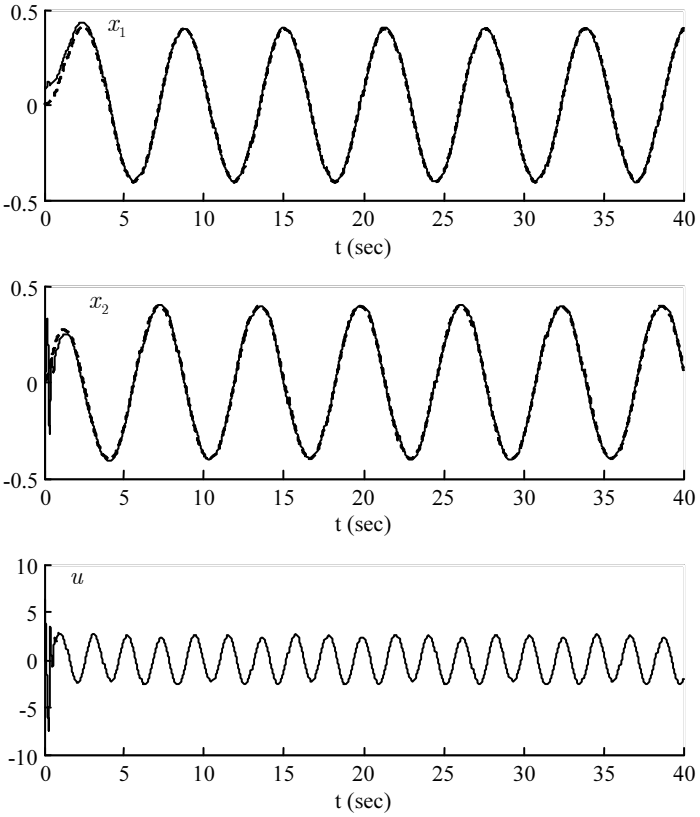


Figure 3.8.a : Performances de la commande MRAC neuronale pour $\mu = 0.5$.

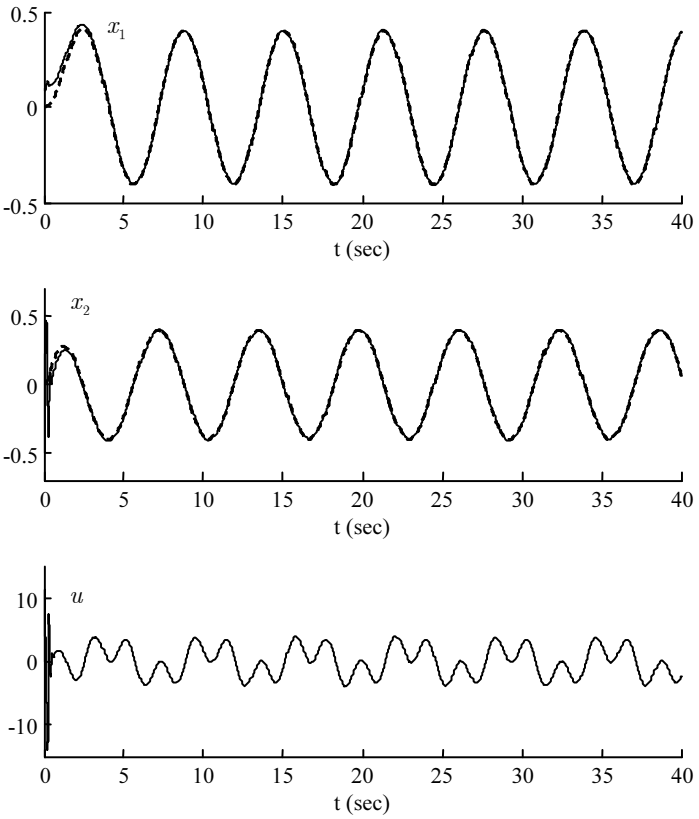


Figure 3.8.b : Performances de la commande MRAC neuronale pour $\mu = 6$.

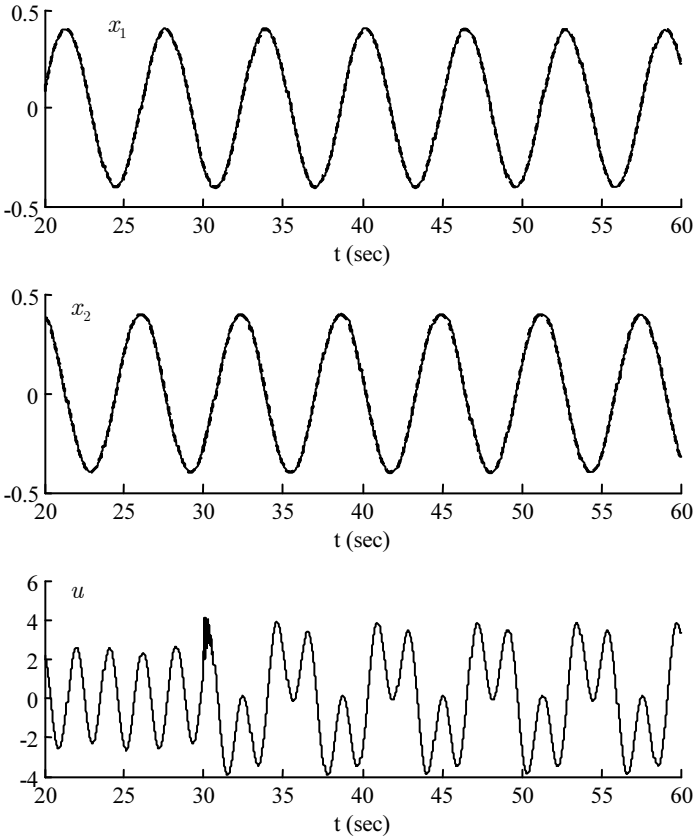


Figure 3.9.a : Performances de la commande MRAC neuronale pour la transition de $\mu = 0.5$ vers $\mu = 6$.

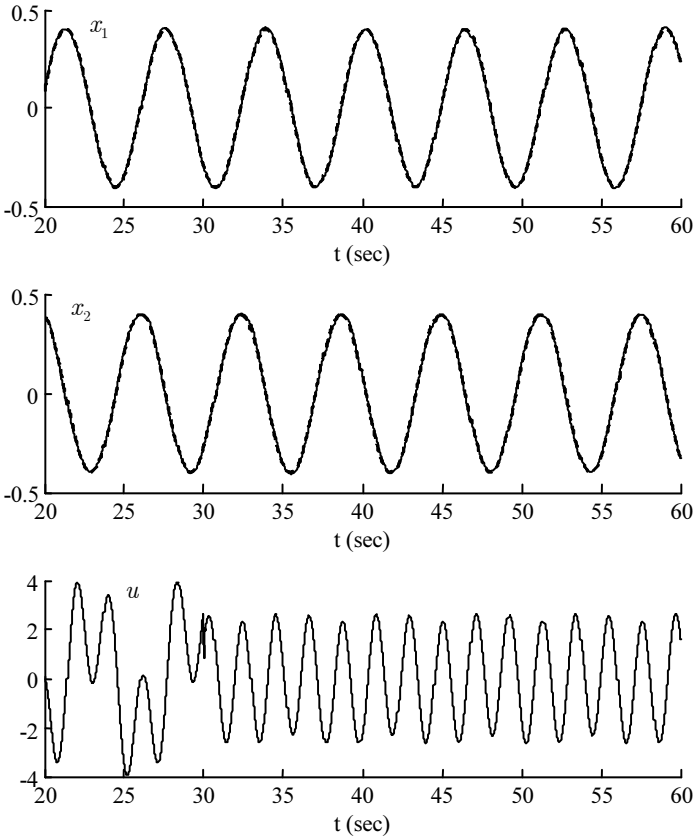


Figure 3.9.b : Performances de la commande MRAC neuronale pour la transition de $\mu = 6$ vers $\mu = 0.5$.

3.6.2 Exemple 2

Pour le deuxième test, nous utiliserons le pendule inversé, dont le modèle est donné par :

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(x_1, x_2) + g(x_1)u + d(x) \end{aligned} \tag{3.57}$$

avec

$$\begin{aligned} f(x_1, x_2) =& \frac{(m_c + m)g \sin x_1 - mlx_2^2 \cos x_1 \sin x_1}{l\left(\frac{4}{3}(m_c + m) - m\cos^2 x_1\right)}, \\ g(x_1) =& \frac{\cos x_1}{l\left(\frac{4}{3}(m_c + m) - m\cos^2 x_1\right)} \end{aligned}$$

où x_1, x_2 et u sont, respectivement, la position angulaire du pendule, la vitesse angulaire et la force d'entrée.

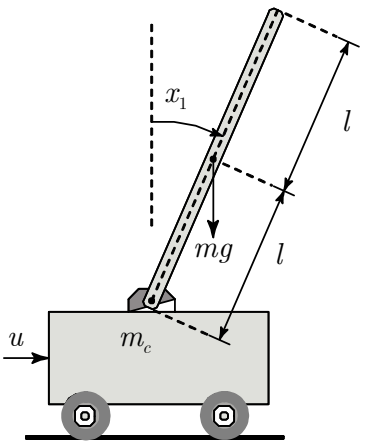


Figure 3.10 : Pendule inversé.

Le modèle (3.57) peut aussi représenter la dynamique d'un satellite, un missile ou une grue.

Remarquons que $f(\underline{x})$ peut être écrite comme $f(\underline{x}) = \underline{a}(\underline{x})\underline{x}$, avec $\underline{x}^T = \begin{bmatrix} x_1 & x_2 \end{bmatrix}$ et

$$a_1(\underline{x}) = \frac{(m_c+m)g}{l\left(\frac{4}{3}(m_c+m) - m\cos^2 x_1\right)} \frac{\sin x_1}{x_1}$$
$$a_2(\underline{x}) = -\frac{mlx_2 \cos x_1 \sin x_1}{l\left(\frac{4}{3}(m_c+m) - m\cos^2 x_1\right)}$$

Pour la simulation, les paramètres nominaux du système sont choisis comme suit : $m_c = 1\text{Kg}$, $m = 0.1\text{Kg}$, $l = 0.5\text{m}$ et $g = 9.8\text{m/s}^2$.

Le modèle de référence est donné par :

$$A_m = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}, b_m = 1$$

L'inspection de la dynamique du modèle de référence montre que les variables de référence sont bornées dans l'intervalle $[-1,1] \times [-2,2]$ (le long de x_1, x_2, respectivement). La loi de commande neuronale est donnée par :

$$u = \underline{\phi}(\underline{x}) K_1 \underline{x} + \underline{\phi}(\underline{x}) \underline{k}_2 r + \underline{\phi}(\underline{x}) \underline{k}_3 \tag{3.58}$$

La loi de commande est construite avec un seul réseau de neurones (figure 3.11) avec neuf fonctions RBF et quatre sorties pour produire les trois termes dans (3.58). Les RBFs sont localisées régulièrement sur l'espace $[-1,1] \times [-2,2]$ avec des pas de 1 et 2 par rapport à x_1 et x_2, et des écarts types de $1/\sqrt{2}$ et $\sqrt{2}$, respectivement.

Les paramètres ajustables du réseau de neurones sont $K_1 \in \Re^{9\times 2}$, $\underline{k}_2 \in \Re^{9\times 1}$ et $\underline{k}_3 \in \Re^{9\times 1}$. Les bornes nécessaires sont fixées à $\varphi_0 = 0.5$, $\kappa_1 = 16$, $\kappa_2 = 2$, $\kappa_3 = 1$, $\overline{\epsilon}_1 = 0.01$, $\overline{\epsilon}_2 = \overline{\epsilon}_3 = 0.001$, d'où $\alpha_0 = 0.2$. Les lois d'ajustement (3.27)-(3.29) sont utilisées avec $\gamma_1 = \gamma_2 = \gamma_3 = 25$.

La solution de l'inégalité matricielle linéaire (3.43), nous permet d'avoir :

$$P = \begin{bmatrix} 1.4497 & 0.7007 \\ 0.7007 & 2.8008 \end{bmatrix} > 0$$

Les performances de poursuite pour le cas des paramètres nominaux ($d(\underline{x}) = 0$) sont montrées sur la figure 3.12. Il est clair que les erreurs de poursuite sont rapidement réduites, et le signal de commande est modéré.

La figure 3.13 montre l'effet de la perturbation externe $d(\underline{x}) = \text{sgn}(x_2) + x_2$ appliquée à l'instant $t = 60\,\text{sec}$. L'effet de cette perturbation est très vite compensé par la commande neuronale.

La figure 3.14 montre l'effet des variations des paramètres. Deux tests sont opérés : le premier consiste à réduire les valeurs des paramètres m_c, m et l de 50% à l'instant $t = 60\,\text{sec}$. On remarque une réduction de l'effort de commande, ce qui est très normal. Le deuxième test consiste à augmenter les valeurs des paramètres m_c, m et l de 50% à l'instant $t = 60\,\text{sec}$. On remarque une augmentation sensible de l'effort de commande pour compenser ce supplément de charge.

La figure 3.15 illustre l'effet de perturbation combinée avec la variation paramétrique. Il est clair, à partir des résultats obtenus, que les effets des incertitudes paramétriques et de la perturbation externe sont rejetés par la commande neuronale, ce qui démontre une grande robustesse de la commande proposée.

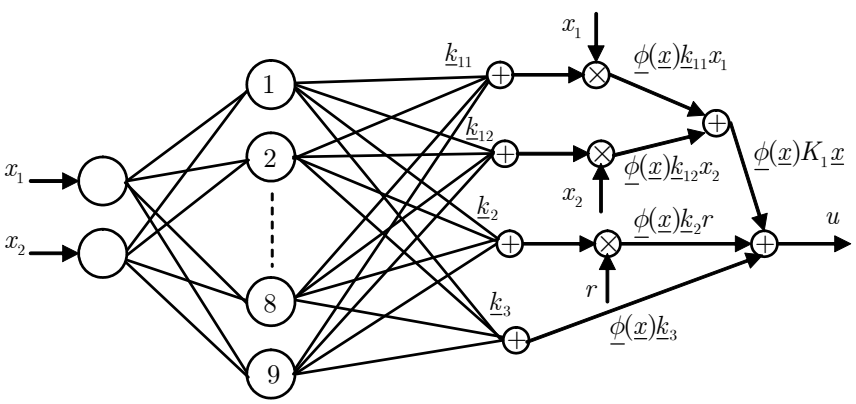


Figure 3.11 : Architecture du contrôleur neuronal à base d'un réseau RBF pour l'exemple 2.

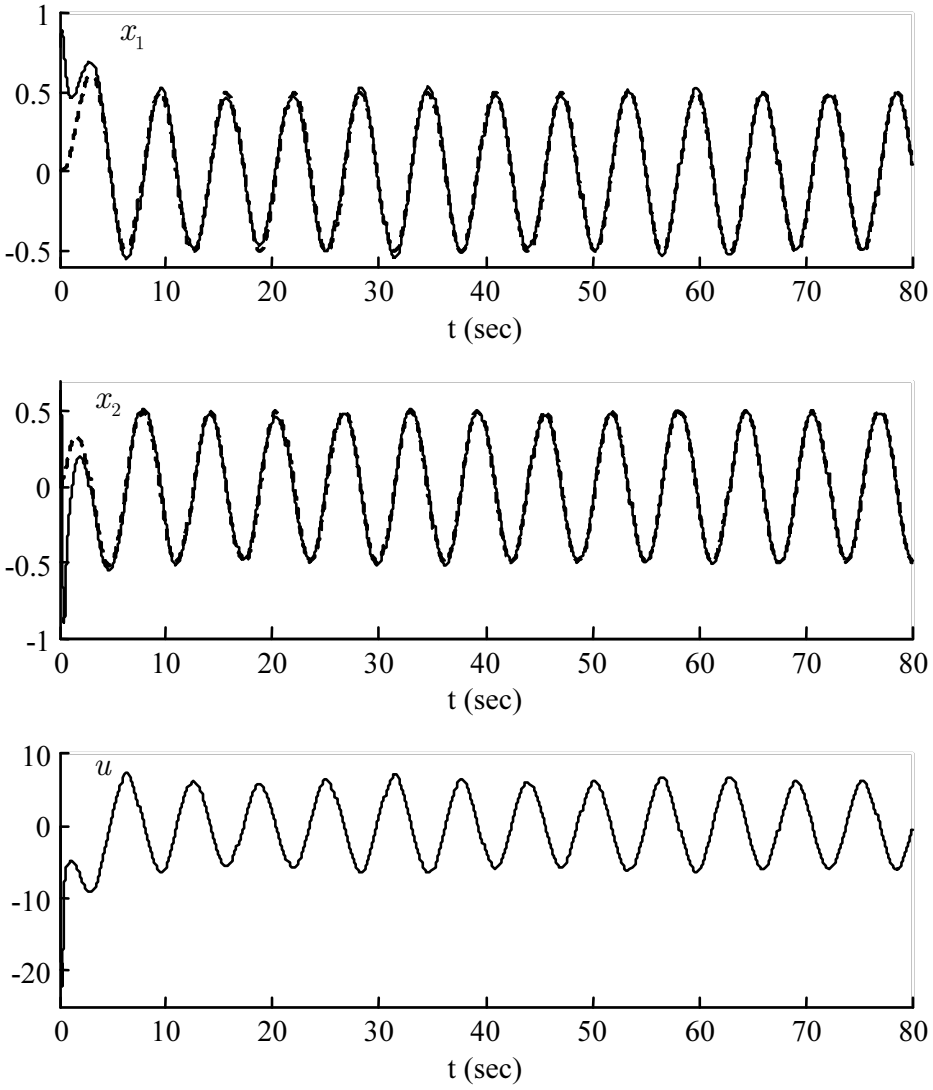


Figure 3.12 : Poursuite avec les paramètres nominaux.

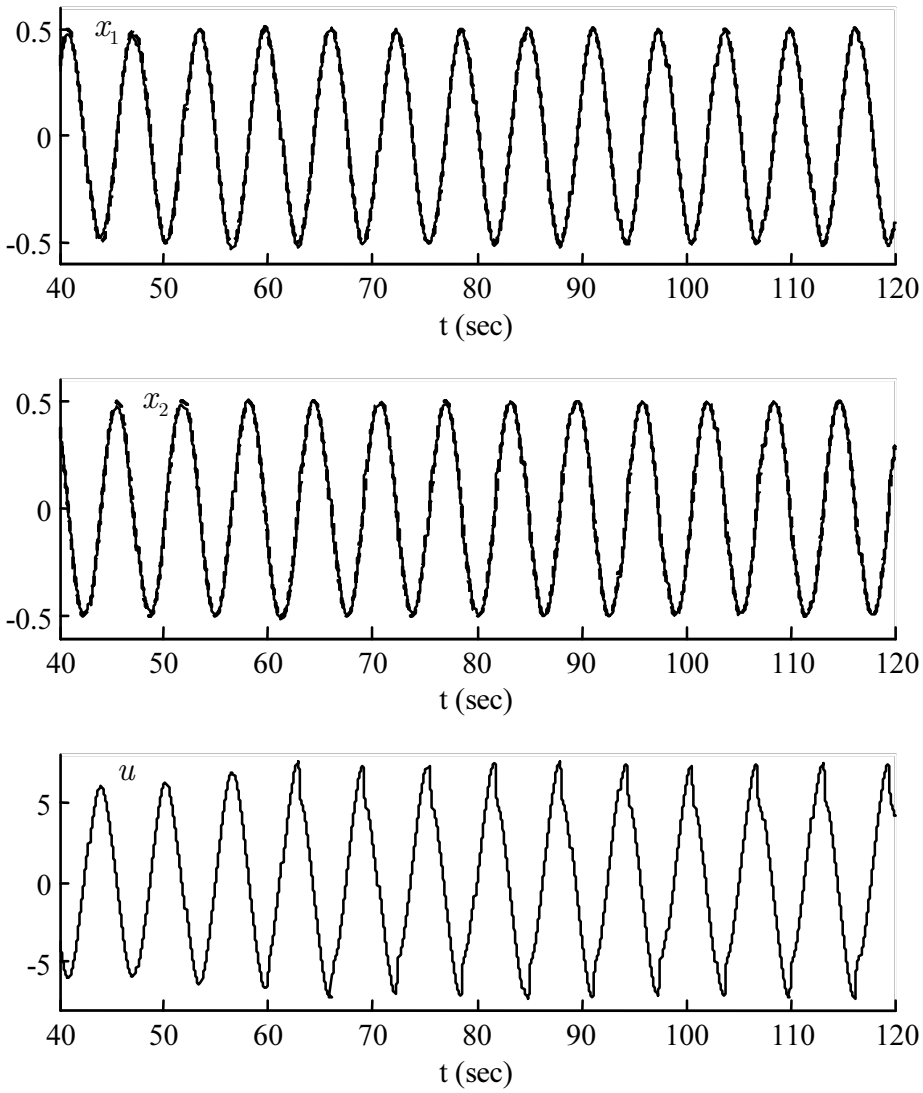


Figure 3.13 : Poursuite avec perturbation externe appliquée à $t = 60\,\text{sec}$.

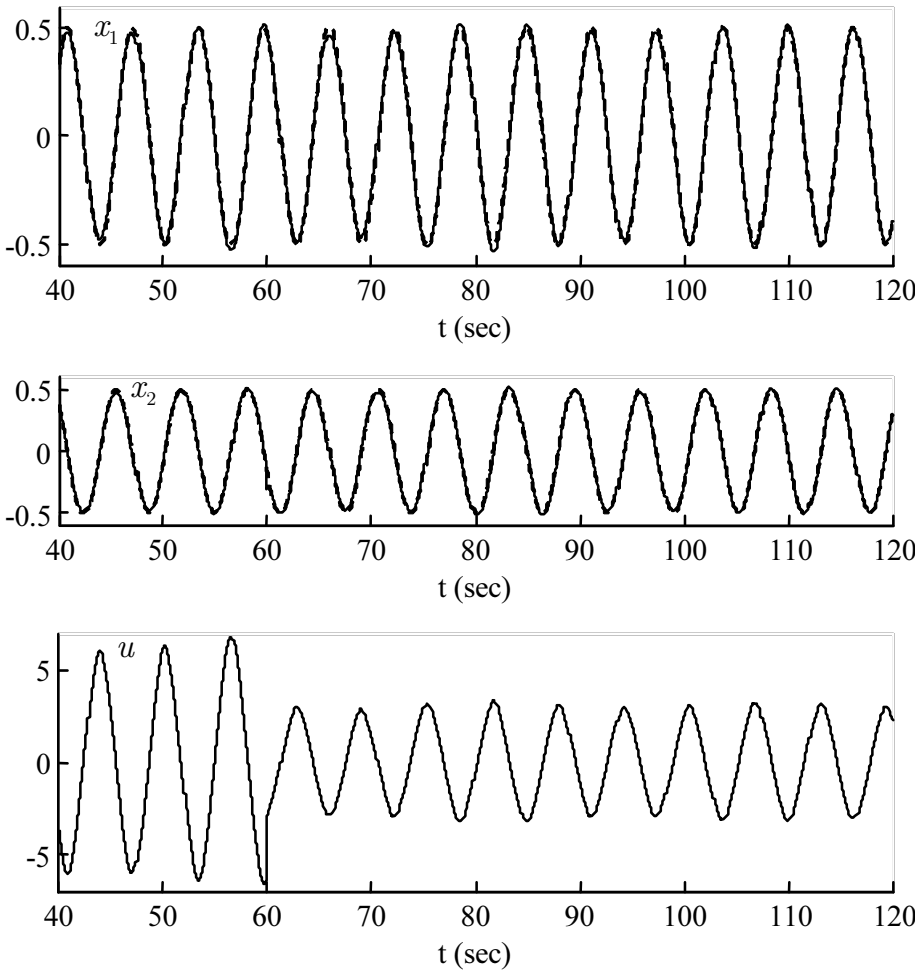


Figure 3.14.a : Poursuite avec paramètres réduits de 50% à $t = 60\,\text{sec}$.

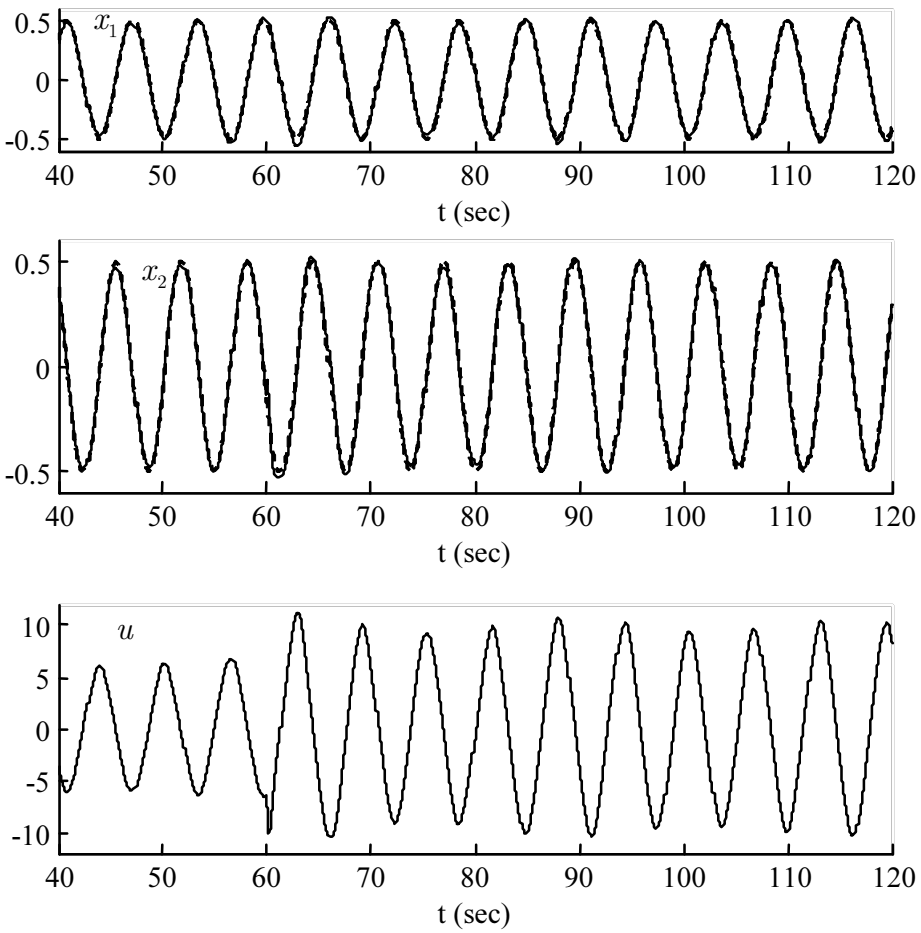


Figure 3.14.b : Poursuite avec paramètres augmentés de 50% à $t = 60\,\text{sec}$.

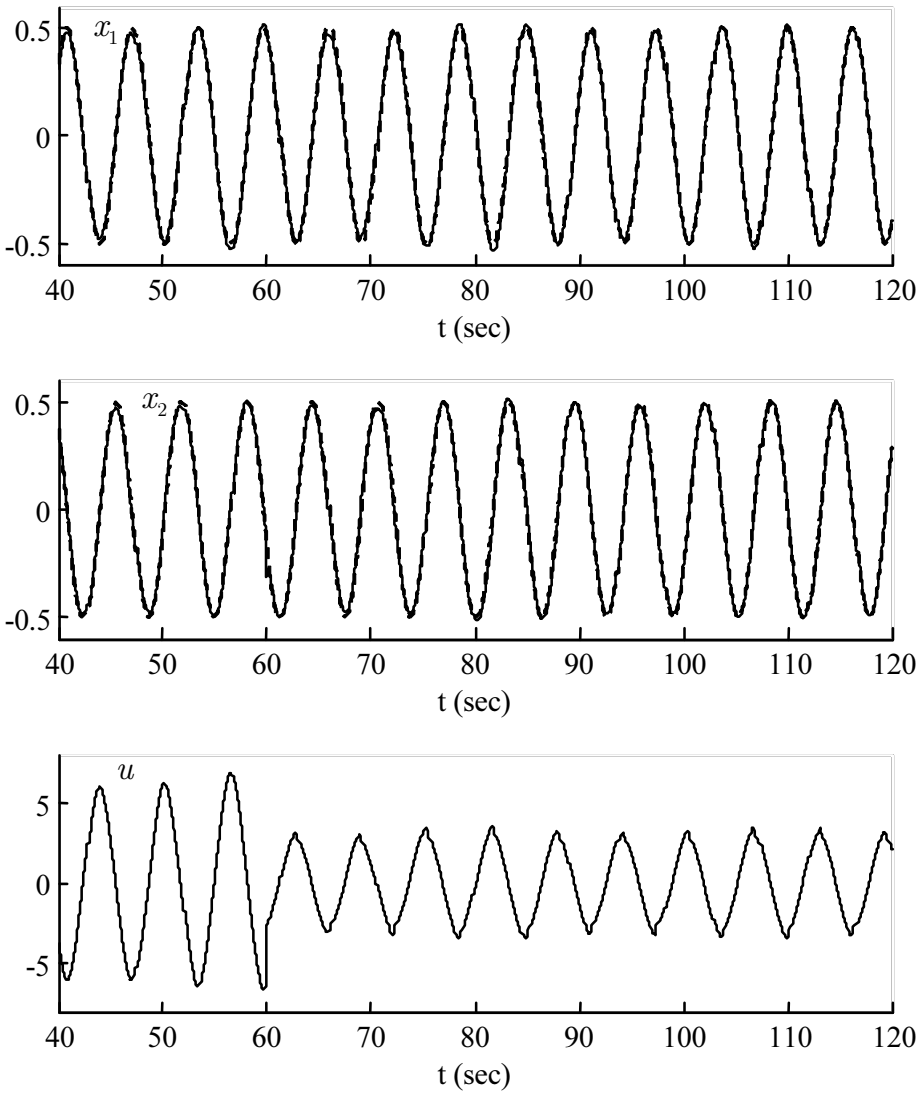


Figure 3.15.a : Poursuite avec perturbation externe et paramètres réduits de 50% à $t = 60\,\text{sec}$.

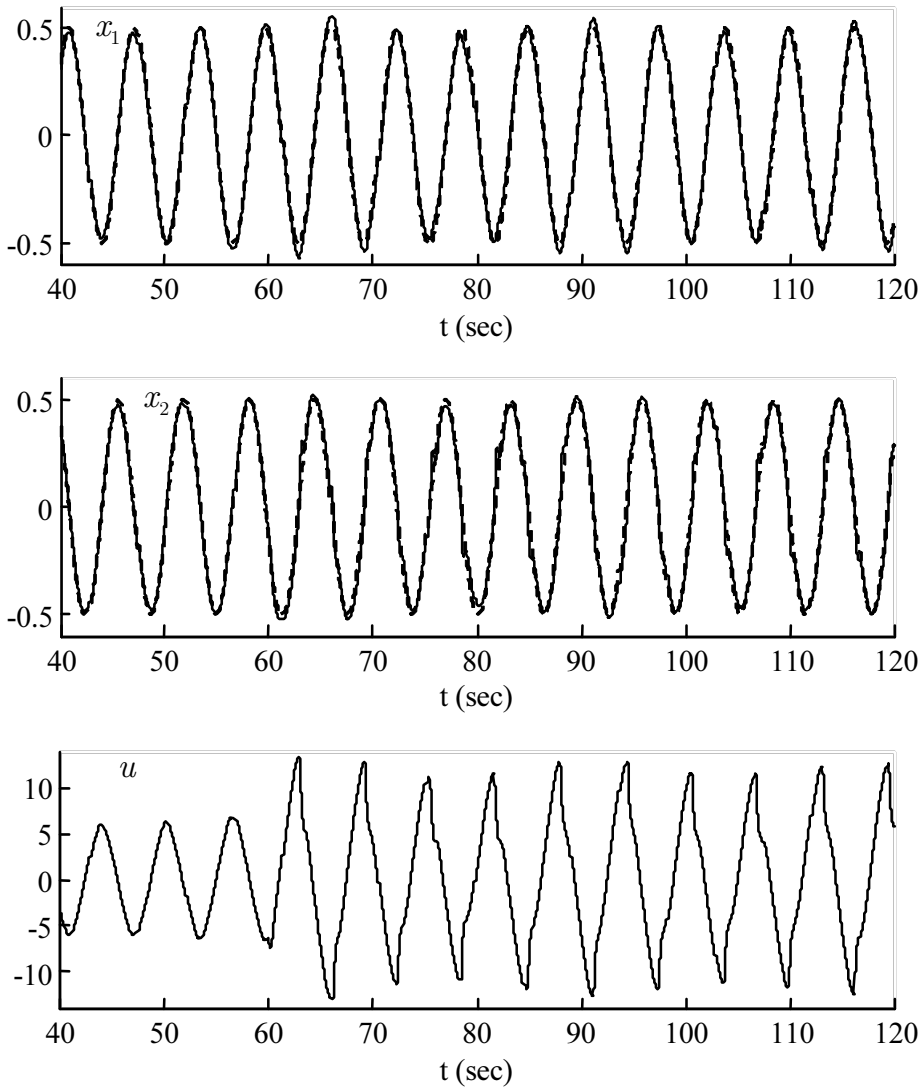


Figure 3.15.b : Poursuite avec perturbation externe et paramètres augmentés de 50% à $t = 60\,\text{sec}$.

3.7 Conclusion

Dans ce chapitre nous avons proposé une approche MRAC neuronale capable de stabiliser et d'améliorer les performances d'une large classe de systèmes non linéaires. Cette approche constitue une extension de la commande MRAC pour la solution du problème de la commande adaptative des systèmes non linéaires. La structure de commande est très simple, et s'adapte à différents types de réseaux de neurones. L'analyse théorique a permis d'établir les conditions de stabilité de cette approche, et le domaine de convergence de l'erreur de poursuite. Les résultats de simulation ont démontré l'efficacité de cette technique pour le contrôle de systèmes incertains et de plus chaotiques ou instables.

Chapitre 4

Commande MRAC neuronale par retour de sortie

4.1 Introduction

Pour éliminer la condition sur la mesure des états du système non linéaire requise dans les développements du chapitre 3, le chapitre 4 étudiera l'utilisation d'un observateur d'erreur pour estimer les états nécessaires à l'implémentation de la commande MRAC neuronale. La section 4.2 donne un aperçu sur le problème d'estimation des états dans les systèmes linéaires et non linéaires. La section 4.3 pose le problème de la commande avec observateur d'état des systèmes non linéaires. La section 4.4 introduit l'observateur d'erreur utilisé pour estimer les états du système, la loi de commande et les lois d'ajustement utilisées. L'analyse de la stabilité de la boucle de commande avec observateur est développée dans la section 4.5. La section 4.6 introduit quelques remarques sur les résultats obtenus. Les tests de simulation sont développés dans la section 4.7. La section 4.8 conclut le chapitre.

4.2 Estimation d'état et observateurs

L'approche de commande neuronale, proposée dans le chapitre précédant, est basée sur l'utilisation du vecteur d'état du système. Ceci suppose que tous les états sont accessibles à la mesure. Cependant, dans de nombreuses applications il peut être très coûteux, voir impossible, d'installer des capteurs physiques afin de mesurer directement certaines quantités. Dans ces cas de figure, une solution alternative est l'utilisation d'un observateur. Un observateur est un algorithme qui reconstitu les états internes non mesurables d'un système à partir des entrées et des sorties mesurables (figure 4.1). Cette technique est appelée aussi retour de sortie (output feedback). Dans le contexte de la commande adaptative, il n'est pas nécessaire que les estimations convergent parfaitement, mais il est essentiel que l'erreur d'estimation soit réduite de sorte que l'erreur de poursuite, qui est l'objectif de la commande, soit acceptable.

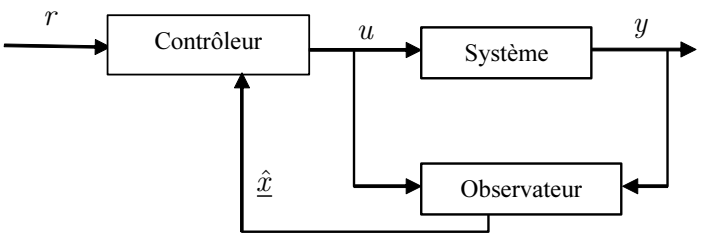


Figure 4.1 : Schéma de principe d'une commande avec observateur.

4.2.1 Observateurs pour les systèmes linéaires

Dans le cas des systèmes linéaires certains (exactement connus), la théorie des observateurs est bien étudiée et les propriétés d'observabilité sont étroitement liée à l'existence d'observateurs avec des propriétés de convergence bien définies [45]. Pour des systèmes linéaires SISO décrits par les équations d'état :

$$\dot{\underline{x}} = A_l \underline{x} + \underline{b}_l u \tag{4.1}$$

$$y = \underline{c}\underline{x} \tag{4.2}$$

La condition pour l'observabilité est que la matrice

$$\left[\begin{array}{cccccc} \underline{c}^T & A_l^T \underline{c}^T & \left(A_l^T\right)^2 \underline{c}^T & \ldots & \left(A_l^T\right)^{n-1} \underline{c}^T \end{array} \right] \in \Re^{n \times n}$$

soit non singulière.

Le principe de séparation nous permet de scinder le problème de commande en deux phases : 1) la conception d'une loi de commande stabilisante $u = -\underline{k}\underline{x}$, faisant comme si les états du système sont accessibles ; 2) conception d'un observateur pour estimer les états non mesurables. Cet observateur peut prendre la forme :

$$\dot{\hat{\underline{x}}} = A_l \hat{\underline{x}} + \underline{b}_l u + \underline{l}\left(y - \underline{c}\hat{\underline{x}}\right) \tag{4.3}$$

où $\hat{\underline{x}}$ est le vecteur d'état estimé et $\underline{l}^T = \left[\begin{array}{cccc} l_1 & l_2 & \ldots & l_n \end{array} \right] \in \Re^n$ est le vecteur gain de l'observateur. Ainsi, la dynamique de l'erreur d'estimation $\tilde{\underline{x}} = \underline{x} - \hat{\underline{x}}$ sera donnée par :

$$\dot{\tilde{\underline{x}}} = \left(A_l - \underline{l}\underline{c}\right) \tilde{\underline{x}} \tag{4.4}$$

Le choix du vecteur $\underline{l}$ permet de produire une erreur d'estimation stable avec la dynamique appropriée.
Pour l'implémentation, on utilisera la loi de commande $u = -\underline{k}\hat{\underline{x}}$, ce qui produit la dynamique en boucle fermée :

$$\dot{\underline{x}} = \left(A_l - \underline{b}_l \underline{k}\right) \underline{x} + \underline{b}_l \underline{k} \tilde{\underline{x}} \tag{4.5}$$

La dynamique en boucle fermée, du système augmenté par l'observateur, est donnée par :

$$\left[\begin{array}{c} \dot{\underline{x}} \\ \dot{\tilde{\underline{x}}} \end{array} \right] = \left[\begin{array}{cc} \left(A_l - \underline{b}_l \underline{k}\right) & \underline{b}_l \underline{k} \\ 0 & \left(A_l - \underline{l}\underline{c}\right) \end{array} \right] \left[\begin{array}{c} \underline{x} \\ \tilde{\underline{x}} \end{array} \right] \tag{4.6}$$

Qui est stable si les dynamiques de commande ($A_l - \underline{b}_l\underline{k}$) et d'estimation ($A_l - \underline{l}\underline{c}$) sont stables. Ceci illustre le principe de séparation.

Si le système linéaire est incertain (incertitude paramétrique), on peut faire recours à des techniques d'observateurs adaptatifs [45].

4.2.2 Observateurs pour les systèmes non linéaires

Du point de vue estimation les systèmes non linéaires posent un défi plus important que celui posé par leur commande. La première difficulté qui surgit, est que le principe de séparation n'est pas, en général, applicable aux systèmes non linéaires, même lorsque le système est parfaitement connu. Ceci revient à dire que la stabilité en boucle fermée, du système augmenté par l'observateur, ne peut être déduite de la stabilité séparée de l'observateur et de la loi de commande. Une combinaison d'une loi de commande et d'un observateur stables peut s'avérer instable. Si de plus le système non linéaire est sujet à des incertitudes non paramétriques alors la solution est plutôt difficile à formuler [2, 4, 46].

Pour les systèmes non linéaires certains de la forme :

$$\dot{\underline{x}} = A\underline{x} + \underline{b}\,[f(y) + g(y)u] \tag{4.7}$$

$$y = \underline{c}\underline{x} \tag{4.8}$$

où les non linéarités dépendent seulement des sorties (c.-à-d. les états mesurables), l'estimation des états non mesurables peut être fournie par l'observateur suivant :

$$\dot{\widehat{\underline{x}}} = A\widehat{\underline{x}} + \underline{b}\,[f(y) + g(y)u] + \underline{l}\,(y - \underline{c}\widehat{\underline{x}}) \tag{4.9}$$

et la dynamique de l'erreur d'estimation sera donnée par :

$$\dot{\widetilde{\underline{x}}} = (A - \underline{l}\underline{c})\,\widetilde{\underline{x}} \tag{4.10}$$

Des versions adaptatives existent pour le cas d'incertitudes paramétriques [2].

Pour les systèmes non linéaires de la forme :

$$\dot{\underline{x}} = A\underline{x} + \underline{b}h(\underline{x}, u) \tag{4.11}$$

$$y = \underline{c}\underline{x} \tag{4.12}$$

avec $h(\underline{x}, u) = f(\underline{x}) + g(\underline{x})u$. Si la fonction non linéaires est Lipchitzienne par rapport au vecteur d'état $\underline{x}$ (c.-à-d., $|h(\underline{x}, u) - h(\widehat{\underline{x}}, u)| \leq c_0\,|\underline{x} - \widehat{\underline{x}}|$, où $c_0 = \max\left|\frac{\partial h(\underline{x},u)}{\partial \underline{x}}\right|$ est la constante de Lipchitz), alors l'observateur d'état peut être décrit par :

$$\dot{\widehat{\underline{x}}} = A\widehat{\underline{x}} + \underline{b}h(\widehat{\underline{x}}, u) + \underline{l}\,(y - \underline{c}\widehat{\underline{x}}) \tag{4.13}$$

et la dynamique de l'erreur d'estimation sera donnée par :

$$\dot{\widetilde{\underline{x}}} = (A - \underline{l}\underline{c})\,\widetilde{\underline{x}} + \underline{b}\,[h(\underline{x}, u) - h(\widehat{\underline{x}}, u)] \tag{4.14}$$

La condition de stabilité de l'observateur peut être démontrée en utilisant la fonction de Lyapunov :

$$V_o = \frac{1}{2}\underline{\widetilde{x}}^T P_o \underline{\widetilde{x}} \tag{4.15}$$

où P_o est une matrice définie positive donnée par la solution de l'équation de Lyapunov :

$$(A - \underline{l}\underline{c})^T P_o + P_o (A - \underline{l}\underline{c}) = -Q_o \tag{4.16}$$

La dérivée de (4.15) le long de l'erreur d'estimation (4.14) produit :

$$\dot{V}_o = -\frac{1}{2}\underline{\widetilde{x}}^T Q_o \underline{\widetilde{x}} + \underline{\widetilde{x}}^T P_o \underline{b} \left[h(\underline{x}, u) - h(\underline{\widehat{x}}, u)\right] \tag{4.17}$$

L'équation (4.17) peut être écrite comme :

$$\dot{V}_o \leq -\frac{1}{2}\lambda_{\min Q_o} |\underline{\widetilde{x}}|^2 + c_0 \lambda_{\max P_o} |\underline{\widetilde{x}}|^2 \tag{4.18}$$

qui est définie négative sous la condition :

$$\frac{\lambda_{\min Q_o}}{\lambda_{\max P_o}} > 2c_0 \tag{4.19}$$

Le cas général, où les fonctions non linéaires $f(\underline{x})$ et $g(\underline{x})$ sont inconnues, nécessite l'utilisation d'approximateurs universels comme les réseaux de neurones. Dans ce cas de figure, l'établissement des conditions de stabilité devient plus délicat, puisque les approximateurs introduisent une incertitude supplémentaire qui est l'erreur d'approximation ; de plus les paramètres des approximateurs sont ajustés en ligne pour améliorer la précision de l'approximation. Enfin, la stabilité doit être vérifiée pour la loi de commande et l'observateur conjointement.

Dans la commande neuronale par retour de sortie, il existe deux approches pour la réalisation de l'estimation des états non mesurables, suivant le type de la commande adaptative envisagée [47-52].

1. Approche indirecte et erreur d'estimation :

   Appelée aussi approche Observateur- Contrôleur. Elle passe par l'introduction d'un observateur de la forme :

   $$\dot{\underline{\widehat{x}}} = A\underline{\widehat{x}} + \underline{b}\left[\widehat{f}(\underline{\widehat{x}}) + \widehat{g}(\underline{\widehat{x}})u\right] + \underline{l}\,(y - \underline{c\widehat{x}}) \tag{4.20}$$

   où $\widehat{f}(\underline{\widehat{x}})$ et $\widehat{g}(\underline{\widehat{x}})$ sont des approximations neuronales de $f(\underline{x})$ et $g(\underline{x})$.

   Du point de vue de l'observateur, le terme de correction $\underline{l}\,(y - \underline{c\widehat{x}})$ est donc un terme bénéfique, mais du point de vue de la commande, il apparaît comme une erreur de dynamique venant perturber le système commandé. Des résultats avec cette technique sont reportés dans [47-49].

2. Approche directe et erreur de poursuite :
   Cette technique part de la connaissance d'une loi de commande stabilisante qui dépend de l'état complet du système. Le bouclage de sortie dynamique a alors pour but de directement reproduire ce bouclage d'état stabilisant. La technique repose sur l'estimation de l'erreur de poursuite $\widehat{\underline{e}} = \underline{x}_m - \widehat{\underline{x}}$ (l'état est estimé d'une manière indirecte). Des résultats avec cette techniques sont reportés dans [50-52].

4.3 Position du problème

Nous considérerons la classe de systèmes non linéaires continus donnée par :

$$\begin{aligned} \dot{\underline{x}} &= A\underline{x} + \underline{b}\left[f(\underline{x}) + g(\underline{x})u + d(\underline{x},t)\right] \\ y &= \underline{cx} \end{aligned} \tag{4.21}$$

où $\underline{x}^T = \left[\begin{array}{cccc} x_1 & x_2 & .. & x_n \end{array}\right] \in \Re^n$ est le vecteur d'état, $u \in \Re$ est l'entrée du système, $y \in \Re$ est la sortie du système, $f(\underline{x})$, $g(\underline{x})$ sont des fonctions non linéaires inconnues mais lisses et $d(\underline{x},t)$ est le terme qui regroupe les incertitudes et les perturbations externes.

Rappelons que le système non linéaire (4.21) satisfait les hypothèses suivantes :

P1. Le gain d'entrée $g(\underline{x})$ est strictement positif dans l'espace de commande $\Omega_{\underline{x}}$. Le cas négatif peut aussi être considéré.

P2. $d(\underline{x},t)$ est bornée.

P3. La fonction non linéaire $f(\underline{x})$ peut être décomposée comme :

$$f(\underline{x}) = \underline{a}(\underline{x})\,\underline{x} + a_0(\underline{x})$$

avec $\underline{a}(\underline{x}) = \left[\begin{array}{cccc} a_1(\underline{x}) & a_2(\underline{x}) & \ldots & a_n(\underline{x}) \end{array}\right] \in \Re^n$ est un vecteur de fonctions non linéaires inconnues, et $a_0(\underline{x})$ est une fonction non linéaire inconnue mais bornée.

Le modèle de référence est défini par les équations d'état :

$$\begin{aligned} \dot{\underline{x}}_m &= A_m\underline{x}_m + \underline{b}b_m r \\ y_m &= \underline{cx}_m \end{aligned} \tag{4.22}$$

La soustraction de (4.21) de (4.22) et l'utilisation de l'hypothèse P3 conduit à la dynamique de l'erreur de poursuite suivante :

$$\begin{aligned} \dot{\underline{e}} &= A_m\underline{e} + \underline{b}\left[-\left(\underline{a}_m + \underline{a}(\underline{x})\right)\underline{x} + b_m r - g(\underline{x})u - a_0(\underline{x}) - d(x,t)\right] \\ e_y &= \underline{ce} \end{aligned} \tag{4.23}$$

avec $e_y = y_m - y$ est l'erreur de poursuite de sortie.

Le problème de commande peut être énoncé comme suit : Concevoir une loi de commande $u(\widehat{\underline{x}})$ tel que les états du système (4.21) suivent ceux du modèle de référence (4.22), sous la condition que tous les signaux en boucle fermée restent bornés, et avec seulement y comme grandeur mesurable. Le problème devra être résolu en utilisant uniquement l'information disponible sur la sortie du système.

4.4 Conception de la commande neuronale

4.4.1 Architecture de commande

Pour surmonter le problème des états non mesurables, nous avons introduit un observateur d'erreur dans la boucle de commande neuronale comme illustrée sur la figure 4.2. La seule information disponible est la sortie du système y. Le contrôleur neuronal est un retour d'état de la même conception proposée dans le chapitre 3, sauf qu'ici le vecteur d'état estimé $\underline{\widehat{x}}$ est utilisé à la place du vecteur d'état réel $\underline{x}$. Du côté algorithme d'ajustement des paramètres, l'erreur de poursuite $\underline{e}$ ne peut être utilisée, puisque inconnue. Pour surmonter ce problème, l'erreur de sortie $e_y = y_m - y$ disponible est utilisée dans les lois d'ajustement. Pour arriver à cette fin une certaine transformation de la dynamique de l'erreur de poursuite est nécessaire. tous ces éléments seront détaillés et analysés dans les sections qui suivront.

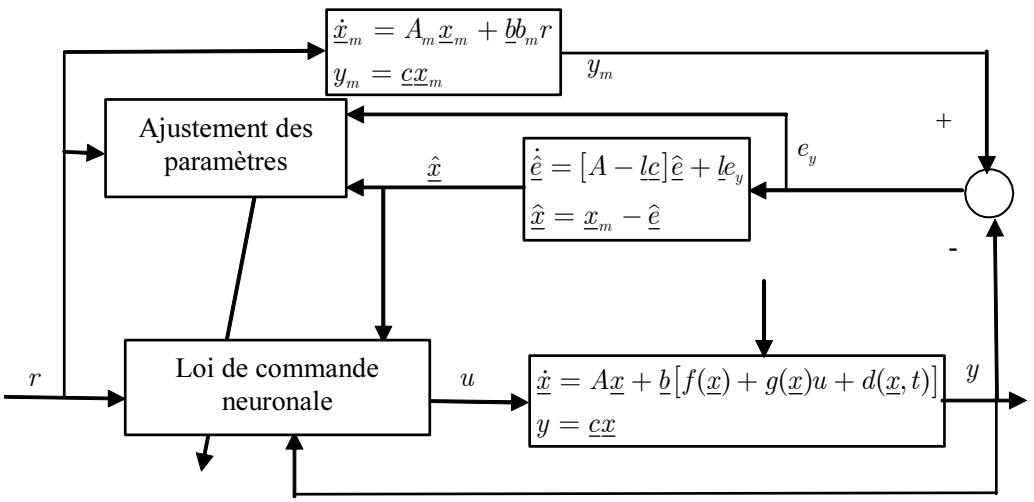


Figure 4.2 : Commande MRAC neuronale avec observateur.

4.4.2 Observateur

L'observateur utilisé est un observateur d'erreur, comme illustré sur la figure 4.2. On commence par estimer l'erreur de poursuite, ensuite l'état du système est estimé à travers une relation algébrique.

Les équations de départ pour l'observateur sont :

$$\begin{aligned} \dot{\underline{\widehat{e}}} &= A\underline{\widehat{e}} + \underline{l}\,(e_y - \widehat{e}_y) \\ \widehat{e}_y &= \underline{c}\underline{\widehat{e}} \end{aligned} \tag{4.24}$$

où $\underline{\widehat{e}} = \underline{x}_m - \underline{\widehat{x}}$ et $\widehat{e}_y = y_m - \widehat{y}$ sont les erreurs de poursuite estimées des états et de la sortie, respectivement. Le vecteur gain $\underline{l}^T = \begin{bmatrix} l_1 & l_2 & \dots & l_n \end{bmatrix} \in \Re^n$ est sélectionné tel que $[A - \underline{l}\underline{c}]$ est une matrice Hurwitz.

Pour l'implémentation pratique, la dynamique de l'observateur est arrangée comme suit :

$$\begin{aligned} \dot{\underline{\widehat{e}}} &= [A - \underline{l}\underline{c}]\,\underline{\widehat{e}} + \underline{l}e_y \\ \underline{\widehat{x}} &= \underline{x}_m - \underline{\widehat{e}} \end{aligned} \tag{4.25}$$

Notons, qu'à partir de l'équation (4.25), il est clair que, comme la matrice $[A - \underline{l}\underline{c}]$ est stable, le comportement de $\underline{\widehat{e}}$ est gouverné par l'erreur de poursuite $\underline{e}$, qui joue le rôle de l'excitation pour l'observateur. Donc, si $\underline{e}$ converge, alors $\underline{\widehat{e}}$ converge aussi.

4.4.3 Loi de commande et Dynamique de l'erreur de poursuite

La loi de commande neuronale est définie par :

$$u = \underline{\phi}_1\left(\underline{\widehat{x}}\right) K_1 \underline{\widehat{x}} + \underline{\phi}_2\left(\underline{\widehat{x}}\right) \underline{k}_2 r + \underline{\phi}_3\left(\underline{\widehat{x}}\right) \underline{k}_3 \tag{4.26}$$

La loi de commande neuronale (4.26) est basée sur les états estimés par l'observateur (4.25) comme montré sur la figure 4.3.

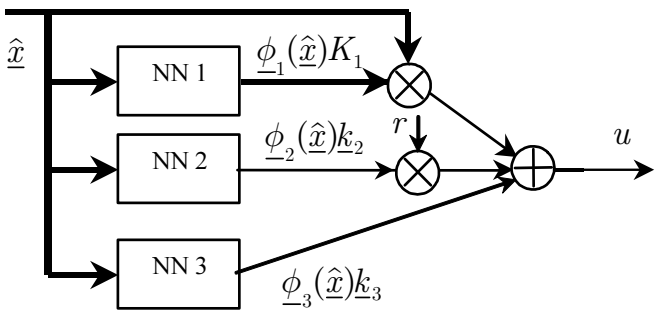


Figure 4.3 : Commande neuronale avec retour de sortie.

L'introduction de la loi de commande (4.26) dans (4.23) donne :

$$\begin{aligned} \dot{\underline{e}} &= A_m \underline{e} + \underline{b}\Big[-\left[\underline{a}_m + \underline{a}(\underline{x})\right]\underline{x} - g(\underline{x})\underline{\phi}_1\left(\underline{\widehat{x}}\right) K_1 \underline{\widehat{x}} - \Big(g(\underline{x})\underline{\phi}_2\left(\underline{\widehat{x}}\right)\underline{k}_2 - b_m\Big) r \\ &\quad - \Big(g(\underline{x})\underline{\phi}_3\left(\underline{\widehat{x}}\right)\underline{k}_3 + a_0\left(\underline{x}\right) + d(x,t)\Big)\Big] \end{aligned} \tag{4.27}$$

En remarquant que $\underline{\widehat{x}} = \underline{x} - \underline{\widehat{e}} + \underline{e}$, on peut développer (4.27) comme :

$$\begin{aligned} \dot{\underline{e}} =& A_m \underline{e} + \underline{b}\Big[-\Big(\left[\underline{a}_m + \underline{a}(\underline{x})\right] + g(\underline{x})\underline{\phi}_1\left(\underline{\widehat{x}}\right) K_1\Big)\underline{x} - \Big(g(\underline{x})\underline{\phi}_2\left(\underline{\widehat{x}}\right)\underline{k}_2 - b_m\Big) r \\ &- \Big(g(\underline{x})\underline{\phi}_3\left(\underline{\widehat{x}}\right)\underline{k}_3 + a_0\left(\underline{x}\right) + d(x,t)\Big) + g(\underline{x})\underline{\phi}_1\left(\underline{\widehat{x}}\right) K_1 \underline{\widehat{e}} - g(\underline{x})\underline{\phi}_1\left(\underline{\widehat{x}}\right) K_1 \underline{e}\Big] \end{aligned} \tag{4.28}$$

A partir des résultats de l'approximation universelle (voir chapitre 2), il existe des paramètres optimaux K_1^*, $\underline{k}_2^*$ et $\underline{k}_3^*$ tel que :

$$\left[\underline{a}_m + \underline{a}(\underline{x})\right] + g(\underline{x})\underline{\phi}_1(\underline{\widehat{x}}) K_1^* = \underline{\epsilon}_1\left(\underline{x}\right) \tag{4.29}$$

$$g(\underline{x})\underline{\phi}_2\left(\underline{\widehat{x}}\right)\underline{k}_2^* - b_m = \epsilon_2\left(\underline{x}\right) \tag{4.30}$$

$$\left(a_0(\underline{x}) + d(\underline{x},t)\right) + g(\underline{x})\underline{\phi}_3(\underline{\widehat{x}})\underline{k}_3^* = \epsilon_3\left(\underline{x}\right) \tag{4.31}$$

où $\underline{\epsilon}_1(\underline{x}) \in \Re^{1\times n}$ et $\epsilon_2(\underline{x}), \epsilon_3(\underline{x}) \in \Re$ sont les erreurs d'approximation idéale pour les trois réseaux de neurones dans (4.26), et les paramètres optimaux K_1^*, $\underline{k}_2^*$ et $\underline{k}_3^*$ sont définis par :

$$
\begin{aligned}
K_1^* &= \arg\min_{K_1\in\Omega_1}\left\{\sup_{\underline{x},\widehat{\underline{x}}\in\Omega_{\underline{x}}}\left|[\underline{a}_m+\underline{a}(\underline{x})]+g(\underline{x})\underline{\phi}_1(\widehat{\underline{x}})K_1^*\right|\right\}\\
\underline{k}_2^* &= \arg\min_{\underline{k}_2\in\Omega_2}\left\{\sup_{\underline{x},\widehat{\underline{x}}\in\Omega_{\underline{x}}}\left|g(\underline{x})\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*-b_m\right|\right\}\\
\underline{k}_3^* &= \arg\min_{\underline{k}_3\in\Omega_3}\left\{\sup_{\underline{x},\widehat{\underline{x}}\in\Omega_{\underline{x}}}\left|(a_0(\underline{x})+d(\underline{x},t))+g(\underline{x})\underline{\phi}_3(\widehat{\underline{x}})\underline{k}_3^*\right|\right\}
\end{aligned}
$$

où Ω_1, Ω_2, Ω_3, $\Omega_{\underline{x}}$ et $\Omega_{\widehat{\underline{x}}}$ sont des ensembles de contraintes pour K_1, $\underline{k}_2$, $\underline{k}_3$, $\underline{x}$ et $\widehat{\underline{x}}$, respectivement.

Ainsi, l'utilisation des propriétés (4.29)-(4.31) dans (4.28) fournie :

$$
\begin{aligned}
\dot{\underline{e}} =& A_m\underline{e}+g(\underline{x})\underline{b}\left[\underline{\phi}_1(\widehat{\underline{x}})\,\widetilde{K}_1\underline{x}+\underline{\phi}_2(\widehat{\underline{x}})\,\widetilde{\underline{k}}_2 r+\underline{\phi}_3(\widehat{\underline{x}})\,\widetilde{\underline{k}}_3\right]\\
&+\underline{b}\left[g(\underline{x})\underline{\phi}_1(\widehat{\underline{x}})\,K_1(\widehat{\underline{e}}-\underline{e})-\underline{\epsilon}_1(\underline{x})\,\underline{x}-\epsilon_2(\underline{x})\,r-\epsilon_3(\underline{x})\right]
\end{aligned}
\tag{4.32}
$$

où $\widetilde{K}_1 = K_1^* - K_1$, $\widetilde{\underline{k}}_2 = \underline{k}_2^* - \underline{k}_2$ et $\widetilde{\underline{k}}_3 = \underline{k}_3^* - \underline{k}_3$ sont les erreurs d'estimation des paramètres des réseaux de neurones.

En utilisant le fait que $\underline{x} = \underline{x}_m - \underline{e}$, (4.32) peut encore être arrangée comme :

$$
\begin{aligned}
\dot{\underline{e}} =& A_m\underline{e}+g(\underline{x})\underline{b}\left[\underline{\phi}_1(\widehat{\underline{x}})\,\widetilde{K}_1\underline{x}_m+\underline{\phi}_2(\widehat{\underline{x}})\,\widetilde{\underline{k}}_2 r+\underline{\phi}_3(\widehat{\underline{x}})\,\widetilde{\underline{k}}_3\right]\\
&+\underline{b}\left[g(\underline{x})\underline{\phi}_1(\widehat{\underline{x}})\,K_1\widehat{\underline{e}}-g(\underline{x})\underline{\phi}_1(\widehat{\underline{x}})\,K_1^*\underline{e}-\underline{\epsilon}_1(\underline{x})\,\underline{x}-\epsilon_2(\underline{x})\,r-\epsilon_3(\underline{x})\right]
\end{aligned}
\tag{4.33}
$$

En exploitant la propriété (4.30), nous avons :

$$
g(\underline{x}) = \frac{b_m+\epsilon_2(\underline{x})}{\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*}
\tag{4.34}
$$

Ce qui permet d'arranger l'équation (4.33) comme :

$$
\begin{aligned}
\dot{\underline{e}} =& A_m\underline{e}+\frac{b_m}{\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*}\underline{b}\left[\underline{\phi}_1(\widehat{\underline{x}})\,\widetilde{K}_1\underline{x}_m+\underline{\phi}_2(\widehat{\underline{x}})\,\widetilde{\underline{k}}_2 r+\underline{\phi}_3(\widehat{\underline{x}})\,\widetilde{\underline{k}}_3\right]\\
&+\underline{b}\left[\frac{\epsilon_2(\underline{x})}{\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*}\underline{\phi}_1(\widehat{\underline{x}})\,\widetilde{K}_1\underline{x}_m+\frac{\epsilon_2(\underline{x})}{\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*}\underline{\phi}_2(\widehat{\underline{x}})\,\widetilde{\underline{k}}_2 r+\frac{\epsilon_2(\underline{x})}{\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*}\underline{\phi}_3(\widehat{\underline{x}})\,\widetilde{\underline{k}}_3\right]\\
&+\underline{b}\left[g(\underline{x})\underline{\phi}_1(\widehat{\underline{x}})\,K_1\widehat{\underline{e}}-g(\underline{x})\underline{\phi}_1(\widehat{\underline{x}})\,K_1^*\underline{e}-\underline{\epsilon}_1(\underline{x})\,\underline{x}_m+\underline{\epsilon}_1(\underline{x})\,\underline{e}-\epsilon_2(\underline{x})\,r-\epsilon_3(\underline{x})\right]
\end{aligned}
\tag{4.35}
$$

Enfin, sous une forme plus compacte, (4.35) est arrangée comme :

$$
\begin{aligned}
\dot{\underline{e}} =& A_m\underline{e}+\frac{b_m}{\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*}\underline{b}\left[\underline{\phi}_1(\widehat{\underline{x}})\,\widetilde{K}_1\underline{x}_m+\underline{\phi}_2(\widehat{\underline{x}})\,\widetilde{\underline{k}}_2 r+\underline{\phi}_3(\widehat{\underline{x}})\,\widetilde{\underline{k}}_3\right]\\
&+\underline{b}\left[\underline{\lambda}_1\underline{e}+\underline{\lambda}_2\widehat{\underline{e}}+\zeta\right]
\end{aligned}
\tag{4.36}
$$

avec

$$\begin{aligned}
\underline{\lambda}_1 &= \underline{\epsilon}_1(\underline{x}) - g(\underline{x})\underline{\phi}_1(\widehat{\underline{x}})K_1^* \\
\underline{\lambda}_2 &= g(\widehat{\underline{x}})\underline{\phi}_1(\widehat{\underline{x}})K_1 \\
\zeta &= \underline{\lambda}_3\underline{x}_m + \lambda_4 r + \lambda_5 \\
\underline{\lambda}_3 &= \frac{\underline{\phi}_1(\widehat{\underline{x}})\widetilde{K}_1}{\underline{\phi}_2(\widehat{\underline{x}})\underline{k}_2^*}\epsilon_2(\underline{x}) - \underline{\epsilon}_1(\underline{x}) \\
\lambda_4 &= -\frac{\underline{\phi}_2(\widehat{\underline{x}})\underline{k}_2}{\underline{\phi}_2(\widehat{\underline{x}})\underline{k}_2^*}\epsilon_2(\underline{x}) \\
\lambda_5 &= \frac{\underline{\phi}_3(\widehat{\underline{x}})\widetilde{\underline{k}}_3}{\underline{\phi}_2(\widehat{\underline{x}})\underline{k}_2^*}\epsilon_2(\underline{x}) - \epsilon_3(\underline{x})
\end{aligned}$$

La dynamique de l'erreur de poursuite (4.36) est gouvernée par plusieurs termes : le premier terme est exponentiellement stable, le deuxième terme dépend des erreurs sur les paramètres du contrôleur adaptatif, et le troisième terme dépend des erreurs d'approximation réalisées par les réseaux de neurones couplées à l'erreur de poursuite et à son estimation ; et enfin, le dernier terme qui est borné.

4.4.4 Transformation SPR

Pour la suite des développements, nous devrons transformer la dynamique de l'erreur de poursuite en une dynamique SPR (réelle strictement positive). L'utilité de cette transformation est d'une importance capitale pour la conception des lois d'ajustement des paramètres. A cette fin, la fonction de transfert de l'erreur de sortie est calculée comme : :

$$\begin{aligned}
e_y = G(s)\frac{b_m}{\underline{\phi}_2(\widehat{\underline{x}})\underline{k}_2^*}\left[\left(\underline{\phi}_1(\widehat{\underline{x}})\widetilde{K}_1\underline{x}_m + \underline{\phi}_2(\widehat{\underline{x}})\widetilde{\underline{k}}_2 r + \underline{\phi}_3(\widehat{\underline{x}})\widetilde{\underline{k}}_3\right)\right] \\
+ G(s)\left[\underline{\lambda}_1\underline{e} + \underline{\lambda}_2\widehat{\underline{e}} + \zeta\right]
\end{aligned} \tag{4.37}$$

avec

$$G(s) = \underline{c}\left[sI_n - A_m\right]^{-1}\underline{b} = \frac{1}{s^n + a_{mn}s^{n-1} + a_{m(n-1)}s^{n-1} + ... + a_{m2}s + a_{m1}} \tag{4.38}$$

A ce point, on définit le polynôme :

$$H(s) = s^{n-1} + \beta_2 s^{n-2} + ... + \beta_{n-2}s + \beta_{n-1} \tag{4.39}$$

qui doit être un polynôme de Hurwitz (stable) et sans facteur commun avec $G(s)$.

Alors, la multiplication et la division de $H(s)$ avec (4.37), produit :

$$\begin{aligned}
e_y \;=\; & G_0(s)\frac{b_{mf}}{\underline{\phi}_2(\widehat{\underline{x}})\underline{k}_2^*}\left[\underline{\phi}_1(\widehat{\underline{x}})\widetilde{K}_1\underline{x}_m + \underline{\phi}_2(\widehat{\underline{x}})\widetilde{\underline{k}}_2 r + \underline{\phi}_3(\widehat{\underline{x}})\widetilde{\underline{k}}_3\right] \\
& + G_0(s)\left[\underline{\lambda}_{1f}\underline{e} + \underline{\lambda}_{2f}\widehat{\underline{e}} + \zeta_f\right]
\end{aligned} \tag{4.40}$$

avec

$$G_0(s) = H(s)G(s) = \frac{s^{n-1} + \beta_2 s^{n-2} + ... + \beta_{n-2}s + \beta_{n-1}}{s^n + a_{mn}s^{n-1} + a_{m(n-1)}s^{n-1} + ... + a_{m2}s + a_{m1}} \tag{4.41}$$

et

$$\begin{aligned} b_{mf} &= H^{-1}(s)\, b_m \\ \underline{\lambda}_{1f} &= H^{-1}(s)\, \underline{\lambda}_1 \\ \underline{\lambda}_{2f} &= H^{-1}(s)\, \underline{\lambda}_2 \\ \zeta_f &= H^{-1}(s)\, \zeta \end{aligned}$$

Pour revenir vers la représentation d'état, remarquons que la fonction de transfert $G_0(s)$ est équivalente à

$$G_0(s) = \underline{c}\,[sI_n - A_m]^{-1}\,\underline{b}_c \tag{4.42}$$

avec

$$\underline{b}_c^T = \left[\begin{array}{ccccccc} 0 & ... & 0 & 1 & \beta_2 & ... & \beta_{n-1} \end{array}\right] \in \Re^n$$

Ainsi, on peut écrire (4.40), sous forme état, comme :

$$\begin{aligned} \dot{\underline{e}} =& A_m \underline{e} + \frac{b_{mf}}{\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*}\underline{b}_c \left[\underline{\phi}_1(\widehat{\underline{x}})\,\widetilde{K}_1 \underline{x}_m + \underline{\phi}_2(\widehat{\underline{x}})\,\widetilde{\underline{k}}_2 r + \underline{\phi}_3(\widehat{\underline{x}})\,\widetilde{\underline{k}}_3\right] \\ &+ \underline{b}_c \left[\underline{\lambda}_{1f}\underline{e} + \underline{\lambda}_{2f}\widehat{\underline{e}} + \zeta_f\right] \\ e_y =& \underline{c}\underline{e} \end{aligned} \tag{4.43}$$

Le théorème et le lemme suivants sont essentiels pour le reste du développement.

Théorème 4.1 : Fonction de transfert SPR [28].

Une fonction de transfert $G_0(s)$ strictement propre est dite SPR si :

1. $G_0(s)$ est stable
2. $\mathrm{Re}\,[G_0(j\omega)] > 0$ pour $\omega \in [-\infty, +\infty]$
3. $\lim\limits_{\omega \to +\infty} \omega^2\, \mathrm{Re}\,[G_0(j\omega)] > 0.$

Dans le plan de Nyquist, le fait que $G_0(s)$ soit SPR est équivalent à ce que son tracé soit entièrement dans le demi-plan droit du plan complexe.

Une conséquence très importante de la propriété SPR est le lemme suivant.

Lemme 4.1 (Kalman-Yakubovich-Popov (KYP) Lemma) [28].

Si la fonction de transfert $G_0(s) = \underline{c}\,[sI_n - A_m]^{-1}\,\underline{b}_c$, avec $(A_m, \underline{b}_c)$ contrôlable et $(A_m, \underline{c})$ observable, est SPR, alors il existe une matrice $P = P^T > 0$ tel que :

$$A_m^T P + P A_m < 0 \tag{4.44}$$

$$P\underline{b}_c = \underline{c}^T \tag{4.45}$$

Ainsi, le fait que $G_0(s)$ soit SPR signifie que P doit satisfaire non seulement l'inégalité de Lyapunov (4.44), mais aussi la contrainte entrée-sortie (4.45), ce qui limite le degré relatif de $G_0(s)$ à une valeur inférieure à 2 et ses zéros doivent être stables.

Il en sort de ces propositions que la dynamique d'erreur (4.43) est SPR pour un choix approprié de $H(s)$, ce qui implique que (4.44)-(4.45) sont vérifiées aussi.

4.4.5 Lois d'ajustement des paramètres

Le contrôleur adaptatif neuronal (4.26) vérifie les hypothèses suivantes :

C1. Les fonctions d'activation du réseau de neurones NN2 $\underline{\phi}_2(\underline{\widehat{x}})$ sont telles que $\left|\dot{\underline{\phi}}_2(\underline{\widehat{x}})\right| < \varphi_0$, où φ_0 est une constante positive connue, et $\underline{\phi}_2(\underline{\widehat{x}})\,\underline{k}_2^* > 0$.

C2. Les fonctions d'activation des réseaux de neurones sont telles que $\left|\underline{\phi}_i(\underline{\widehat{x}})\right| < 1, i = 1,2,3$.

C3. Les majorations suivantes : $\Omega_1 = \{K_1 \,|\, |K_1| < \kappa_1\}$, $\Omega_2 = \{\underline{k}_2 \,|\, |\underline{k}_2| < \kappa_2\}$ et $\Omega_3 = \{\underline{k}_3 \,|\, |\underline{k}_3| < \kappa_3\}$, avec $\kappa_1, \kappa_2, \kappa_3 > 0$, sont données.

C4. Les erreurs d'approximation sont bornées par $|\underline{\epsilon}_1(\underline{x})| \leq \epsilon_1$, $|\epsilon_2(\underline{x})| \leq \epsilon_2$ et $|\epsilon_3(\underline{x})| \leq \epsilon_3$, pour certaines constantes positives ϵ_1, ϵ_2 et ϵ_3.

Comme déjà introduit au chapitre 3, on peut utiliser les algorithmes du gradient avec projection pour assurer la bornitude des paramètres ajustés. A cet effet, les paramètres du contrôleur neuronal sont ajustés en utilisant l'algorithme suivant :

$$\dot{K}_1 = \gamma_1 b_{mf} e_y \left(\underline{\phi}_1^T(\underline{\widehat{x}})\underline{x}_m^T - I_1 \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{\widehat{x}})\underline{x}_m^T\right] \left(\frac{1+|K_1|}{\kappa_1}\right)^2 K_1 \right) \tag{4.46}$$

$$\dot{\underline{k}}_2 = \gamma_2 b_{mf} e_y \left(\underline{\phi}_2^T(\underline{\widehat{x}}) - I_2 \frac{\underline{k}_2 \underline{k}_2^T \underline{\phi}_2^T(\underline{\widehat{x}})}{|\underline{k}_2|^2} \right) r \tag{4.47}$$

$$\dot{\underline{k}}_3 = \gamma_3 b_{mf} e_y \left(\underline{\phi}_3^T(\underline{\widehat{x}}) - I_3 \frac{\underline{k}_3 \underline{k}_3^T \underline{\phi}_3^T(\underline{\widehat{x}})}{|\underline{k}_3|^2} \right) \tag{4.48}$$

où

$$I_1 = \begin{cases} 0, \text{ si } |K_1| < \kappa_1 \text{ ou } (|K_1| = \kappa_1 \text{ et } e_y \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{\widehat{x}})\underline{x}_m^T\right] \leq 0) \\ 1, \text{ si } |K_1| = \kappa_1 \text{ et } e_y \mathrm{tr}\left[K_1^T \underline{\phi}_1^T(\underline{\widehat{x}})\underline{x}_m^T\right] > 0 \end{cases}$$

$$I_2 = \begin{cases} 0, \text{ si } |\underline{k}_2| < \kappa_2 \text{ ou } |\underline{k}_2| = \kappa_2 \text{ et } e_y \underline{\phi}_2(\underline{\widehat{x}})\underline{k}_2 r \leq 0 \\ 1, \text{ si } |\underline{k}_2| = \kappa_2 \text{ et } e_y \underline{\phi}_2(\underline{\widehat{x}})\underline{k}_2 r > 0 \end{cases}$$

$$I_3 = \begin{cases} 0, \text{ si } |\underline{k}_3| < \kappa_3 \text{ ou } |\underline{k}_3| = \kappa_3 \text{ et } e_y \underline{\phi}_3(\underline{\widehat{x}})\underline{k}_3 \leq 0 \\ 1, \text{ si } |\underline{k}_3| = \kappa_3 \text{ et } e_y \underline{\phi}_3(\underline{\widehat{x}})\underline{k}_3 > 0 \end{cases}$$

et $\gamma_1, \gamma_2, \gamma_3 > 0$ sont les pas d'ajustement sélectionnés par le concepteur.

Il peut être vérifié, en utilisant une analyse similaire à celle du chapitre 3, que les lois d'ajustement (4.46)-(4.48) garantissent que : $\|K_1\| \leq \kappa_1$, $|\underline{k}_2| \leq \kappa_2$ et $|\underline{k}_3| \leq \kappa_3$ $\forall t \geq 0$.

4.5 Analyse de Stabilité

Pour plus de commodité, nous allons arranger la dynamique de l'observateur (4.25) et la dynamique de l'erreur de poursuite (4.43) sous la forme de la dynamique augmentée suivante :

$$\begin{aligned}\dot{\underline{v}} &= A_a\underline{v} + \frac{b_{mf}}{\underline{\phi}_2(\widehat{\underline{x}})\,k_2^*}\underline{b}_a\left[\underline{\phi}_1(\widehat{\underline{x}})\,\widetilde{K}_1\underline{x}_m + \underline{\phi}_2(\widehat{\underline{x}})\,\widetilde{\underline{k}}_2 r + \underline{\phi}_3(\widehat{\underline{x}})\,\widetilde{\underline{k}}_3\right] \\ &\quad + \underline{b}_a\left[\underline{\lambda}_f\underline{v} + \zeta_f\right] \\ e_y &= \underline{c}_a\underline{v}\end{aligned} \tag{4.49}$$

avec

$$\begin{aligned}&\underline{v}^T = \left[\begin{array}{cc}\widehat{\underline{e}}^T & \underline{e}^T\end{array}\right], \underline{b}_a^T = \left[\begin{array}{cc}\underline{0}_{1\times n} & \underline{b}_c^T\end{array}\right], \\ &\underline{c}_a = \left[\begin{array}{cc}\underline{0}_{1\times n} & \underline{c}\end{array}\right], \underline{\lambda}_f = \left[\begin{array}{cc}\underline{\lambda}_{2f} & \underline{\lambda}_{1f}\end{array}\right]\end{aligned}$$

et

$$A_a = \left[\begin{array}{cc} A - \underline{l}\underline{c} & \underline{l}\underline{c} \\ 0_{n\times n} & A_m\end{array}\right]$$

Notons que la réalisation d'état (4.49) d'ordre $2n$ est toujours SPR (c.-à-d. $P_a\underline{b}_a = \underline{c}_a^T$), puisqu'on peut facilement démontrer que $\underline{c}_a\left[sI_{2n} - A_a\right]\underline{b}_a = G_0(s)$.

Il faut remarquer que la stabilité de la dynamique (4.49) engendre la stabilité de l'erreur d'estimation et l'erreur de poursuite simultanément.

Pour l'analyse de stabilité, considérons la fonction de Lyapunov :

$$V = \frac{1}{2}\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*\underline{v}^T P_a\underline{v} + \frac{1}{2\gamma_1}\mathrm{tr}\left[\widetilde{K}_1^T\widetilde{K}_1\right] + \frac{1}{2\gamma_2}\widetilde{\underline{k}}_2^T\widetilde{\underline{k}}_2 + \frac{1}{2\gamma_3}\widetilde{\underline{k}}_3^T\widetilde{\underline{k}}_3 \tag{4.50}$$

où $P_a \in \Re^{2n\times 2n}$ est une matrice définie positive à définir par la suite.

La dérivée de (4.50) le long des trajectoires de (4.49) produit :

$$\begin{aligned}\dot{V} &= \frac{1}{2}\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*\underline{v}^T\left[A_a^T P_a + P_a A_a\right]\underline{v} + \frac{1}{2}\dot{\underline{\phi}}_2(\widehat{\underline{x}})\,\underline{k}_2^*\underline{v}^T P_a\underline{v} \\ &\quad + b_{mf}\underline{v}^T P_a\underline{b}_a\left[\underline{\phi}_1(\widehat{\underline{x}})\,\widetilde{K}_1\underline{x}_m + \underline{\phi}_2(\widehat{\underline{x}})\,\widetilde{\underline{k}}_2 r + \underline{\phi}_3(\widehat{\underline{x}})\,\widetilde{\underline{k}}_3\right] \\ &\quad + \underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*\underline{v}^T P_a\underline{b}_a\left[\underline{\lambda}_f\underline{v} + \zeta_f\right] + \frac{1}{\gamma_1}\mathrm{tr}\left[\widetilde{K}_1^T\dot{\widetilde{K}}_1\right] + \frac{1}{\gamma_2}\widetilde{\underline{k}}_2^T\dot{\widetilde{\underline{k}}}_2 + \frac{1}{\gamma_3}\widetilde{\underline{k}}_3^T\dot{\widetilde{\underline{k}}}_3\end{aligned} \tag{4.51}$$

Alors, en exploitant la relation $\underline{v}^T P_a\underline{b}_a = e_y$, (4.51) peut être arrangée comme :

$$\begin{aligned}\dot{V} &= \frac{1}{2}\underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*\underline{v}^T\left[A_a^T P_a + P_a A_a\right]\underline{v} + \frac{1}{2}\dot{\underline{\phi}}_2(\widehat{\underline{x}})\,\underline{k}_2^*\underline{v}^T P_a\underline{v} \\ &\quad + \underline{\phi}_2(\widehat{\underline{x}})\,\underline{k}_2^*\underline{v}^T P_a\underline{b}_a\left[\underline{\lambda}_f\underline{v} + \zeta_f\right] + \frac{1}{\gamma_1}\mathrm{tr}\left[\widetilde{K}_1^T\left(\gamma_1 b_{mf}e_y\underline{\phi}_1^T(\widehat{\underline{x}})\,\underline{x}_m^T - \dot{K}_1\right)\right] \\ &\quad + \frac{1}{\gamma_2}\widetilde{\underline{k}}_2^T\left(\gamma_2 b_{mf}e_y\underline{\phi}_2^T(\widehat{\underline{x}})\,r - \dot{\underline{k}}_2\right) + \frac{1}{\gamma_3}\widetilde{\underline{k}}_3^T\left(\gamma_3 b_{mf}e_y\underline{\phi}_3^T(\widehat{\underline{x}}) - \dot{\underline{k}}_3\right)\end{aligned} \tag{4.52}$$

Ensuite, en utilisant les lois d'ajustement (4.46)-(4.48) avec une analyse similaire à celle de la section 3.4 du chapitre 3, il peut être montré que les trois derniers termes dans (4.52) sont toujours ≤ 0. Ainsi, (4.52) devient :

$$\begin{aligned}\dot{V} \leq & \frac{1}{2}\underline{\phi}_2\left(\widehat{\underline{x}}\right)\underline{k}_2^*\underline{v}^T\left[A_a^TP_a + P_aA_a\right]\underline{v} + \frac{1}{2}\dot{\underline{\phi}}_2\left(\widehat{\underline{x}}\right)\underline{k}_2^*\underline{v}^TP_a\underline{v} \\ & + \underline{\phi}_2\left(\widehat{\underline{x}}\right)\underline{k}_2^*\underline{v}^TP_a\underline{b}_a\underline{\lambda}_f\underline{v} + \underline{\phi}_2\left(\widehat{\underline{x}}\right)\underline{k}_2^*\underline{v}^TP_a\underline{b}_a\zeta_f \end{aligned} \tag{4.53}$$

Sous l'hypothèse de bornitude énoncée auparavant, nous avons : $\left|\underline{b}_a\underline{\lambda}_f\right| \leq \lambda_0$, où λ_0 est une constante positive. Alors l'expression (4.53) peut être bornée par :

$$\begin{aligned}\dot{V} \leq & \frac{1}{2}\kappa_2\underline{v}^T\left[A_a^TP_a + P_aA_a\right]\underline{v} + \frac{1}{2}\varphi_0\kappa_2\underline{v}^TP_a\underline{v} + \kappa_2\lambda_0\underline{v}^TP_a\underline{v} \\ & + \underline{\phi}_2\left(\widehat{\underline{x}}\right)\underline{k}_2^*\underline{v}^TP_a\underline{b}_a\zeta_f \end{aligned} \tag{4.54}$$

Sous forme plus compacte, (4.54) peut être arrangée comme :

$$\dot{V} \leq \frac{1}{2}\kappa_2\underline{v}^T\left(A_a^TP_a + P_aA_a + \varphi_0P_a + 2\lambda_0P_a\right)\underline{v} + \underline{\phi}_2\left(\widehat{\underline{x}}\right)\underline{k}_2^*\underline{v}^TP_a\underline{b}_a\zeta_f \tag{4.55}$$

Maintenant, si la matrice P_a est choisie telle que :

$$A_a^TP_a + P_aA_a + \left(\varphi_0 + 2\lambda_0\right)P_a \leq -Q_a \tag{4.56}$$

Pour une certaine matrice définie positive $Q_a \in \Re^{2n\times 2n}$, alors

$$\dot{V} \leq -\frac{1}{2}\kappa_2\underline{v}^TQ_a\underline{v} + \underline{\phi}_2\left(\widehat{\underline{x}}\right)\underline{k}_2^*\underline{v}^TP_a\underline{b}_a\zeta_f \tag{4.57}$$

Enfin, (4.57) peut être bornée par :

$$\dot{V} \leq -\frac{1}{2}\kappa_2\lambda_{\min Q_a}\left|\underline{v}\right|^2 + \kappa_2\left|P_a\underline{b}_a\right|\left|\underline{v}\right|\left|\zeta_f\right| \tag{4.58}$$

où $\lambda_{\min Q_a}$ est la plus petite valeur propre de Q_a.

Alors, $\dot{V} \leq 0$ en dehors d'une région bornée définie par :

$$\left|\underline{v}\right| \leq \frac{2\left|P_a\underline{b}_a\right|}{\lambda_{\min Q_a}}\left|\zeta_f\right| \tag{4.59}$$

Les propriétés de l'approche neuronale adaptative basée sur un observateur sont résumées par le théorème suivant.

Théorème 4.2 : Le système de commande composé par le système non linéaire (4.21) satisfaisant P1-P3, le modèle de référence (4.22), le contrôleur neuronal (4.26) satisfaisant C1-C4, l'observateur d'état (4.25), et les lois d'ajustement (4.46)-(4.48), garantit ce qui suit :

1.

$$\left|\widehat{\underline{x}}\right| \leq \left|\underline{x}_m\right| + c_5\left|\zeta_f\right| \tag{4.60}$$

$$\left|\underline{x}\right| \leq \left|\underline{x}_m\right| + c_6\left|\zeta_f\right| \tag{4.61}$$

2.

$$|u| \leq \kappa_1 \left|\underline{x}_m\right| + \kappa_2 |r| + \kappa_3 + c_7 \left|\zeta_f\right| \tag{4.62}$$

3.

$$\int_0^T |\widehat{\underline{e}}|^2 \, dt \leq c_8 + c_9 \int_0^T \left|\zeta_f\right|^2 dt \tag{4.63}$$

$$\int_0^T |\underline{e}|^2 \, dt \leq c_{10} + c_{11} \int_0^T \left|\zeta_f\right|^2 dt \tag{4.64}$$

Preuve :

1. Nous définissons la fonction de Lyapunov suivante

$$V_o = \frac{1}{2} \widehat{\underline{e}}^T P_o \widehat{\underline{e}} \tag{4.65}$$

où P_o est la solution, pour une certaine matrice définie positive Q_o, de l'équation de Lyapunov :

$$\left[A - \underline{lc}\right]^T P_o + P_o \left[A - \underline{lc}\right] = -Q_o \tag{4.66}$$

La dérivée de (4.65) le long de (4.25) produit :

$$\dot{V}_o = -\frac{1}{2} \widehat{\underline{e}}^T Q_o \widehat{\underline{e}} + \widehat{\underline{e}}^T P_o \underline{lce} \tag{4.67}$$

qui est bornée par :

$$\dot{V}_o \leq -\frac{1}{2} \lambda_{\min Q_o} |\widehat{\underline{e}}|^2 + |P_o \underline{lc}| \, |\underline{e}| \, |\widehat{\underline{e}}| \tag{4.68}$$

où $\lambda_{\min Q_o}$ est la plus petite valeur propre de Q_o.

Alors, $\dot{V}_o \leq 0$ si $|\widehat{\underline{e}}|$ est en dehors d'une région définie par :

$$|\widehat{\underline{e}}| \leq 2 \frac{|P_o \underline{lc}|}{\lambda_{\min Q_o}} |\underline{e}| \tag{4.69}$$

D'autre part, nous avons $|\underline{v}| = \sqrt{|\underline{e}|^2 + |\widehat{\underline{e}}|^2}$, qui combinée avec (4.69) donne :

$$|\underline{e}| \leq \frac{1}{\sqrt{1 + 4\left(\frac{|P_o \underline{lc}|}{\lambda_{\min Q_o}}\right)^2}} |\underline{v}| \tag{4.70}$$

Aussi, l'utilisation de (4.70) et (4.59) donne la borne supérieure suivante :

$$|\underline{e}| \leq \frac{2 \left|P_a \underline{b}_a\right|}{\lambda_{\min Q_a} \sqrt{1 + 4\left(\frac{|P_o \underline{lc}|}{\lambda_{\min Q_o}}\right)^2}} \left|\zeta_f\right| \tag{4.71}$$

Alors, en introduisant (4.71), (4.69) devient :

$$|\underline{\widehat{e}}| \leq \frac{4\,|P_o\underline{lc}|\,|P_a\underline{b}_a|}{\lambda_{\min Q_o}\lambda_{\min Q_a}\sqrt{1+4\left(\frac{|P_o\underline{lc}|}{\lambda_{\min Q_o}}\right)^2}}\left|\zeta_f\right| \tag{4.72}$$

A partir de (4.72) et avec le fait que $|\underline{\widehat{x}}| \leq |\underline{x}_m| + |\underline{\widehat{e}}|$, nous obtenons :

$$|\underline{\widehat{x}}| \leq |\underline{x}_m| + \frac{4\,|P_o\underline{lc}|\,|P_a\underline{b}_a|}{\lambda_{\min Q_o}\lambda_{\min Q_a}\sqrt{1+4\left(\frac{|P_o\underline{lc}|}{\lambda_{\min Q_o}}\right)^2}}\left|\zeta_f\right| \tag{4.73}$$

Ce qui donne (4.60) avec :

$$c_5 = \frac{4\,|P_o\underline{lc}|\,|P_a\underline{b}_a|}{\lambda_{\min Q_o}\lambda_{\min Q_a}\sqrt{1+4\left(\frac{|P_o\underline{lc}|}{\lambda_{\min Q_o}}\right)^2}}$$

Maintenant, en utilisant le fait que $|\underline{x}| \leq |\underline{e}| + |\underline{x}_m|$ avec (4.71) nous aurons :

$$|\underline{x}| \leq |\underline{x}_m| + \frac{2\,|P_a\underline{b}_a|}{\lambda_{\min Q_a}\sqrt{1+4\left(\frac{|P_o\underline{lc}|}{\lambda_{\min Q_o}}\right)^2}}\left|\zeta_f\right| \tag{4.74}$$

Alors (4.61) est vérifiée avec :

$$c_6 = \frac{2\,|P_a\underline{b}_a|}{\lambda_{\min Q_a}\sqrt{1+4\left(\frac{|P_o\underline{lc}|}{\lambda_{\min Q_o}}\right)^2}}$$

2. L'utilisation de (4.73) et le fait que

   $$|u| \leq \kappa_1\,|\underline{\widehat{x}}| + \kappa_2\,|r| + \kappa_3 \tag{4.75}$$

   donne :

   $$|u| \leq \kappa_1\,|\underline{x}_m| + \kappa_2\,|r| + \kappa_3 + \kappa_1 c_5\left|\zeta_f\right| \tag{4.76}$$

   Ainsi (4.62) est prouvée avec $c_7 = \kappa_1 c_5$.
3. L'introduction de (4.71) et (4.72) dans (4.68) produit :

   $$\dot{V}_o \leq -\frac{1}{2}\lambda_{\min Q_o}\,|\underline{\widehat{e}}|^2 + \frac{1}{2}\lambda_{\min Q_o} c_5^2\left|\zeta_f\right|^2 \tag{4.77}$$

   Alors, en intégrant les deux membres de (4.77) nous aurons :

   $$\int_0^T |\underline{\widehat{e}}|^2\,dt \leq \frac{2}{\lambda_{\min Q_o}}V_o(0) + c_5^2\int_0^T \left|\zeta_f\right|^2 dt \tag{4.78}$$

   ce qui donne (4.63) avec $c_8 = \frac{2}{\lambda_{\min Q_o}}V_o(0)$ et $c_9 = c_5^2$.

Pour prouver (4.64), nous introduisons (4.71) dans (4.58), ce qui produit :

$$\dot{V} \leq -\frac{1}{2}\kappa_2 \lambda_{\min Q_a} |\underline{v}|^2 + \frac{2\,|P_a \underline{b}_a|}{\lambda_{\min Q_a}} |\zeta_f|^2 \tag{4.79}$$

L'intégration des deux côtés de (4.79) fournit :

$$\int_0^T |\underline{v}|^2 \, dt \leq \frac{2}{\kappa_2 \lambda_{\min Q_a}} V(0) + \frac{4\,|P_a \underline{b}_a|}{\kappa_2 \lambda^2_{\min Q_a}} \int_0^T |\zeta_f|^2 \, dt$$

Alors, en utilisant (4.70) nous obtenons :

$$\begin{aligned} \int_0^T |\underline{e}|^2 \, dt \quad \leq \quad & \frac{2}{\kappa_2 \lambda_{\min Q_a} \sqrt{1 + 4\left(\frac{|P_o \underline{lc}|}{\lambda_{\min Q_o}}\right)^2}} V(0) \\ & + \frac{4\,|P_a \underline{b}_a|}{\kappa_2 \lambda^2_{\min Q_a} \sqrt{1 + 4\left(\frac{|P_o \underline{lc}|}{\lambda_{\min Q_o}}\right)^2}} \int_0^T |\zeta_f|^2 \, dt \end{aligned} \tag{4.80}$$

ce qui produit (4.64), avec :

$$c_{10} = \frac{2\,|V(0)|}{\kappa_2 \lambda_{\min Q_a} \sqrt{1 + 4\left(\frac{|P_o \underline{lc}|}{\lambda_{\min Q_o}}\right)^2}}$$

et

$$c_{11} = \frac{4\,|P_a \underline{b}_a|}{\kappa_2 \lambda^2_{\min Q_a} \sqrt{1 + 4\left(\frac{|P_o \underline{lc}|}{\lambda_{\min Q_o}}\right)^2}}$$

□

4.6 Remarques

1. A partir de (4.63)-(4.64), si $\zeta_f \in L_2$, alors par le lemme de Barbalat [28], les erreurs de poursuite et d'estimation convergent vers zéro.
2. Les résultats 1-3, du théorème 4.2, précisent les faits suivants : d'abord les erreurs de poursuite et d'estimation sont bornées et leurs bornes dépendent de quelques paramètres de conception qui peuvent être ajustés par le concepteur. En second lieu, l'entrée de commande est également bornée et la quantité d'énergie à fournir peut être contrôlée par les paramètres de conception.

3. La satisfaction des conditions de stabilité (4.44) - (4.45) et (4.56), peut être exprimée comme un problème LMI de la forme [53] :

$$P_a > 0 \tag{4.81}$$

$$\begin{bmatrix} -A_a^T P_a - P_a A_a - (\varphi_0 + 2\lambda_0) P_a & P_a \underline{b}_a - \underline{c}_a^T \\ \underline{b}_a^T P_a - \underline{c}_a & 0 \end{bmatrix} > 0 \tag{4.82}$$

La faisabilité de (4.81)-(4.82) dépend du choix du modèle de référence (4.22), du filtre $H(s)$ et de la dynamique de l'observateur (4.25). Les valeurs propres du modèle de référence et de l'observateur influencent la grandeur des valeurs propres de la matrice P_a, qui à son tour influe sur la positivité de (4.82). D'un autre côté, l'augmentation de la norme de $H(s)$ réduit la norme des termes incertains $\underline{\lambda}_f$ ce qui réduit la grandeur de λ_0, mais augmente la dynamique du filtrage.

4.7 Résultats de simulation

La validité de la commande adaptative neuronale par retour de sortie est testée sur les deux exemples étudiés au chapitre précédent.

4.7.1 Exemple 1

La dynamique de l'oscillateur forcé de Vander Pol est réécrite comme :

$$\dot{\underline{x}} = A\underline{x} + \underline{b}\left[-x_1 - \mu(x_1^2 - 1)x_2 + u + d(t)\right]$$

avec

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \underline{b} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Le modèle de référence est donné par :

$$A_m = \begin{bmatrix} 0 & 1 \\ -4 & -4 \end{bmatrix}, b_m = 4$$

Si x_1 est la variable mesurée, alors, l'observateur d'état prendra la forme :

$$\dot{\widehat{\underline{e}}} = [A - \underline{l}\underline{c}]\,\widehat{\underline{e}} + l e_1$$
$$\widehat{\underline{x}} = \underline{x}_m - \widehat{\underline{e}}$$

avec

$$\underline{e} = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} x_{1m} - x_1 \\ x_{2m} - x_2 \end{bmatrix}, \widehat{\underline{e}} = \begin{bmatrix} x_{1m} - \widehat{x}_1 \\ x_{2m} - \widehat{x}_2 \end{bmatrix}, \underline{c} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Le gain de l'observateur est choisi de la forme : $\underline{l}^T = \begin{bmatrix} \frac{1}{\delta} & \frac{1}{\delta^2} \end{bmatrix}$ avec $\delta = 0.02$.

Le réseau de neurones utilisé est de même architecture que celui utilisé au chapitre 3, sauf que x_2 est remplacée par $\widehat{x}_2$. La loi de commande neuronale est donnée par (4.26).

Les bornes nécessaires sont fixées à $\varphi_0 = 1$, $\kappa_1 = 7$, $\kappa_2 = 2$, $\kappa_3 = 3$, $\overline{\epsilon}_1 = 0.1$, $\overline{\epsilon}_2 = \overline{\epsilon}_3 = 0.01$, le polynôme filtre est $H(s) = (s + 200)$, d'où nous avons $\lambda_0 = 0.0498$.

Les lois d'ajustement (4.46)-(4.48) sont utilisées avec $\gamma_1 = \gamma_2 = \gamma_3 = 100$ et $\gamma_{c1} = \gamma_{c2} = \gamma_{c3} = 0.2$.

La figure 4.4.a montre les performances de la commande neuronale pour le comportement oscillatoire et la figure 4.4.b les performances de l'observateur d'état. On remarque qu'après une brève période transitoire, les états du système convergent vers les états de références. Les oscillations initiales sont dues au comportement du système et à la dynamique très rapide de l'observateur. Les observateurs de type "grand gain" convergent rapidement vers les états réels, mais exhibent des pics initiaux assez élevés [4]. Dans la pratique le signal de commande doit être limité pour ne pas provoquer une saturation des organes de commande.

Les figures 4.5.a et 4.5.b montrent les performances de la commande neuronale et de l'observateur pour le comportement chaotique. On peut noter les mêmes remarques que pour le cas précédent, sauf que les oscillations initiales sont moins importantes.

Les figures 4.6 et 4.7 illustrent les performances de la commande neuronale lors du passage d'un comportement oscillatoire vers un comportement chaotique à $t = 50\,\text{sec}$ et inversement. On peut noter une légère oscillation de la commande à cet instant, et une perturbation de l'état x_2, ce qui est très normal pour ce changement de dynamique.

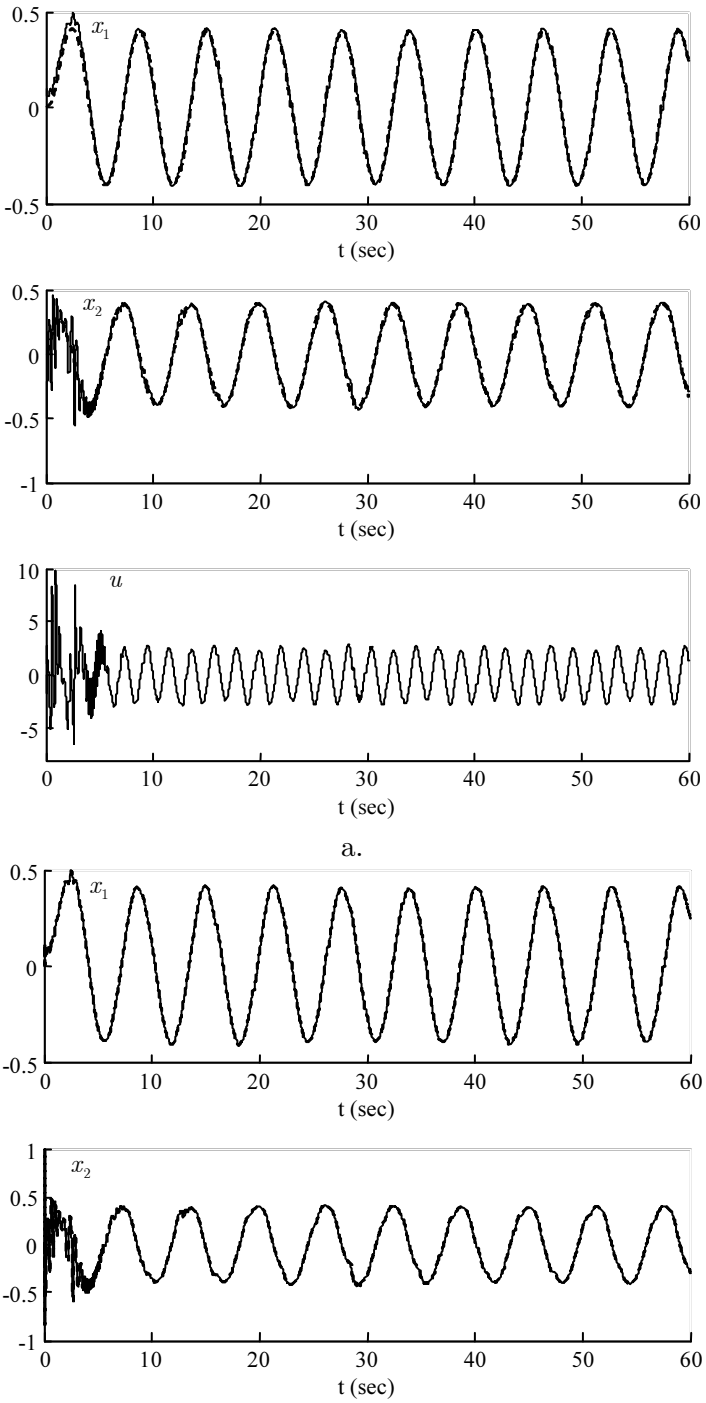
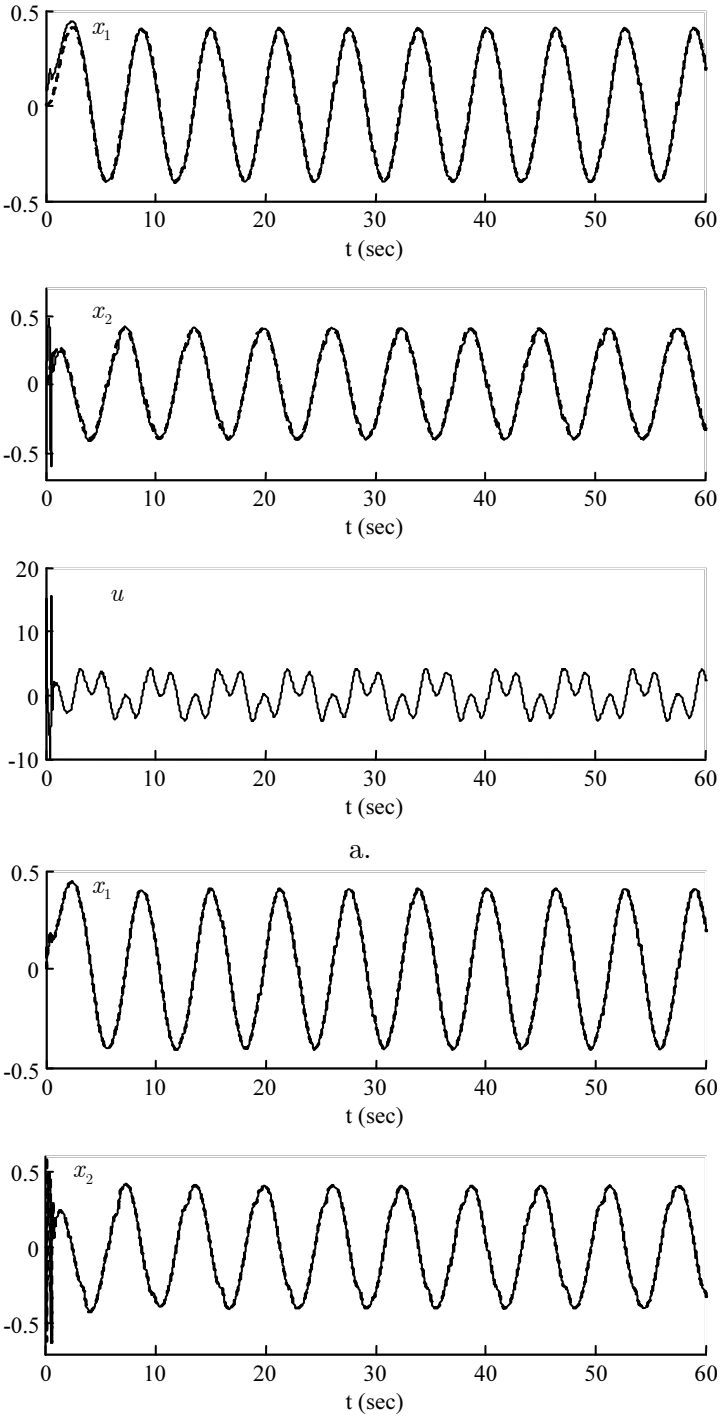


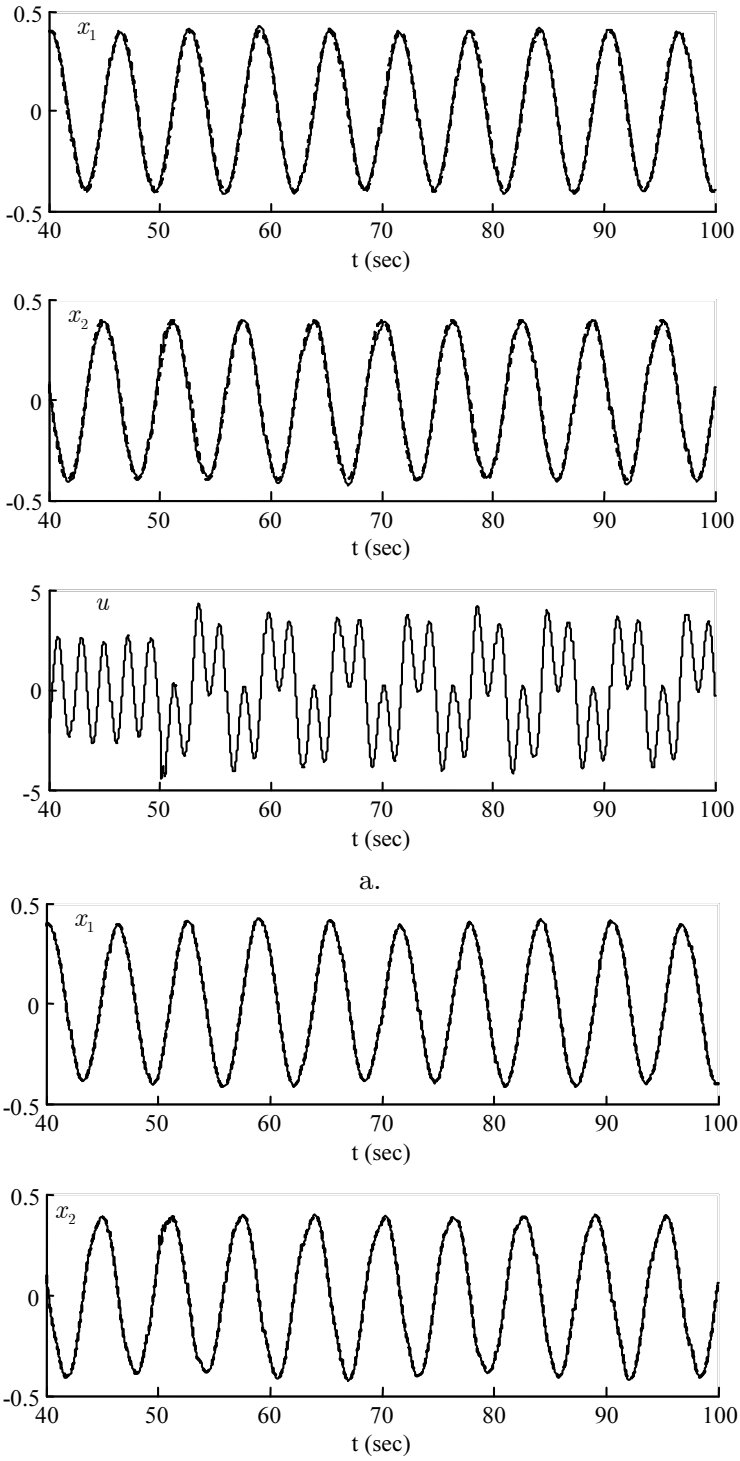
Figure 4.4 : Performances de la commande MRAC neuronale pour $\mu = 0.5$, a) poursuite b) estimation des états.



a.

b.

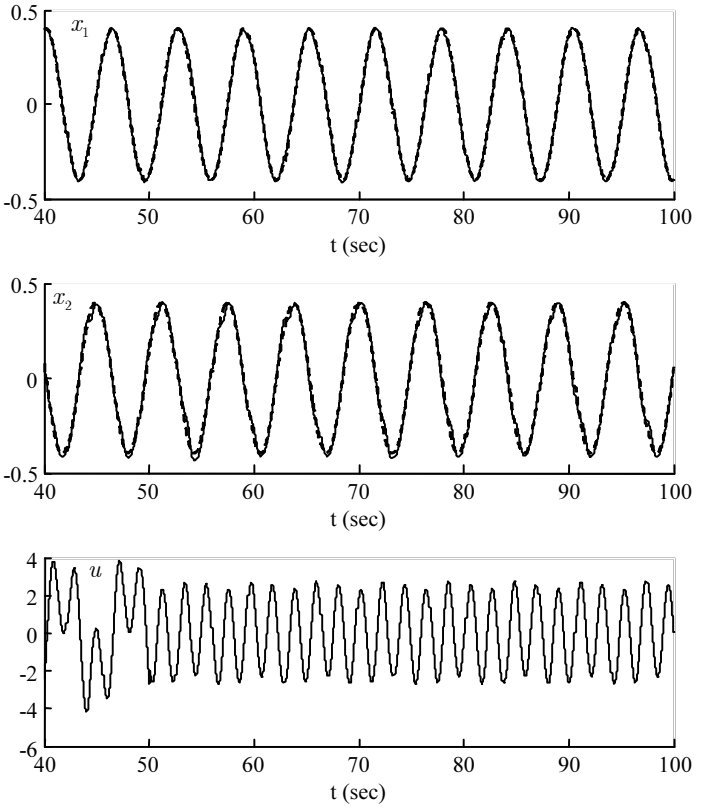
Figure 4.5 : Performances de la commande MRAC neuronale. pour $\mu = 6$, a) poursuite b) estimation des états.



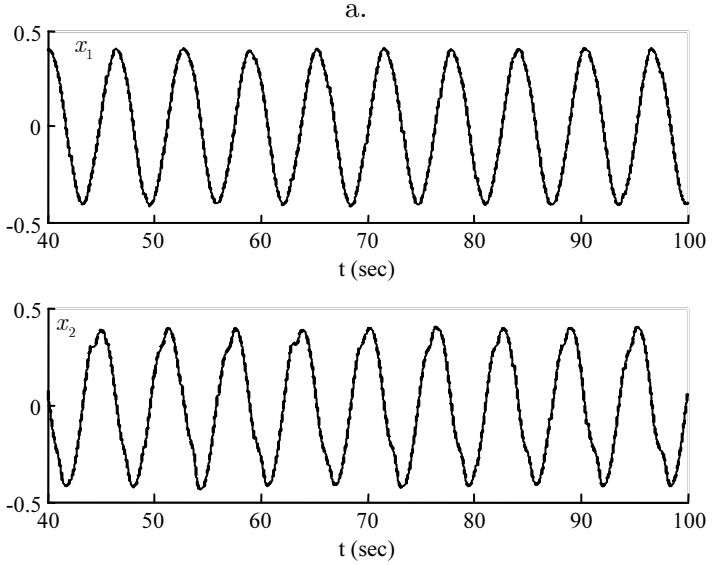
a.

b.

Figure 4.6 : Performances de la commande MRAC neuronale pour la transition de $\mu = 0.5$ vers $\mu = 6$. a) poursuite b) estimation des états.



a.



b.

Figure 4.7 : Performances de la commande MRAC neuronale pour la transition de $\mu = 6$ vers $\mu = 0.5$. a) poursuite b) estimation des états.

4.7.2 Exemple 2

La dynamique du pendule inversé est réécrite comme :

$$\dot{\underline{x}} = A\underline{x} + \underline{b}\,[f(x_1, x_2) + g(x_1)u + d(\underline{x}, t)]$$

avec

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \underline{b} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\begin{aligned} f(x_1, x_2) &= \frac{(m_c + m)g \sin x_1 - mlx_2^2 \cos x_1 \sin x_1}{l\left(\frac{4}{3}(m_c + m) - m\cos^2 x_1\right)} \\ g(x_1) &= \frac{\cos x_1}{l\left(\frac{4}{3}(m_c + m) - m\cos^2 x_1\right)} \end{aligned}$$

Le modèle de référence est donné par :

$$A_m = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}, b_m = 1$$

Si on considère que x_1 est la variable mesurée, alors l'observateur est de la forme :

$$\begin{aligned} \dot{\widehat{\underline{e}}} &= [A - \underline{lc}]\,\widehat{\underline{e}} + le_1 \\ \widehat{\underline{x}} &= \underline{x}_m - \widehat{\underline{e}} \end{aligned}$$

Le gain de l'observateur est choisi de la forme : $\underline{l}^T = \begin{bmatrix} \frac{1}{\delta} & \frac{1}{\delta^2} \end{bmatrix}$ avec $\delta = 0.01$.

Le réseau de neurones utilisé est de même architecture que celui du chapitre 3, avec la vitesse du pendule x_2 remplacée par sont estimation $\widehat{x}_2$.

Les bornes nécessaires sont fixées à $\varphi_0 = 0.5$, $\kappa_1 = 16$, $\kappa_2 = 2$, $\kappa_3 = 1$, $\overline{\epsilon}_1 = 0.1$, $\overline{\epsilon}_2 = \overline{\epsilon}_3 = 0.01$, le polynôme filtre est $H(s) = (s + 400)$, d'où nous avons $\lambda_0 = 0.0567$.

Les lois d'ajustement (4.46)-(4.48) sont utilisées avec $\gamma_1 = \gamma_2 = \gamma_3 = 100$.

La loi de commande neuronale est donnée par (4.26).

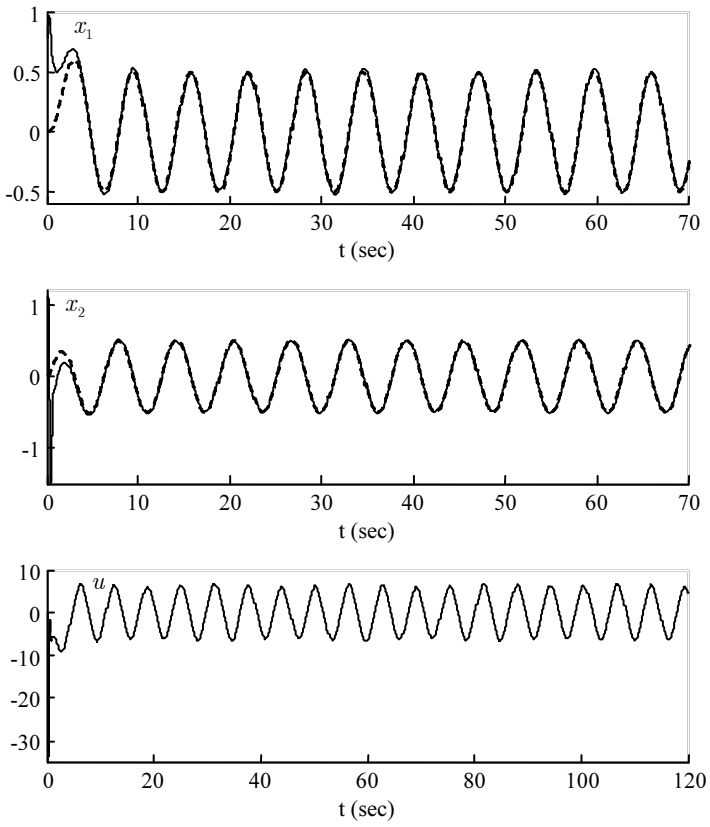
Les performances de poursuite du pendule dans le cas nominal sont montrées dans la figure 4.8.a. La vitesse et la position du pendule sont rapidement estimées par l'observateur utilisé (figure 4.8.b).

La figure 4.9 montre les performances de poursuite en présence d'une perturbation externe $d = \text{sgn}(x_2) + x_2$ introduite à $t = 60$ sec. On remarque un léger effet sur la vitesse du pendule et les estimations des états. Le signal de commande s'adapte assez bien pour compenser cet effet indésirable.

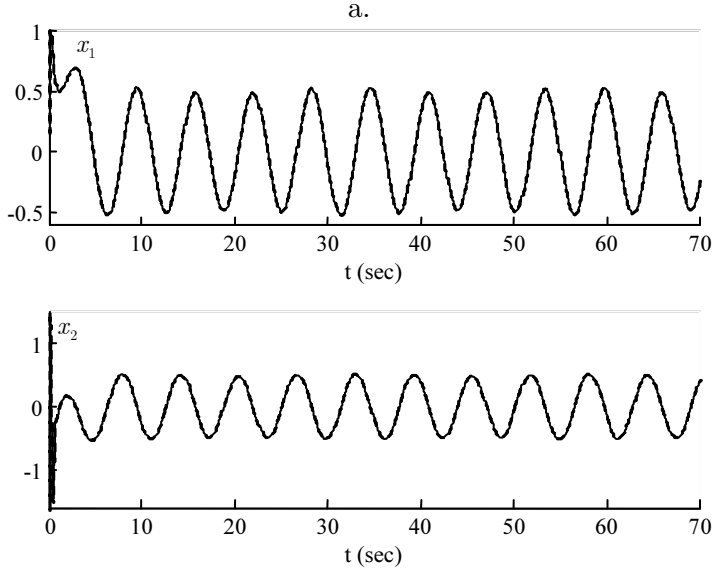
Les figures 4.10 et 4.11 illustrent l'effet d'une incertitude paramétrique, où les masses et la longueur du pendule sont, respectivement, réduites et augmentées de 50% à $t = 60$ sec. On remarque une légère oscillation du signal de commande à l'instant de changement. Cependant, la commande s'adapte assez bien et l'effet de l'incertitude est rapidement rejeté.

Les figures 4.12 et 4.13 illustrent les effets combinés de la perturbation externe et des variations paramétriques à $t = 60\,\text{sec}$. La perturbation est assez sévère, surtout dans le cas de l'augmentation des valeurs des paramètres (figure 4.13), et son effet est apparent sur le signal de commande et la vitesse du pendule. Toutefois, il est clair, que la position du pendule n'est pas perturbée et suit assez bien sa référence.

Enfin, pour la conception avec observateur, les gains des lois d'ajustement des paramètres sont augmentés, et ceci dans le but de compenser l'erreur initiale sur l'estimation des états.

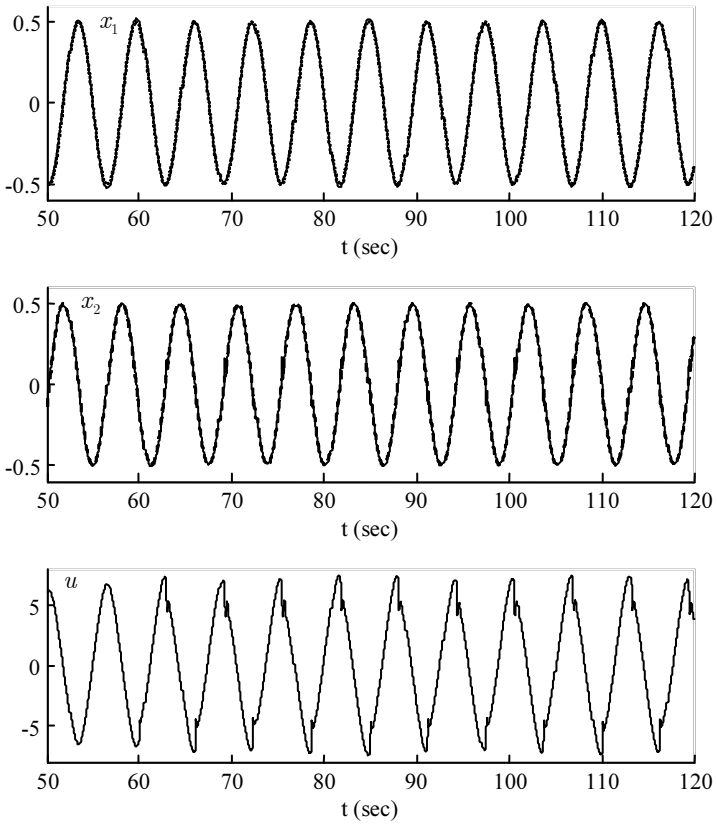


a.

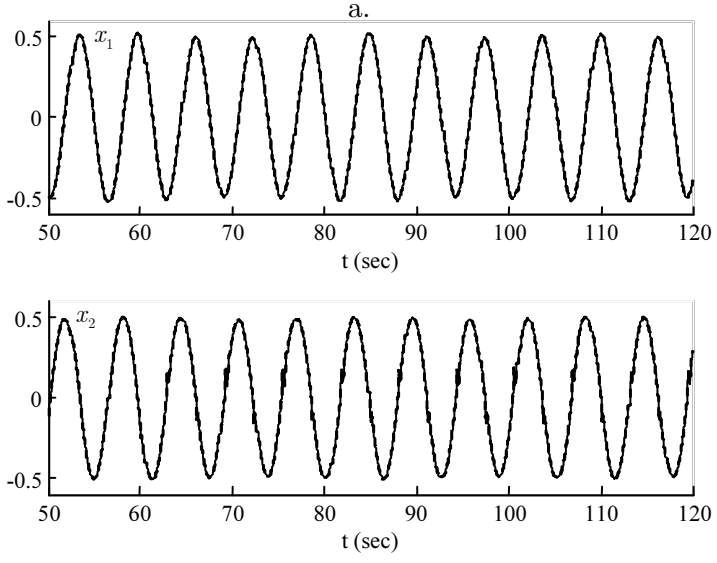


b.

Figure 4.8 : Performances de la commande MRAC du pendule (cas nominal). a) poursuite b) estimation des états.

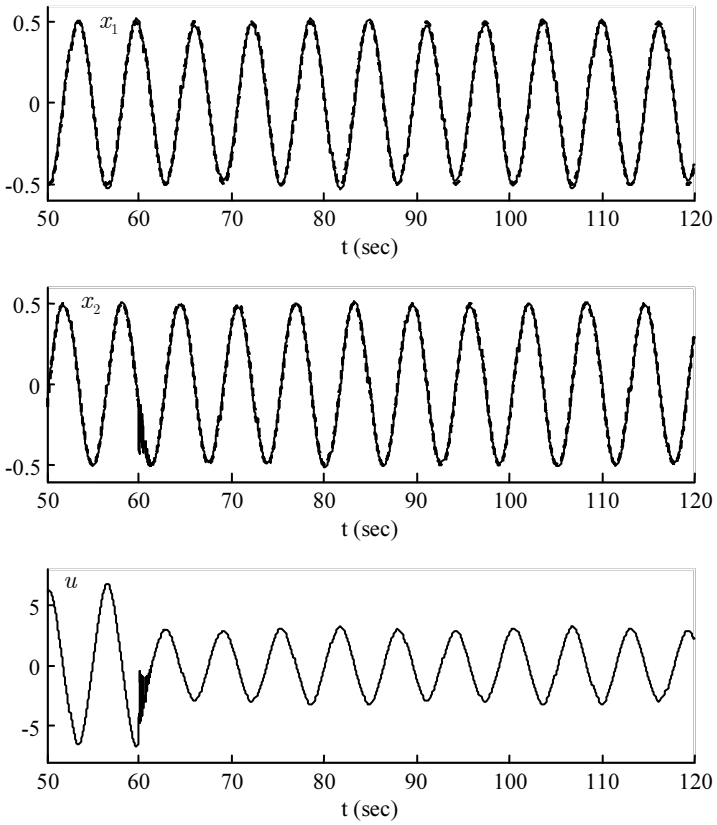


a.

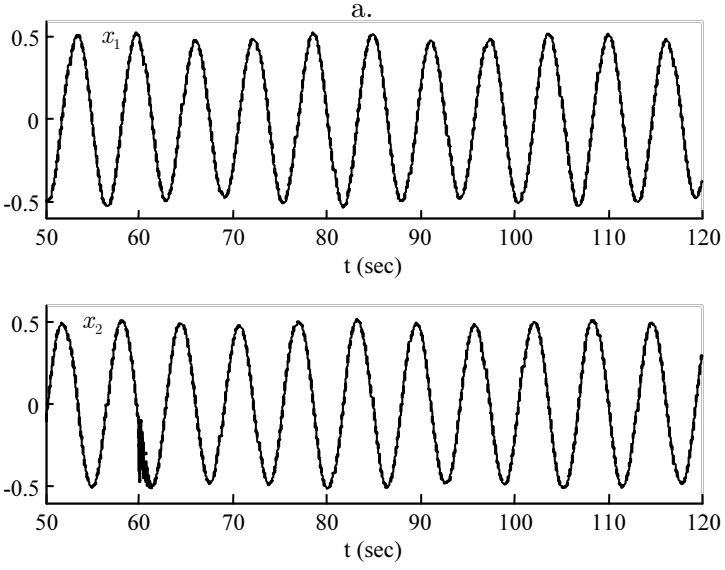


b.

Figure 4.9 : Performances de la commande MRAC du pendule avec perturbation externe à $t = 60\,\text{sec}$. a) poursuite b) estimation des états.



a.



b.

Figure 4.10 : Performances de la commande MRAC du pendule avec réduction des paramètres de 50% à $t = 60\,\text{sec}$. a) poursuite b) estimation des états.

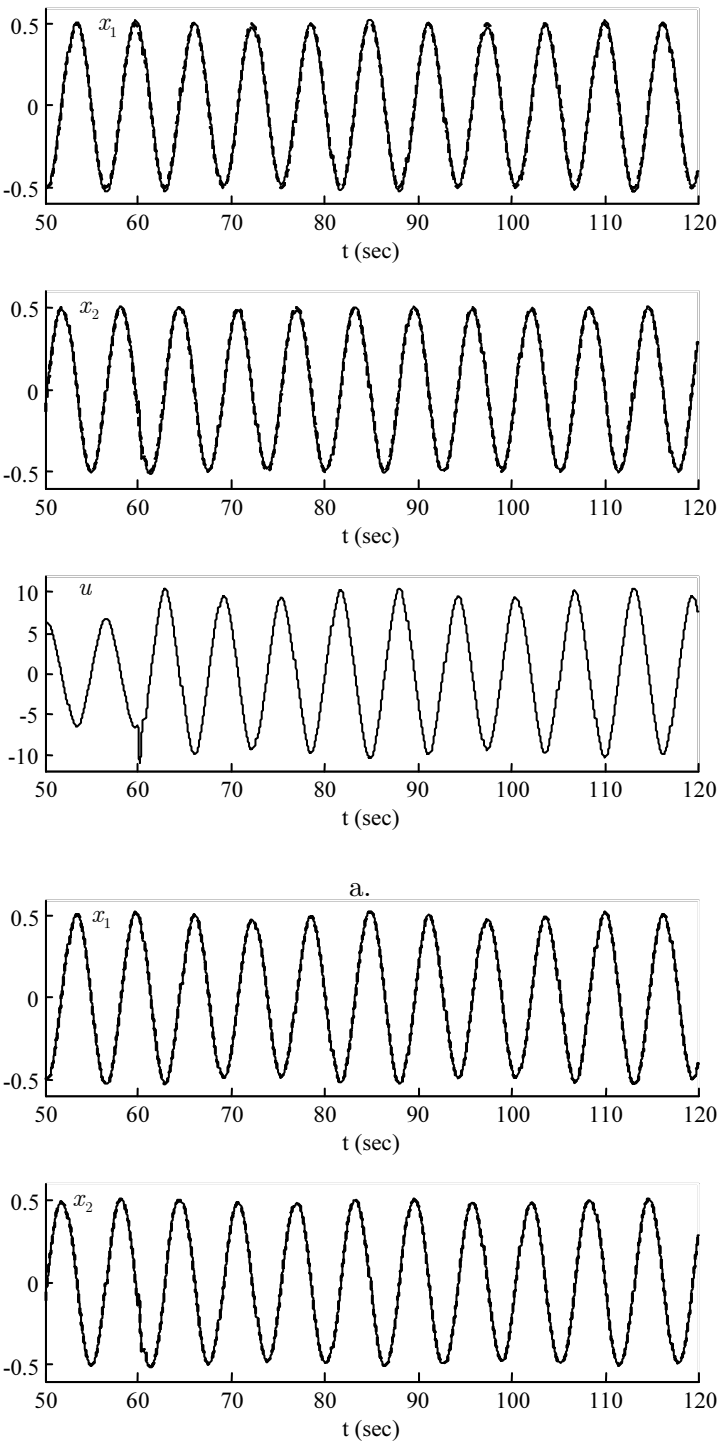
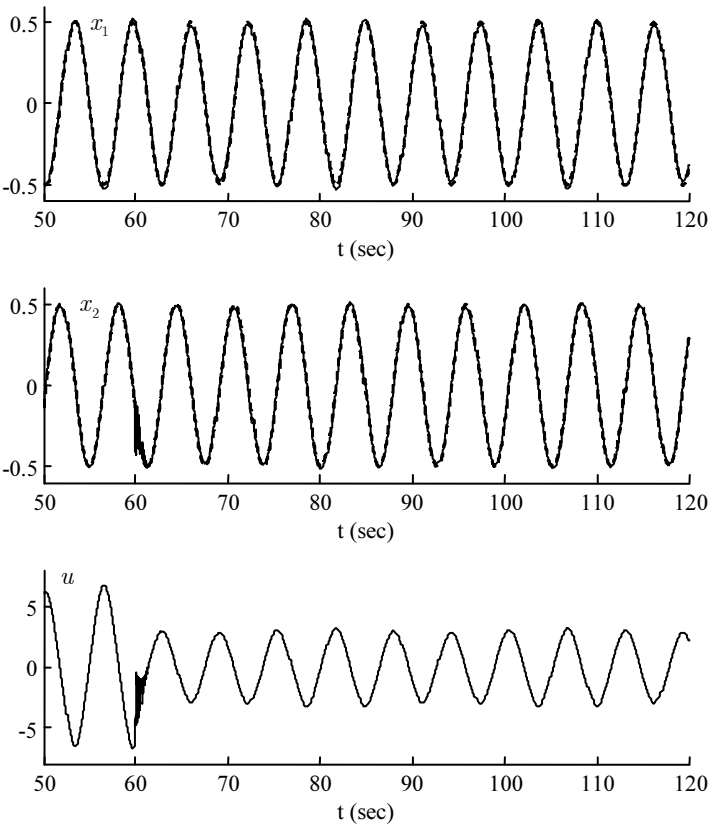
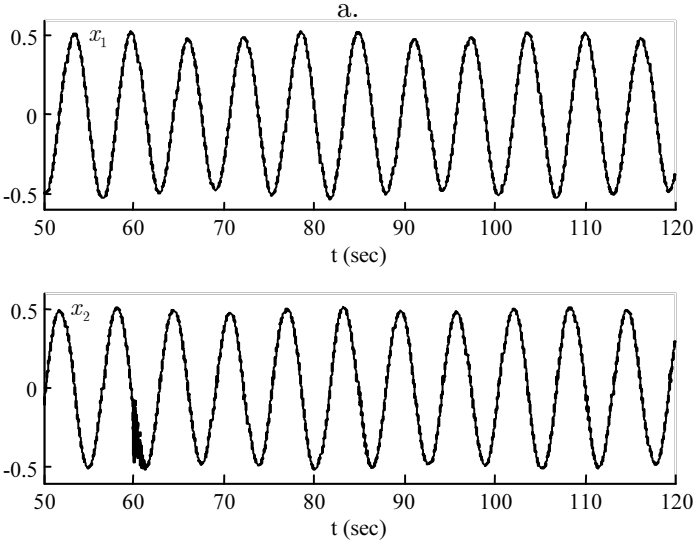


Figure 4.11 : Performances de la commande MRAC du pendule avec augmentation des paramètres de 50% à $t = 60\,\text{sec}$. a) poursuite b) estimation des états.



a.



b.

Figure 4.12 : Performances de la commande MRAC du pendule avec réduction des paramètres de 50% et perturbation externe à $t = 60\,\text{sec}$. a) poursuite b) estimation des états.

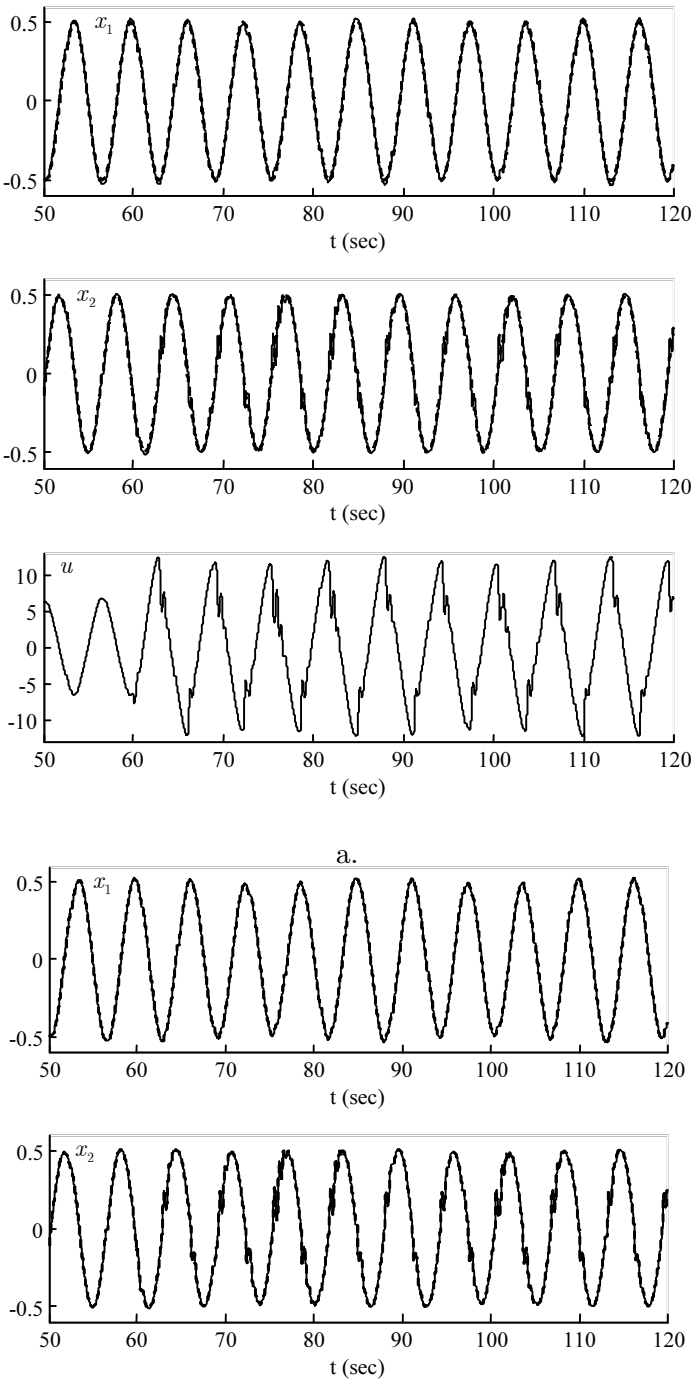


Figure 4.13 : Performances de la commande MRAC du pendule avec augmentation des paramètres de 50% et perturbation externe à $t = 60\,\text{sec}$. a) poursuite b) estimation des états.

4.8 Conclusion

Ce chapitre a présenté une extension de l'approche de la commande à modèle de référence aux systèmes non linéaires avec des états non accessibles à la mesure. L'approche basée sur un observateur à été analysée et la stabilité a été prouvée. Les résultats de simulation ont montré l'efficacité et la simplicité de l'implémentation de l'algorithme proposé.

Chapitre 5

Commande neuronale adaptative des systèmes non linèaires MIMO

5.1 Introduction

Dans ce chapitre nous présenterons une architecture de commande neuronale adaptative indirecte. Cette architecture est applicable à une large classe de systèmes non linéaires multivariables et incertains. La section 5.2 discute les approches utilisées pour l'estimation neuronale des systèmes non linéaires dans le cadre de la commande indirecte. La section 5.3 pose le problème de la commande neuronale indirecte des systèmes non linéaires multivariables. La section 5.4 détaille la conception proposée dans ce travail, à savoir l'architecture neuronale pour l'estimation de la dynamique du système, la loi de commande decouplée et les lois d'ajustement des paramètres. La section 5.5 présente l'analyse de stabilité et de robustesse de la boucle de commande. La section 5.6 donne quelques remarques sur les résultats obtenus. La section 5.7 présente les tests en simulation sur un robot manipulateur. La section 5.8 conclut le chapitre.

5.2 Commande adaptative indirecte et identification

Une approche indirecte de la commande adaptative est constituée d'un modèle d'identification (approximateur) qui doit capter l'essentiel du comportement du système, et une loi de commande (contrôleur) basée sur le principe "d'équivalence de certitude" [28]. Ce principe calcule la loi de commande en supposant qu'à chaque instant l'approximateur reproduit fidèlement le comportement réel du système à contrôler. La commande adaptative indirecte est illustrée par la figure 5.1.

Dans les applications de contrôle, l'objectif de l'identification du système est de déterminer, à partir d'un certain nombre de signaux, une description de la dynamique du système qui peut être utilisée pour l'analyse du système et pour la conception du contrôleur. L'identification des systèmes non linéaires est plus difficile, comparée à celle des systèmes linéaires, puisque les termes non linéaires dans la dynamique du système

ne répondent pas à une description générale, comme une fonction linéaire. Depuis le travail de Narendra [16], qui a posé les fondements de l'identification des systèmes non linéaires par des réseaux MLP, de nombreuses approches d'identification des systèmes non linéaires par les réseaux de neurones ont été développées [49]. Dans la plupart des méthodes d'identification proposées, des modèles discrets sont employés. Le schéma général de l'identification des systèmes non linéaires par réseaux de neurones est représenté sur la figure 5.2.

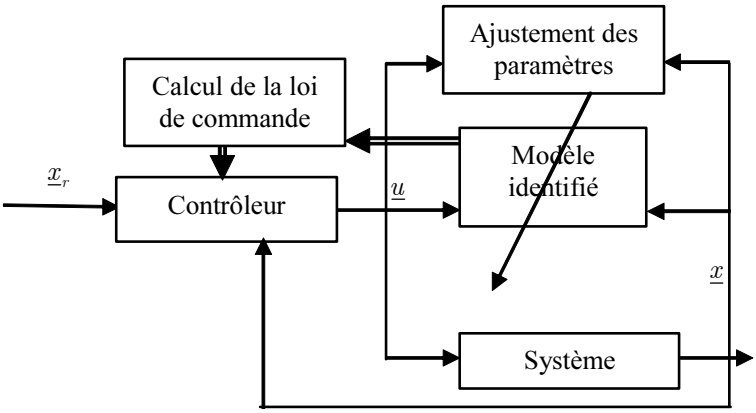


Figure 5.1. Schéma de principe de la commande adaptative indirecte.

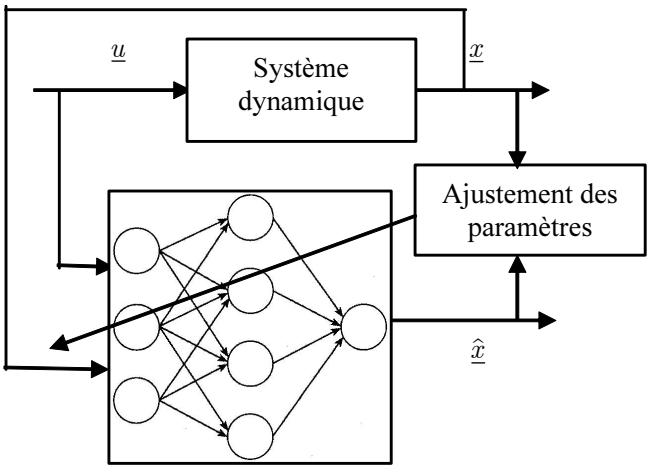


Figure 5.2. Identification neuronale du modèle direct d'un système dynamique.

Pour les applications de contrôle, des modèles dynamiques en temps continu sont plus utiles puisque les systèmes physiques à identifier sont généralement continus. De plus, les méthodes de commande non linéaire, telles que la linéarisation par bouclage ou le backstepping [1-4], utilisent des modèles d'état continus pour l'analyse et la conception de contrôleurs pour des systèmes non linéaires. En outre, les techniques de la commande discrète ne sont pas facilement applicables aux systèmes non linéaires.

Les premières approches proposées pour la commande neuronale indirecte des systèmes non linéaires, peuvent être classées en deux catégories [16, 54].

5.2.1 Approche inverse

Dans ce type d'approches, un réseau de neurones est entraîné hors ligne pour obtenir le modèle inverse du système à commander (figure 5.3.a). Par la suite ce modèle inverse est utilisé comme contrôleur neuronal, qui pour une référence désirée produit la commande nécessaire au système (figure 5.3.b). Cependant, cette technique présente deux inconvénients majeurs. Le premier inconvénient est que l'identification doit être réalisée en boucle ouverte, ce qui suppose que le système est stable en boucle ouverte ; sinon l'identification doit être réalisée avec le système sous contrôle (avec p. ex. un régulateur classique), ce qui produit un modèle neuronal entraîné seulement pour le point de fonctionnement considéré. Le deuxième inconvénient, et que le contrôleur neuronal (figure 5.3.b) fonctionne en boucle ouverte, donc il n'a aucune chance de répondre correctement à une variation du comportement du système ou à des perturbations externes.

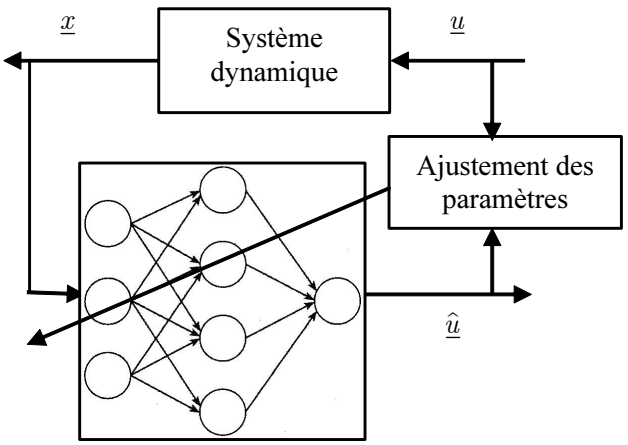


Figure 5.3.a. Identification neuronale du modèle inverse d'un système dynamique.

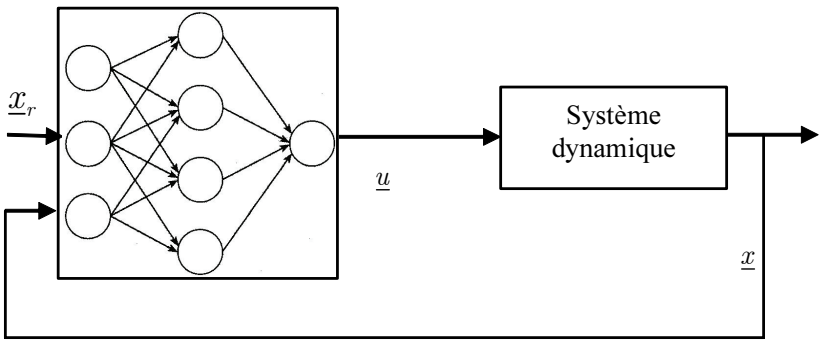


Figure 5.3.b. Commande neuronale inverse d'un système dynamique.

5.2.2 Approche identification/contrôle

Ce type de commande utilise une combinaison de deux réseaux de neurones (figure 5.4). Le premier réseau est entraîné pour identifier le modèle dynamique du système

non linéaire. Les paramètres de ce réseau sont ajustés en utilisant comme signal d'erreur la différence entre la sortie du système et celle prédite par le réseau de neurones. Une phase d'apprentissage préliminaire peut être effectuée hors ligne comme sur le schéma de la figure 5.2. Le deuxième réseau de neurones est entraîné pour produire la commande nécessaire au suivi de la trajectoire de référence. Il utilise comme signal d'erreur l'erreur de poursuite entre la sortie estimée du réseau de neurones et la trajectoire de référence. L'erreur est rétro-propagée à travers le réseau d'identification, sans que ses paramètres soient ajustés. Les inconvénients de cette technique sont la complexité de la structure de commande avec deux réseaux et le temps nécessaire aux calculs et aux ajustements des paramètres.

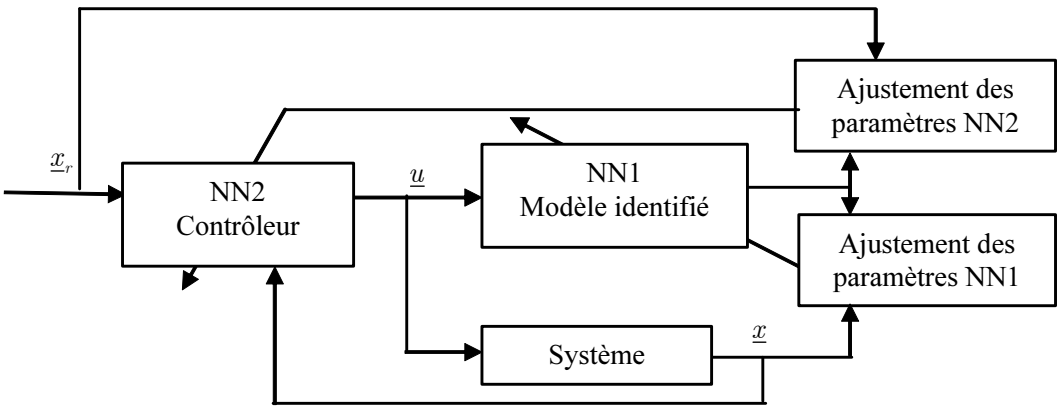


Figure 5.4. Commande neuronale indirecte (identification/contrôle) d'un système dynamique.

Deux inconvénients communs à ces premières techniques sont :

1. L'absence d'une analyse rigoureuse de la stabilité. Ce qui rend toute application pratique hasardeuse.
2. Ces techniques utilisent principalement comme signal d'ajustement l'erreur d'identification. Ce qui signifie que la qualité de la poursuite dépend largement de la convergence du modèle identifié. D'un autre côté, le souci principal de la commande adaptative est d'assurer une bonne performance de poursuite, en dépit de la qualité de l'identification.

Comme, nous le virons dans l'approche proposée, la conception doit être repensée dans le sens à utiliser comme critère de performance l'erreur de poursuite. De plus il faut unifier les paramètres du contrôleur avec ceux du modèle d'identification, de sorte à simplifier l'architecture de commande et réduire le temps de calcul.

5.3 Position du problème

Nous considérerons la classe de systèmes non linéaires MIMO donnée par :

$$\dot{\underline{x}} = A\underline{x} + B\left[\underline{f}(\underline{x}) + G(\underline{x})\,\underline{u} + \underline{d}(\underline{x},t)\right] \tag{5.1}$$

$$\underline{y} = C\underline{x} \tag{5.2}$$

où $\underline{y}^T = \left[\begin{array}{cccc} y_1 & y_2 & ... & y_p \end{array}\right] \in \Re^p$ est le vecteur des sorties, $\underline{u}^T = \left[\begin{array}{cccc} u_1 & u_2 & ... & u_p \end{array}\right] \in \Re^p$ est le vecteur des entrées, et

$$\underline{x}^T = \left[\begin{array}{ccccccccccc} y_1 & \dot{y}_1 & ... & y_1^{(n_1-1)} & y_2 & \dot{y}_2 & ... & y_2^{(n_2-1)} & ... & y_p & \dot{y}_p & ... & y_p^{(n_p-1)} \end{array}\right]$$

est le vecteur d'état, avec $n_1 + n_2 + .. + n_p = n$. $f(\underline{x}) \in \Re^{p\times 1}$, $G(\underline{x}) \in \Re^{p\times p}$ sont non linéaires inconnues mais lisses, $\underline{d}(\underline{x},t) \in \Re^{p\times 1}$ est le vecteur qui regroupe les incertitudes et les perturbations externes. Les structures détaillées des termes intervenant dans (5.1)-(5.2) sont comme suit :

$$\underline{f}(\underline{x}) = \left[\begin{array}{c} f_1(\underline{x}) \\ f_2(\underline{x}) \\ \vdots \\ f_p(\underline{x}) \end{array}\right], G(\underline{x}) = \left[\begin{array}{cccc} g_{11}(\underline{x}) & g_{12}(\underline{x}) & \dots & g_{1p}(\underline{x}) \\ g_{21}(\underline{x}) & g_{22}(\underline{x}) & \dots & g_{2p}(\underline{x}) \\ \vdots & \vdots & \ddots & \vdots \\ g_{p1}(\underline{x}) & g_{p2}(\underline{x}) & \dots & g_{pp}(\underline{x}) \end{array}\right],$$

$$\underline{d}(\underline{x},t) = \left[\begin{array}{c} d_1(\underline{x},t) \\ d_2(\underline{x},t) \\ \vdots \\ d_p(\underline{x},t) \end{array}\right]$$

et

$$A = \left[\begin{array}{cccc} A_1 & 0 & \dots & 0 \\ 0 & A_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & A_p \end{array}\right] \in \Re^{n\times n}, B = \left[\begin{array}{cccc} \underline{b}_1 & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{b}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underline{0} \\ \underline{0} & \dots & \underline{0} & \underline{b}_p \end{array}\right] \in \Re^{n\times p}$$

$$C = \left[\begin{array}{cccc} \underline{c}_1 & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{c}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underline{0} \\ \underline{0} & \dots & \underline{0} & \underline{c}_p \end{array}\right] \in \Re^{p\times n}$$

où

$$A_i = \left[\begin{array}{c} \underline{0}_{(n_i-1)\times 1} \quad I_{n_i-1} \\ \underline{0}_{1\times n_i} \end{array}\right] \in \Re^{n_i\times n_i}, \underline{b}_i = \left[\begin{array}{c} 0 \\ \vdots \\ 0 \\ 1 \end{array}\right] \in \Re^{n_i\times 1},$$

$$\underline{c}_i = \left[\begin{array}{cccc} 1 & 0 & ... & 0 \end{array}\right] \in \Re^{1\times n_i}, i = 1, ..., p$$

Le système non linéaire (5.1)-(5.2) satisfait les hypothèses suivantes :

P1 Le vecteur de fonctions $\underline{f}(\underline{x})$ est borné, tel que $\left|\underline{f}(\underline{x})\right| \leq f_{\max}$, où $f_{\max}$ est une constante donnée.

P2 Les éléments diagonaux $g_{ii}(\underline{x}), i = 1, ..., p$ sont strictement positifs et bornés dans l'espace de commande $\Omega_{\underline{x}}$, tel que

$$g_{\min} \leq \sqrt{g_{11}^2(\underline{x}) + g_{22}^2(\underline{x}) + ... + g_{pp}^2(\underline{x})} \leq g_{\max}$$

où $g_{\min}$ et $g_{\max}$ sont des constantes données.

P3 Les éléments non diagonaux $g_{ij}(\underline{x}), i, j = 1, ..., p$ et $i \neq j$ sont bornés.

P4 Les éléments de $\underline{d}(\underline{x}, t)$ sont bornés tels que $|\underline{d}(\underline{x}, t)| \leq d_{\max}$ avec $d_{\max}$ est une constante donnée.

Avant de poser le problème de commande, commençons par définir les trajectoires de références. Nous supposerons que nous avons p trajectoires de référence $y_{ri}, i = 1, ..., p$ bornées, avec des dérivées $y_{ri}^{(j)}, i = 1, ..., p, j = 1, ..., n_i$ bornées. Si on définit le vecteur de référence comme :

$$\underline{x}_r^T = \left[\begin{array}{cccccccccccc} y_{r1} & \dot{y}_{r1} & ... & y_{r1}^{(n_1-1)} & y_{r2} & \dot{y}_{r2} & ... & y_{r2}^{(n_2-1)} & ... & y_{rp} & \dot{y}_{rp} & ... \quad y_{rp}^{(n_p-1)} \end{array} \right]$$

alors, la dynamique de la trajectoire de référence peut être décrite par l'équation d'état :

$$\dot{\underline{x}}_r = A\underline{x}_r + B\underline{v} \tag{5.3}$$

avec $\underline{v}^T = \left[\begin{array}{cccc} y_{r1}^{(n_1)} & y_{r2}^{(n_2)} & ... & y_{rp}^{(n_p)} \end{array} \right] \in \Re^{1\times p}$.

Maintenant le problème de commande revient à concevoir une loi de commande $\underline{u}(\underline{x})$ telle que les états du système (5.1)-(5.2) suivent les états de référence $\underline{x}_r$ (5.3), avec la condition que tous les signaux en boucle fermée restent bornés.

5.4 Conception de la commande neuronale

5.4.1 Approximation neuronale

Suivant les résultats de l'approximation universelle exposés au chapitre 2, il existe des réseaux de neurones capables d'approcher les non linéarités du système non linéaire (5.1), telle que :

$$\begin{array}{l} f_i(\underline{x}) = \underline{\theta}_{f_i}^{*T}\underline{\phi}_i(\underline{x}) + \epsilon_i(\underline{x}) \\ g_{ii}(\underline{x}) = \underline{\theta}_{g_i}^{*T}\underline{\psi}_i(\underline{x}) + \varepsilon_i(\underline{x}) \end{array}, i = 1, ..., p \tag{5.4}$$

où $\underline{\phi}_i(\underline{x}) \in \Re^{q_i\times 1}$, $\underline{\psi}_i(\underline{x}) \in \Re^{p_i\times 1}, i = 1, ..., p$ sont les vecteurs des sorties des neurones des couches cachées, avec q_i et p_i les nombres de neurones dans la couche cachée de chaque réseau. Les termes $\epsilon_i(\underline{x})$, $\varepsilon_i(\underline{x}), i = 1, ..., p$ sont les erreurs d'approximation inhérentes dues à la taille finie des réseaux de neurones. Les poids optimaux $\underline{\theta}_{f_i}^* \in \Re^{q_i\times 1}$ et $\underline{\theta}_{g_i}^* \in \Re^{p_i\times 1}$ définissent les paramètres ajustables des réseaux de neurones. Typiquement

$\underline{\theta}_{f_i}^*$ et $\underline{\theta}_{g_i}^*$ sont choisis pour minimiser $\epsilon_i(\underline{x})$ et $\varepsilon_i(\underline{x})$ dans une région compacte $\Omega_{\underline{x}}$, tel que :

$$\underline{\theta}_{f_i}^* = \arg \min_{\underline{\theta}_{f_i} \in \Omega_{\underline{\theta}_f}} \left\{ \sup_{\underline{x} \in \Omega_{\underline{x}}} \left| f_i(\underline{x}) - \underline{\theta}_{f_i}^{*T} \underline{\phi}_i(\underline{x}) \right| \right\}$$

$$\underline{\theta}_{g_i}^* = \arg \min_{\underline{\theta}_{g_i} \in \Omega_{\underline{\theta}_g}} \left\{ \sup_{\underline{x} \in \Omega_{\underline{x}}} \left| g_{ii}(\underline{x}) - \underline{\theta}_{g_i}^{*T} \underline{\psi}_i(\underline{x}) \right| \right\}$$

C1. Les erreurs d'approximation des réseaux de neurones sont bornées par $|\epsilon_i(\underline{x})| \leq \epsilon_i$ et $|\varepsilon_i(\underline{x})| \leq \varepsilon_i$, $i = 1, ..., p$, pour certaines constantes ϵ_i et ε_i.

L'hypothèse C1 résulte de la propriété d'approximation universelle des réseaux de neurones, qui stipule que les réseaux décrits au chapitre 2 peuvent approximer n'importe quelle fonction régulière sur un espace compact avec une erreur d'approximation finie.

A partir des expressions (5.4), nous pouvons écrire :

$$\begin{aligned} \underline{f}(\underline{x}) &= \Theta_f^* \Phi(\underline{x}) + \underline{\epsilon}(\underline{x}) \\ G(\underline{x})\,\underline{u} &= \Theta_g^* \Psi(\underline{x})\,\underline{u} + H(\underline{x})\,\underline{u} \end{aligned} \tag{5.5}$$

où

$$\Phi(\underline{x}) = \begin{bmatrix} \underline{\phi}_1(\underline{x}) \\ \underline{\phi}_2(\underline{x}) \\ \vdots \\ \underline{\phi}_p(\underline{x}) \end{bmatrix} \in \Re^{(q_i p)\times 1}, \Psi(\underline{x}) = \begin{bmatrix} \underline{\psi}_1(\underline{x}) & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{\psi}_2(\underline{x}) & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underline{0} \\ \underline{0} & \dots & \underline{0} & \underline{\psi}_p(\underline{x}) \end{bmatrix} \in \Re^{(p_i p)\times p}$$

$$\Theta_f^* = \begin{bmatrix} \underline{\theta}_{f_1}^{*T} & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{\theta}_{f_2}^{*T} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underline{0} \\ \underline{0} & \dots & \underline{0} & \underline{\theta}_{f_p}^{*T} \end{bmatrix} \in \Re^{p\times(q_i p)}, \Theta_g^* = \begin{bmatrix} \underline{\theta}_{g_1}^{*T} & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{\theta}_{g_2}^{*T} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underline{0} \\ \underline{0} & \dots & \underline{0} & \underline{\theta}_{g_p}^{*T} \end{bmatrix} \in \Re^{p\times(p_i p)}$$

et

$$\underline{\epsilon}(\underline{x}) = \begin{bmatrix} \epsilon_1(\underline{x}) \\ \epsilon_2(\underline{x}) \\ \vdots \\ \epsilon_p(\underline{x}) \end{bmatrix} \in \Re^{p\times 1}, H(\underline{x}) = \begin{bmatrix} \varepsilon_1(\underline{x}) & g_{12}(\underline{x}) & \dots & g_{1p}(\underline{x}) \\ g_{21}(\underline{x}) & \varepsilon_2(\underline{x}) & & \vdots \\ \vdots & & \ddots & g_{(p-1)p}(\underline{x}) \\ g_{p1}(\underline{x}) & \dots & g_{p(p-1)}(\underline{x}) & \varepsilon_p(\underline{x}) \end{bmatrix} \in \Re^{p\times p}$$

Notons que les $\underline{0}$ sont des vecteurs nuls, lignes ou colonnes de dimensions appropriées suivant le cas.

L'introduction de (5.5) dans (5.1), permet d'écrire la dynamique du système comme :

$$\dot{\underline{x}} = A\underline{x} + B\left[\Theta_f^* \Phi(\underline{x}) + \Theta_g^* \Psi(\underline{x})\,\underline{u} + H(\underline{x})\,\underline{u} + \underline{\omega}(\underline{x})\right] \tag{5.6}$$

où

$$\underline{\omega}\left(\underline{x}\right) = \underline{\epsilon}\left(\underline{x}\right) + \underline{d}\left(\underline{x}, t\right)$$

C2. La matrice $H\left(\underline{x}\right)$ est bornée par $\left|H\left(\underline{x}\right)\right| \leq h_{\max}$, et le vecteur $\underline{\omega}\left(\underline{x}\right)$ est borné par $\left|\underline{\omega}\left(\underline{x}\right)\right| \leq \omega_{\max}$.

Cette hypothèse découle de la bornitude de la matrice $G\left(\underline{x}\right)$, de la bornitude du vecteur de perturbation $\underline{d}\left(\underline{x}, t\right)$ et des erreurs d'approximation finies.

5.4.2 Loi de commande neuronale adaptative

La soustraction de (5.6) de (5.3) permet d'écrire la dynamique de l'erreur de poursuite comme :

$$\dot{\underline{e}} = A\underline{e} - B\left[-\underline{v} + \Theta_f^*\Phi\left(\underline{x}\right) + \Theta_g^*\Psi\left(\underline{x}\right)\underline{u} + H\left(\underline{x}\right)\underline{u} + \underline{\omega}\left(\underline{x}\right)\right] \tag{5.7}$$

où $\underline{e} = \underline{x}_r - \underline{x}$ est l'erreur de poursuite.

En se basant sur le principe d'équivalence de certitude, utilisé dans les techniques de commande adaptative indirecte [28], et le modèle estimé dans (5.6), les entrées de commande sont définies comme :

$$\underline{u} = \left[\Theta_g\Psi\left(\underline{x}\right)\right]^{-1}\left[-\Theta_f\Phi\left(\underline{x}\right) + \underline{v} + K\underline{e}\right] \tag{5.8}$$

où Θ_g, Θ_f sont les paramètres actuels (estimés) des réseaux de neurones, et $K \in \Re^{p\times n}$ est une matrice de gain, choisie telle que la matrice $A_c = [A - BK]$ est Hurwitz.

Alors, l'introduction de la loi de commande (5.8) dans (5.7) produit :

$$\dot{\underline{e}} = A_c\underline{e} - B\left[\widetilde{\Theta}_f\Phi\left(\underline{x}\right) + \widetilde{\Theta}_g\Psi\left(\underline{x}\right)\underline{u} + H\left(\underline{x}\right)\underline{u} + \underline{\omega}\left(\underline{x}\right)\right] \tag{5.9}$$

où $\widetilde{\Theta}_f = \Theta_f^* - \Theta_f$ et $\widetilde{\Theta}_g = \Theta_g^* - \Theta_g$ sont les erreurs d'estimation des paramètres.

A partir de (5.9), il peut être remarqué que la dynamique de l'erreur de poursuite est gouvernée par : 1) les erreurs d'estimation des paramètres des réseaux de neurones représentées par les termes $\widetilde{\Theta}_f\Phi\left(\underline{x}\right)$ et $\widetilde{\Theta}_g\Psi\left(\underline{x}\right)\underline{u}$, et 2) par les termes de couplage, les erreurs d'approximation et les incertitudes représentés par les termes $H\left(\underline{x}\right)$ et $\underline{\omega}\left(\underline{x}\right)$.

Pour concevoir les lois d'ajustement des paramètres des réseaux de neurones et pour assurer la bornitude des signaux impliqués dans la boucle de commande, l'hypothèse suivante est employée :

C3. Les paramètres du réseau de neurones sont bornés par les ensembles de contraintes Ω_f et Ω_g telles que : $\Omega_f = \{\Theta_f \,|\, |\Theta_f| \leq f_{\max}\}$ et $\Omega_g = \{\Theta_g | g_{\min} \leq |\Theta_g| \leq g_{\max}\}$, respectivement.

Les bornes employées dans l'hypothèse C3 résultent des hypothèses C1-C2 et sont introduites pour assurer la bornitude des sorties des réseaux de neurones.

5.4.3 Lois d'ajustement des paramètres

Pour contraindre les paramètres Θ_f et Θ_g dans les régions Ω_f et Ω_g, respectivement, nous utiliserons l'algorithme de projection suivant [28] :

$$\dot{\Theta}_f = -\gamma_1 B^T P \underline{e} \Phi^T(\underline{x}) + I_f \gamma_1 \mathrm{tr}\left[B^T P \underline{e} \Phi^T(\underline{x}) \Theta_f^T\right] \left(\frac{1+|\Theta_f|}{f_{\max}}\right)^2 \Theta_f \tag{5.10}$$

$$\dot{\Theta}_g = -\gamma_2 B^T P \underline{eu}^T \Psi^T(\underline{x}) + I_g \gamma_2 \mathrm{tr}\left[B^T P \underline{eu}^T \Psi^T(\underline{x}) \Theta_g^T\right] \left(\frac{1+|\Theta_g|}{g_{\max}}\right)^2 \Theta_g \tag{5.11}$$

avec

$$I_f = \begin{cases} 0 & \text{si } |\Theta_f| < f_{\max} \text{ ou } (|\Theta_f| = f_{\max} \text{ et } \mathrm{tr}\left[B^T P \underline{e} \Phi^T(\underline{x}) \Theta_f^T\right] \geq 0) \\ 1 & \text{si } |\Theta_f| = f_{\max} \text{ et } \mathrm{tr}\left[B^T P \underline{e} \Phi^T(\underline{x}) \Theta_f^T\right] < 0 \end{cases}$$

$$I_g = \begin{cases} 0 & \text{si } |\Theta_g| < g_{\max} \text{ ou } (|\Theta_g| = g_{\max} \text{ et } \mathrm{tr}\left[B^T P \underline{eu}^T \Psi^T(\underline{x}) \Theta_g^T\right] \geq 0) \\ 1 & \text{si } |\Theta_g| = g_{\max} \text{ et } \mathrm{tr}\left[B^T P \underline{eu}^T \Psi^T(\underline{x}) \Theta_g^T\right] < 0 \end{cases}$$

et $\gamma_1, \gamma_2 > 0$ sont des paramètres de conception, et $P = P^T > 0$ est la solution, pour une matrice donnée $Q = Q^T > 0$, de l'équation de Lyapunov

$$A_c^T P + P A_c = -Q \tag{5.12}$$

Cependant, pour garantir $|\Theta_g| \geq g_{\min}$ telle que l'inverse de $\Theta_g \Psi(\underline{x})$ existe toujours, nous utiliserons la loi d'ajustement des paramètre Θ_g suivante :

1. Pour tout élément $[\Theta_g]_{ij} = g_{\min}$, nous utiliserons la relation :

$$\left[\dot{\Theta}_g\right]_{ij} = \begin{cases} -\gamma_2 \left[B^T P \underline{eu}^T \Psi^T(\underline{x})\right]_{ij} & \text{si } \left[B^T P \underline{eu}^T \Psi^T(\underline{x})\right]_{ij} < 0 \\ 0 & \text{si } \left[B^T P \underline{eu}^T \Psi^T(\underline{x})\right]_{ij} \geq 0 \end{cases} \tag{5.13}$$

2. sinon, nous utiliserons (5.11).

où $[\Theta_g]_{ij}$ indique l'élément ij de la matrice Θ_g.
Théorème 5.1 :
Les lois d'ajustement (5.10)-(5.11) et (5.13) garantissent que :

1. $g_{\min} \leq |\Theta_g| \leq g_{\max}$
2. $|\Theta_f| \leq f_{\max}$

Preuve :

1. Pour prouver que $|\Theta_g| \leq g_{\max}, \forall t$, prenons la fonction de Lyapunov :

$$V_g = \frac{1}{2\gamma_2} \mathrm{tr}\left[\Theta_g^T \Theta_g\right]$$

alors

$$\dot{V}_g = \frac{1}{\gamma_2} \mathrm{tr}\left[\dot{\Theta}_g^T \Theta_g\right]$$

Si la première ligne de (5.11) est vraie, nous avons $|\Theta_g| < g_{\max}$ ou

$$\dot{V}_g = -\mathrm{tr}\left[\left(B^T P \underline{e} \underline{u}^T \Psi^T(\underline{x})\right)^T \Theta_g\right] \leq 0$$

lorsque $|\Theta_g| = g_{\max}$, ce qui nous donne toujours $|\Theta_g| \leq g_{\max}$.

Si la deuxième ligne de (5.11) est vraie, nous avons $|\Theta_g| = g_{\max}$ et

$$\begin{aligned}\dot{V}_g &= \mathrm{tr}\left[-\left(B^T P \underline{e} \underline{u}^T \Psi^T(\underline{x})\right)^T \Theta_g + \mathrm{tr}\left[B^T P \underline{e} \underline{u}^T \Psi^T(\underline{x})\,\Theta_g^T\right]\left(\frac{1+|\Theta_g|}{g_{\max}}\right)^2 \Theta_g^T \Theta_g\right] \\ &= -\mathrm{tr}\left[\left(B^T P \underline{e} \underline{u}^T \Psi^T(\underline{x})\right)^T \Theta_g\right] + \mathrm{tr}\left[B^T P \underline{e} \underline{u}^T \Psi^T(\underline{x})\,\Theta_g^T\right]\left(\frac{1+|\Theta_g|}{g_{\max}}\right)^2 \mathrm{tr}\left[\Theta_g^T \Theta_g\right] \\ &= -\mathrm{tr}\left[\left(B^T P \underline{e} \underline{u}^T \Psi^T(\underline{x})\right)^T \Theta_g\right] + \mathrm{tr}\left[B^T P \underline{e} \underline{u}^T \Psi^T(\underline{x})\,\Theta_g^T\right]\left(\frac{1+|\Theta_g|}{g_{\max}}\right)^2 |\Theta_g|^2\end{aligned} \tag{5.14}$$

Puisque $|\Theta_g| = g_{\max}$, nous obtenons

$$\dot{V}_g = \mathrm{tr}\left[B^T P \underline{e} \underline{u}^T \Psi^T(\underline{x})\,\Theta_g^T\right](2g_{\max} + g_{\max}^2) \leq 0$$

Ce qui produit $|\Theta_g| \leq g_{\max}$. Ainsi, nous avons $|\Theta_g| \leq g_{\max}$, $\forall t \geq 0$.

A partir de (5.13) nous remarquons que si $|\Theta_g|_{ij} = g_{\min}$, alors $\left[\dot{\Theta}_g\right]_{ij} \geq 0$; ainsi, nous avons que $|\Theta_g|_{ij} \geq g_{\min}$. □

2. En utilisant la même analyse, nous pouvons prouver que $|\Theta_f| \leq f_{\max}$, $\forall t \geq 0$.

5.5 Analyse de stabilité

Les propriétés de stabilité de la commande neuronale adaptative indirecte proposée sont résumées par le théorème suivant.

Théorème 5.2 : La commande neuronale adaptative indirecte du système non linéaire multivariable (5.1) vérifiant P1-P4, avec la loi de commande (5.8) vérifiant C1-C3, les lois d'ajustement (5.10)-(5.11) et (5.13), garantit ce qui suit :

1. $|\underline{e}| \in L_\infty$
2. $|\underline{u}| \in L_\infty$

Preuve :

1. Considérons la fonction de Lyapunov

$$V = \frac{1}{2}\underline{e}^T P \underline{e} + \frac{1}{2\gamma_1}\mathrm{tr}\left[\widetilde{\Theta}_f^T \widetilde{\Theta}_f\right] + \frac{1}{2\gamma_2}\mathrm{tr}\left[\widetilde{\Theta}_g^T \widetilde{\Theta}_g\right] \tag{5.15}$$

La dérivée de (5.15) le long de (5.9) fournie :

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q\underline{e} - \underline{e}^T PB\left[\widetilde{\Theta}_f \Phi\left(\underline{x}\right) + \widetilde{\Theta}_g \Psi\left(\underline{x}\right)\underline{u} + H\left(\underline{x}\right)\underline{u} + \underline{\omega}\left(\underline{x}\right)\right] + \frac{1}{\gamma_1}\mathrm{tr}\left[\dot{\widetilde{\Theta}}_f^T \widetilde{\Theta}_f\right] + \frac{1}{\gamma_2}\mathrm{tr}\left[\dot{\widetilde{\Theta}}_g^T \widetilde{\Theta}_g\right] \tag{5.16}$$

Ce qui peut être arrangée comme :

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q\underline{e} - \underline{e}^T PB\left[H\left(\underline{x}\right)\underline{u} + \underline{\omega}\left(\underline{x}\right)\right] + \frac{1}{\gamma_1}\mathrm{tr}\left[\left(\dot{\widetilde{\Theta}}_f^T - \gamma_1 \Phi\left(\underline{x}\right)\underline{e}^T PB\right)\widetilde{\Theta}_f\right] + \frac{1}{\gamma_2}\mathrm{tr}\left[\left(\dot{\widetilde{\Theta}}_g^T - \gamma_2 \Psi\left(\underline{x}\right)\underline{u}\underline{e}^T PB\right)\widetilde{\Theta}_g\right] \tag{5.17}$$

Alors, en utilisant (5.10)-(5.11) et le fait que $\dot{\widetilde{\Theta}}_f = -\dot{\Theta}_f$ ($\dot{\widetilde{\Theta}}_g = -\dot{\Theta}_g$), on peut montrer que le troisième et le quatrième termes dans (5.17) sont toujours ≤ 0.
Ainsi, (5.17) peut être écrite comme :

$$\dot{V} \leq -\frac{1}{2}\underline{e}^T Q\underline{e} - \underline{e}^T PB\left[H\left(\underline{x}\right)\underline{u} + \underline{\omega}\left(\underline{x}\right)\right] \tag{5.18}$$

De plus, (5.18) peut être bornée par :

$$\dot{V} \leq -\frac{1}{2}\lambda_{\min Q}\left|\underline{e}\right|^2 + \left|\underline{e}\right|\left|PB\right|\left(\left|H\left(\underline{x}\right)\right|\left|\underline{u}\right| + \left|\underline{\omega}\left(\underline{x}\right)\right|\right) \tag{5.19}$$

Alors, en utilisant (5.8) et le résultat du théorème 5.1, l'entrée de commande peut être bornée par :

$$\begin{aligned}\left|\underline{u}\right| &\leq \left|\left[\Theta_g \Psi\left(\underline{x}\right)\right]^{-1}\right|\left(\left|\Theta_f \Phi\left(\underline{x}\right)\right| + \left|\underline{v}\right| + \left|K\right|\left|\underline{e}\right|\right) \\ &\leq \frac{1}{g_{\min}}\left(f_{\max} + v_{\max} + \left|K\right|\left|\underline{e}\right|\right)\end{aligned} \tag{5.20}$$

Alors, l'utilisation de (5.20) et des hypothèses C1-C3 dans (5.19), donne :

$$\dot{V} \leq -\frac{1}{2}\lambda_{\min Q}\left|\underline{e}\right|^2 + \alpha_1\left|\underline{e}\right|^2 + \alpha_2\left|\underline{e}\right| \tag{5.21}$$

où

$$\alpha_1 = \frac{h_{\max}}{g_{\min}}\left|PBK\right| \tag{5.22}$$

$$\alpha_2 = \left|PB\right|\left(\frac{h_{\max}}{g_{\min}}\left(f_{\max} + v_{\max}\right) + \omega_{\max}\right) \tag{5.23}$$

Ainsi, (5.21) peut être arrangée comme :

$$\dot{V} \leq -\frac{1}{2}\left(\lambda_{\min Q} - 2\alpha_1\right)\left|\underline{e}\right|^2 + \alpha_2\left|\underline{e}\right| \tag{5.24}$$

Si la matrice Q est choisie telle que $\lambda_{\min Q} > 2\alpha_1$, alors $\dot{V} \leq 0$ en dehors de la région bornée définie par :

$$|\underline{e}| \leq \frac{2\alpha_2}{(\lambda_{\min Q} - 2\alpha_1)} \tag{5.25}$$

Ce qui implique que $|\underline{e}| \in L_\infty$.

2. L'utilisation du résultat (5.25) dans (5.20) produit :

$$|\underline{u}| \leq \frac{1}{g_{\min}} \left(f_{\max} + v_{\max} + |K| \frac{2\alpha_2}{(\lambda_{\min Q} - 2\alpha_1)} \right) \tag{5.26}$$

Ce qui implique que $|\underline{u}| \in L_\infty$.□

5.6 Remarques

1. Seuls les éléments diagonaux de $G(\underline{x})$ sont estimés et employés dans la conception des entrées de commande. En faisant ceci, nous évitons l'estimation des termes de couplage (considérés ici comme des perturbations) et la nécessité de calculer l'inverse de l'estimation de $G(\underline{x})$.
2. Bien que, les entrées de commande (5.8) soient présentées sous forme vectorielle, elles peuvent, dans la pratique, être calculées indépendamment puisque $\Theta_g \Psi(\underline{x})$ et K sont des matrices diagonales et aucune information n'est nécessaire des autres entrées.
3. A partit de (5.26), il peut être vu que des contraintes sur les entrées de commande, c.-à-d., $|u_i| \leq u_{i\max}$ peuvent être satisfaites en ajustant la matrice de gain K et les accélér ations désirées $v_{\max}$.

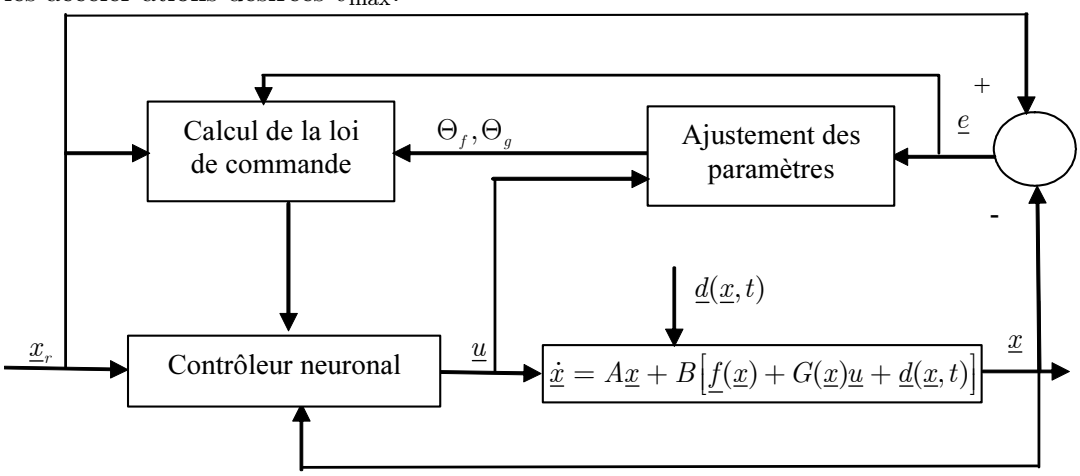


Figure 5.5. Commande neuronale adaptative indirecte proposée.

4. L'approche proposée (figure 5.5) est une commande indirecte implicite, puisque les paramètres du modèle implicitement identifié

$$\dot{\underline{\widehat{x}}} = A\underline{\widehat{x}} + B\left[\Theta_f \Phi\left(\underline{x}\right) + \Theta_g \Psi\left(\underline{x}\right)\underline{u}\right] \tag{5.27}$$

sont estimés et coïncident avec ceux du contrôleur. Ceci permet d'utiliser un seul réseau de neurones, et simplifie ainsi l'architecture de commande.

5.7 Résultats de simulation

Pour tester la commande neuronale proposée, nous allons considérer dans cette simulation, la commande d'un bras manipulateur à deux articulations (fig. 5.6). La dynamique de mouvement de ce manipulateur est donnée par [55] :

$$M(\underline{q})\ddot{\underline{q}} + \underline{c}(\underline{q},\dot{\underline{q}}) + \underline{g}(\underline{q}) + \underline{\tau}_c(\underline{q},\dot{\underline{q}}) + \underline{\tau}_d(\underline{q},\dot{\underline{q}}) = \underline{u} \tag{5.28}$$

avec $M(\underline{q}) \in \Re^{2\times 2}$ est la matrice d'inertie (qui est bornée et définie positive) ; $\underline{c}(\underline{q},\dot{\underline{q}}) \in \Re^{2\times 1}$ est le vecteur des forces centrifuges et de Coriolis ; $\underline{g}(\underline{q}) \in \Re^{2\times 1}$ est le vecteur des forces de gravité ; $\underline{\tau}_c(\underline{q},\dot{\underline{q}})$, $\underline{\tau}_d(\underline{q},\dot{\underline{q}}) \in \Re^{2\times 1}$ sont les vecteurs des couples des frottements ; $\underline{u} \in \Re^{2\times 1}$ est le vecteur des couples de commande ; et enfin $\underline{q},\dot{\underline{q}},\ddot{\underline{q}} \in \Re^{2\times 1}$ sont les vecteurs positions, vitesses et accélérations angulaires des deux articulations.

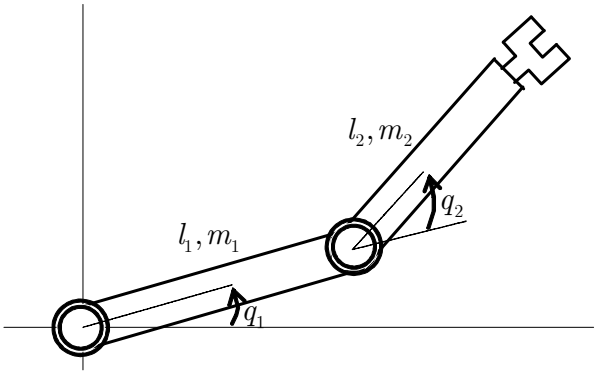


Figure 5.6. Robot à deux articulations.

Les expressions détaillées des termes intervenant dans (5.28), sont données par :

$$M\left(\underline{q}\right) = \begin{bmatrix} \left(m_1+m_2\right) l_1^2 & m_2 l_1 l_2 \cos(q_1-q_2) \\ m_2 l_1 l_2 \cos(q_1-q_2) & m_2 l_2^2 \end{bmatrix}$$

$$\underline{c}\left(\underline{q},\dot{\underline{q}}\right) = \begin{bmatrix} m_2 l_1 l_2 \sin(q_1-q_2)\dot{q}_2^2 \\ m_2 l_1 l_2 \sin(q_1-q_2)\dot{q}_1^2 \end{bmatrix}, \underline{g}\left(\underline{q}\right) = \begin{bmatrix} -g\left(m_1+m_2\right) l_1 \sin(q_1) \\ -g m_2 l_2 \sin(q_2) \end{bmatrix}$$

Les termes de perturbations dans (5.28) sont donnés par :

$$\underline{\tau}_c = \begin{bmatrix} \dot{q}_1 + \sin(3q_1) \\ 1.2\dot{q}_2 + 0.5\sin(2q_2) \end{bmatrix}, \ \underline{\tau}_d = \begin{bmatrix} 0.2\text{sign}(\dot{q}_1) \\ 0.1\text{sign}(\dot{q}_2) \end{bmatrix}$$

Les paramètres physiques du robot sont : $l_1 = l_2 = 0.432\text{m}$, $m_1 = 15.91\text{kg}$, $m_2 = 11.36\text{kg}$, et $g = 9.81\text{m/s}^2$.

Pour mettre la dynamique du robot, sous la forme (5.1), on réécrit la dynamique (5.28) comme suit :

$$\ddot{\underline{q}} = -M^{-1}(\underline{q})\left[\underline{c}(\underline{q},\dot{\underline{q}}) + \underline{g}(\underline{q}) + \underline{\tau}_c(\underline{q},\dot{\underline{q}})\right] + M^{-1}(\underline{q})\underline{u} - M^{-1}(\underline{q})\underline{\tau}_d(\underline{q},\dot{\underline{q}}) \tag{5.29}$$

De plus, prenons comme variable d'état le vecteur $\underline{x}^T = \begin{bmatrix} q_1 & \dot{q}_1 & q_2 & \dot{q}_2 \end{bmatrix}$, ce qui permet d'écrire (5.29) sous la forme (5.1) avec :

$$\underline{f}(\underline{x}) = \begin{bmatrix} f_1(\underline{x}) \\ f_2(\underline{x}) \end{bmatrix} := -M^{-1}(\underline{q})\left[\underline{c}(\underline{q},\dot{\underline{q}}) + \underline{g}(\underline{q}) + \underline{\tau}_c(\underline{q},\dot{\underline{q}})\right]$$

$$\begin{aligned} G(\underline{x}) &= \begin{bmatrix} g_{11}(\underline{q}) & g_{12}(\underline{q}) \\ g_{21}(\underline{q}) & g_{22}(\underline{q}) \end{bmatrix} \\ &: = M^{-1}(\underline{q}) = \begin{bmatrix} \frac{1}{l_1 l_2(m_1+m_2-m_2\cos^2(q_1-q_2))} & -\frac{\cos(q_1-q_2)}{l_1 l_2(m_1+m_2-m_2\cos^2(q_1-q_2))} \\ -\frac{\cos(q_1-q_2)}{l_1 l_2(m_1+m_2-m_2\cos^2(q_1-q_2))} & \frac{m_1+m_2}{l_2^2 m_2(m_1+m_2-m_2\cos^2(q_1-q_2))} \end{bmatrix} \end{aligned}$$

$$\underline{d}(\underline{x}) = \begin{bmatrix} d_1(\underline{x}) \\ d_2(\underline{x}) \end{bmatrix} := -M^{-1}(\underline{q})\underline{\tau}_d\left(\underline{q},\dot{\underline{q}}\right)$$

et

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

Le problème de commande peut être énoncé comme suit : Pour une trajectoire désirée (bornée) pour les deux articulations du robot $\underline{q}_r^T = \begin{bmatrix} q_{r1} & q_{r2} \end{bmatrix}$, avec la première dérivée $\dot{\underline{q}}_r$ et la deuxième dérivée $\ddot{\underline{q}}_r$ bornées, concevoir une loi de commande de la forme (5.8) qui permet aux deux articulations du robot de suivre cette trajectoire désirée.

La conception de la commande adaptative neuronale pour le robot manipulateur est résumée par les étapes suivantes :

1. Construire 4 réseaux de neurones pour approximer les fonctions non linéaires $f_1(\underline{x}), f_2(\underline{x}), g_{11}(\underline{x})$ et $g_{22}(\underline{x})$. A cette fin, les intervalles de variations des variables $x_1, x_3 \in [-\pi, \pi]\,(\text{rad})$ et $x_2, x_4 \in [-2\pi, 2\pi]\,(\text{rad/sec})$ sont divisées en 3 sous-domaines.

On peut réduire la complexité des réseaux construits, en remarquant que, les fonctions non linéaires à approximer ne dépendent que de certaines variables, à savoir, $f_1(x_1, x_3, x_4)$, $f_2(x_1, x_2, x_3)$, $g_{11}(x_1, x_3)$ et $g_{22}(x_1, x_3)$.
Les réseaux RBF conçus (avec des fonctions d'activation gaussiennes) sont comme suit :

$$\underline{\theta}_{f_1}^T \underline{\phi}_1(x_1, x_3, x_4) \rightarrow f_1(x_1, x_3, x_4) \tag{5.30}$$
$$\underline{\theta}_{f_2}^T \underline{\phi}_2(x_1, x_2, x_3) \rightarrow f_2(x_1, x_2, x_3) \tag{5.31}$$
$$\underline{\theta}_{g_1}^T \underline{\psi}_1(x_1, x_3) \rightarrow g_{11}(x_1, x_3) \tag{5.32}$$
$$\underline{\theta}_{g_2}^T \underline{\psi}_2(x_1, x_3) \rightarrow g_{22}(x_1, x_3) \tag{5.33}$$

où $\underline{\theta}_{f_1}, \underline{\theta}_{f_2} \in \Re^{27\times 1}$ et $\underline{\theta}_{g_1}, \underline{\theta}_{g_2} \in \Re^{9\times 1}$ sont les paramètres ajustables des réseaux de neurones. Cette approche n'est pas optimale, mais elle a le mérite de réduire la complexité de l'algorithme d'ajustement et le temps d'exécution.

2. La loi de commande est conçue comme dans (5.8) avec les réseaux de neurones adaptatifs définis dans (5.30)-(5.33). Le gain est défini par

$$K = \begin{bmatrix} 4 & 4 & 0 & 0 \\ 0 & 0 & 4 & 4 \end{bmatrix}$$

3. Pour le choix de $Q = 100 I_4$ et les solutions de (5.12) nous obtenons :

$$P = \begin{bmatrix} 112.5 & 12 & 0 & 0 \\ 12 & 15 & 0 & 0 \\ 0 & 0 & 112.5 & 12 \\ 0 & 0 & 12 & 15 \end{bmatrix}$$

4. En analysant la dynamique du robot, les bornes suivantes sont fixées $g_{\min} = 13.75$, $g_{\min} = 0.73$ et $f_{\max} = 20$.
5. Les paramètres des réseaux de neurones sont ajustés en utilisant (5.10)-(5.11) et (5.13) avec $\gamma_1 = 0.001, \gamma_2 = 0.0001$.

Dans la simulation, les paramètres des réseaux de neurones sont initialisés par $\left|\underline{\theta}_{g_1}(0)\right| = 0.5, \left|\underline{\theta}_{g_2}(0)\right| = 1$ et $\left|\underline{\theta}_{f_1}(0)\right| = \left|\underline{\theta}_{f_2}(0)\right| = 0$. Les états initiaux, dans toutes les simulations, sont $\underline{x}^T(0) = \begin{bmatrix} 0.5 & 2 & -0.5 & 1 \end{bmatrix}$.

5.7.1 Régulation

La première simulation concerne la régulation des positions articulaires vers des valeurs désirées constantes $q_{r1} = q_{r2} = 3rad$.

Le premier test est réalisé pour des conditions de fonctionnement nominales, c.-à-d., aucun changement des paramètres et pas de perturbations. Comme représenté sur la

figure 5.7, les positions des articulations montrent une bonne performance, et aucune erreur n'est remarquée en régime permanent.

Le deuxième test (figure 5.8) montre la performance de régulation sous les effets des perturbations des couples de frottements. Il est clair que ces perturbations affectent peu la performance de régulation et une petite erreur est présente en régime permanant. La commande adaptative neuronale réalise une bonne compensation des effets des perturbations.

Le troisième test (figure 5.9) montre la performance de régulation sous l'effet d'un changement de la charge utile introduit à $t = 20$sec lorsque m_2 passe de 11.36kg à 20kg. Puis revient à 11.36kg à $t = 40$sec. Il est clair qu'après une brève période transitoire, la commande adaptative neuronale compense cette perturbation et rétablie la performance de régulation.

Dans le quatrième test (figure 5.10), le robot est sujet aux effets combinés des frottements et de la variation de charge. Cette situation est rapidement prise en compte par la commande neuronale et la performance de régulation est maintenue.

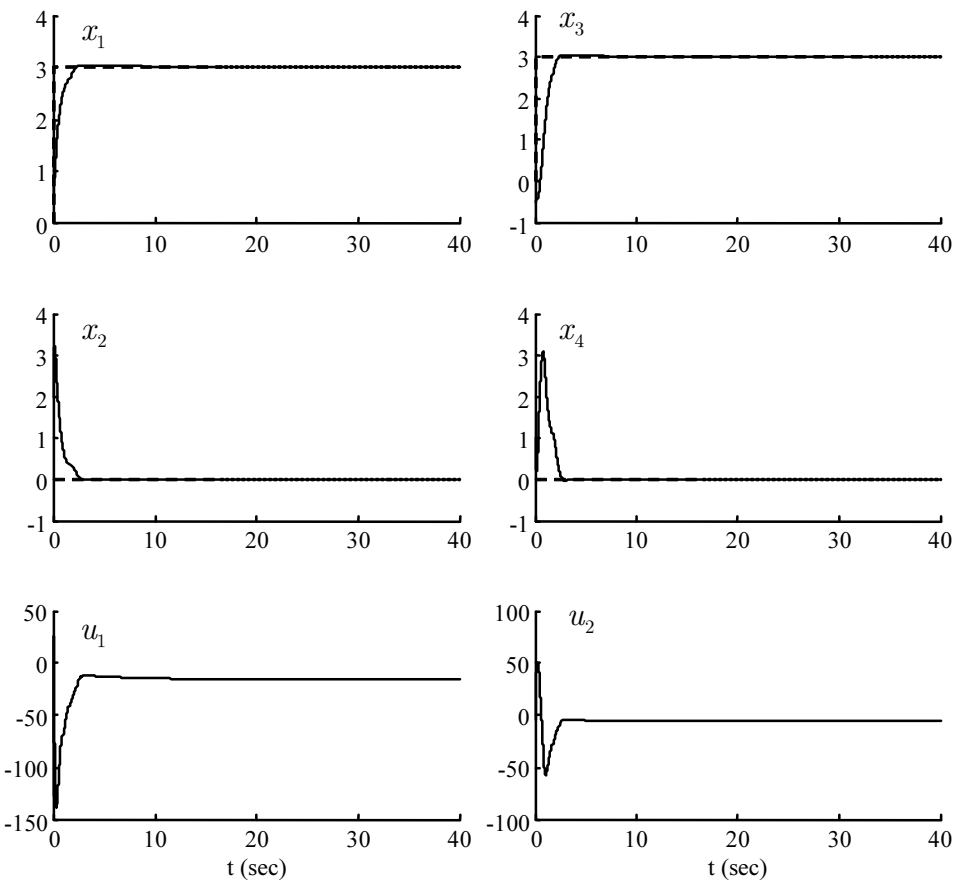


Figure 5.7. Performance de régulation (cas nominal).

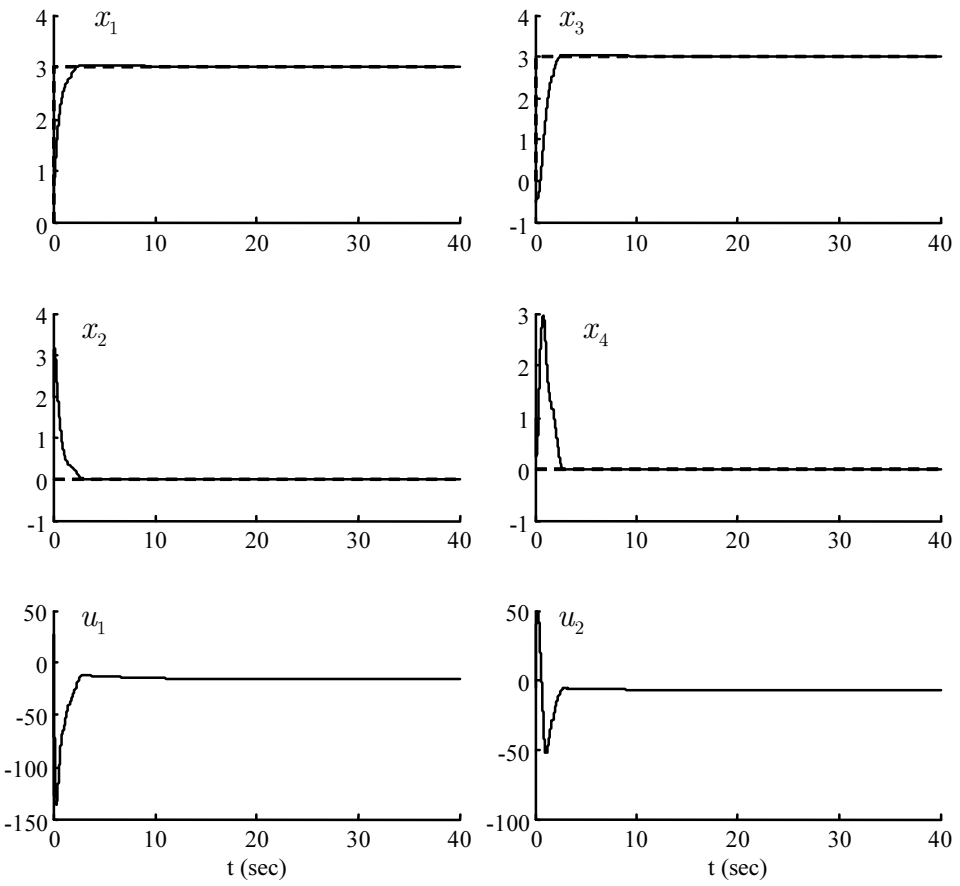


Figure 5.8. Performance de régulation (sous les effets des frottements).

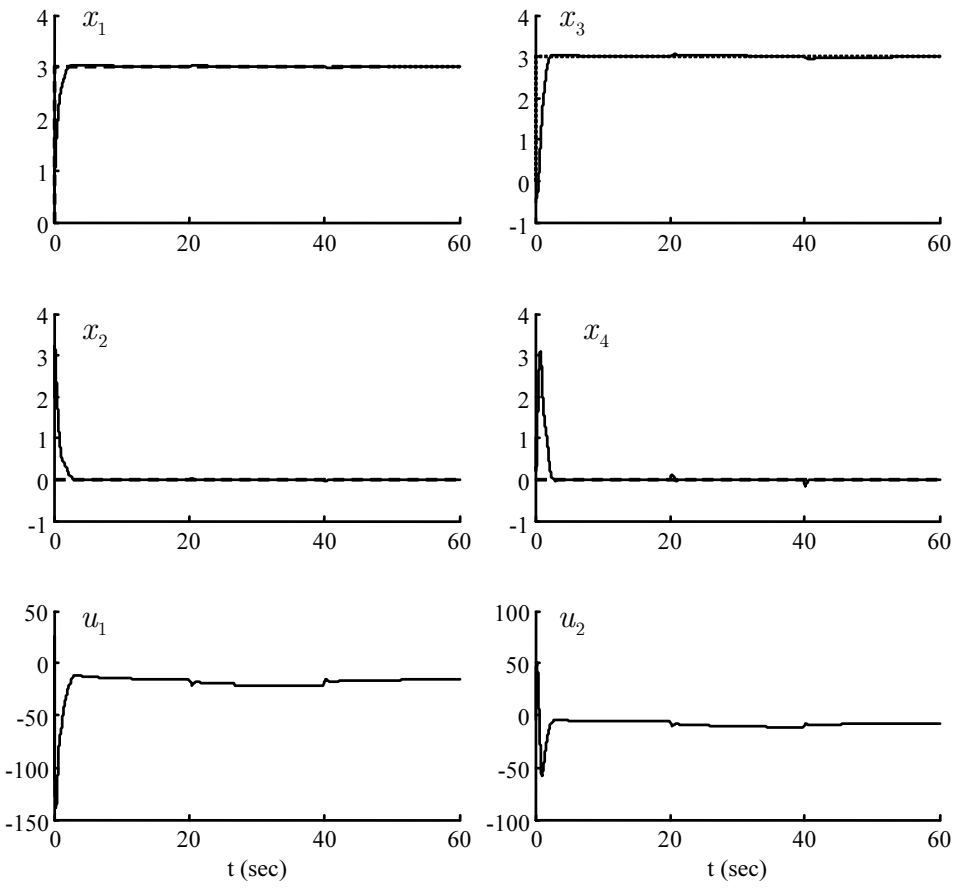


Figure 5.9. Performance de régulation (sous l'effet de la variation de charge).

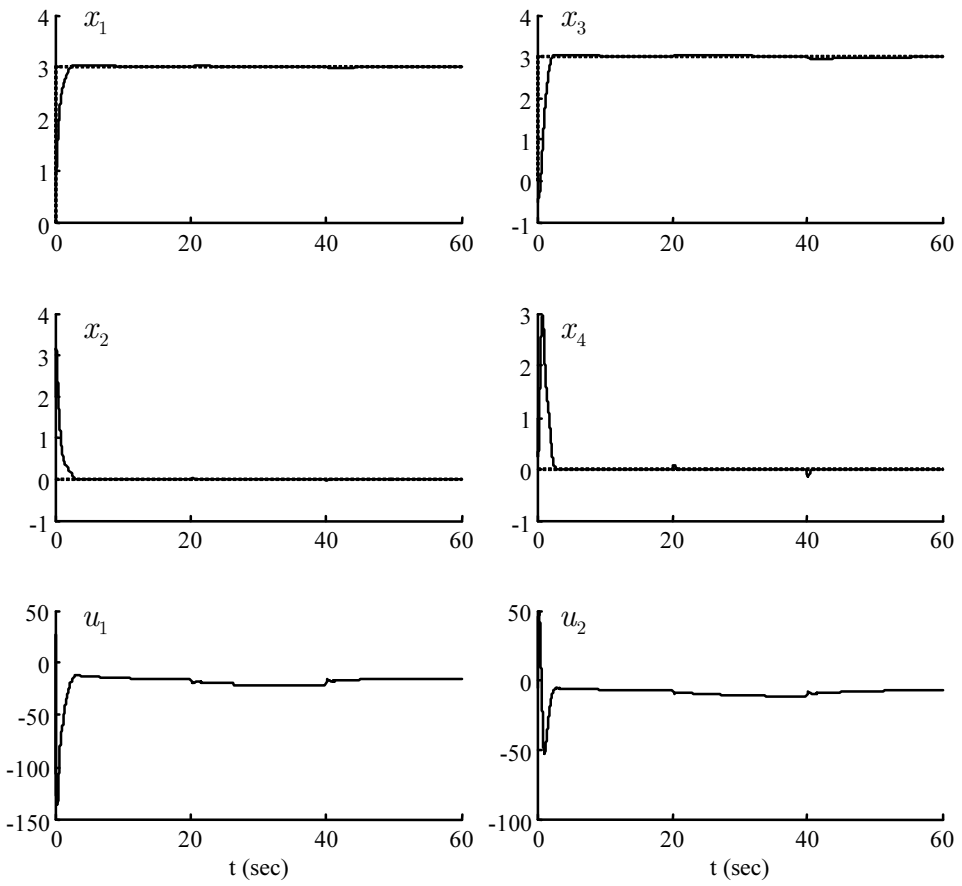


Figure 5.10. Performance de régulation (sous les effets des frottements et la variation de charge).

5.7.2 Poursuite

La deuxième simulation concerne l'évaluation des performances en poursuite de la commande neuronale pour des trajectoires désirées $q_{r1} = \sin(t)+\sin(2t)$ et $q_{r2} = \cos(t)+\cos(2t)$.

Le premier test concerne les performances de poursuite dans le cas nominal. Comme il est illustré sur la figure 5.11, les positions des articulations montrent une bonne performance de poursuite et une petite erreur est remarquée en régime établi.

Le deuxième test (figure 5.12) montre les performances de poursuite sous les effets des perturbations des couples de frottements. Il est clair que les erreurs de poursuite sont acceptables, et que la commande neuronale compense ces perturbations.

Le troisième test (figure 5.13) montre les performances de poursuite sous l'effet d'un changement de la charge utile introduit à $t = 20$sec lorsque m_2 passe de 11.36kg à 20kg. Puis revient à 11.36kg à $t = 40$sec. La commande neuronale compense cette perturbation avec un effort de commande additionnel.

Dans le quatrième test (figure 5.14), nous ajoutons aux effets des frottements, un changement de la charge. Cette variation affecte essentiellement les couples développés pour compenser la masse additionnelle, et une petite erreur est remarquée.

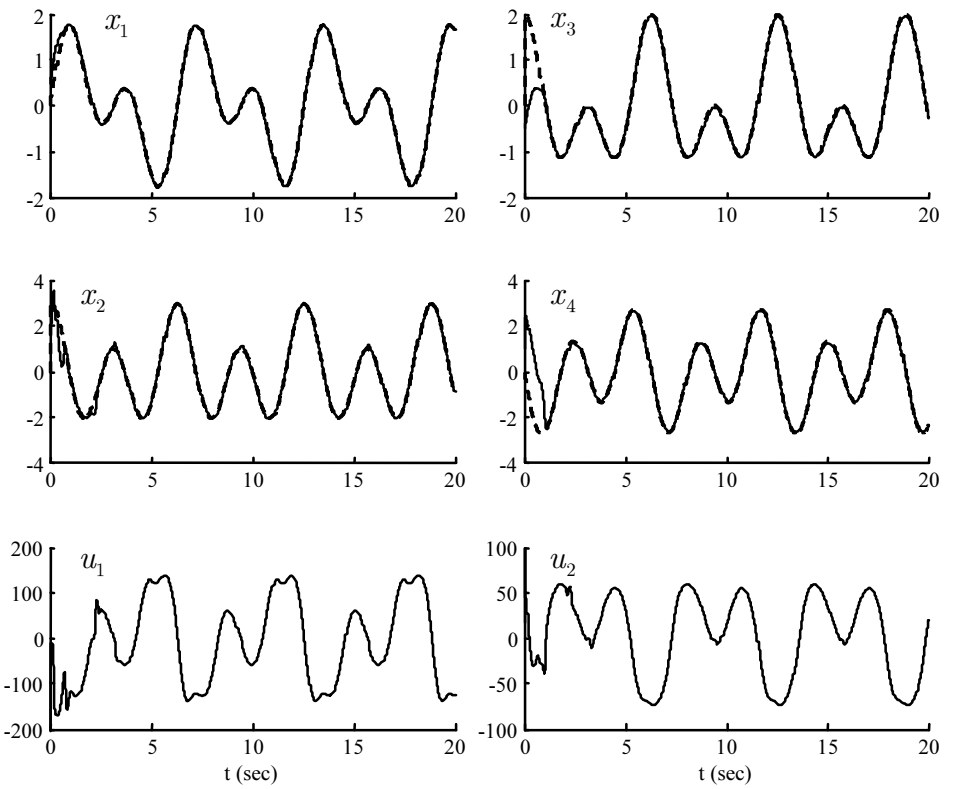


Figure 5.11. Performance de poursuite (cas nominal).

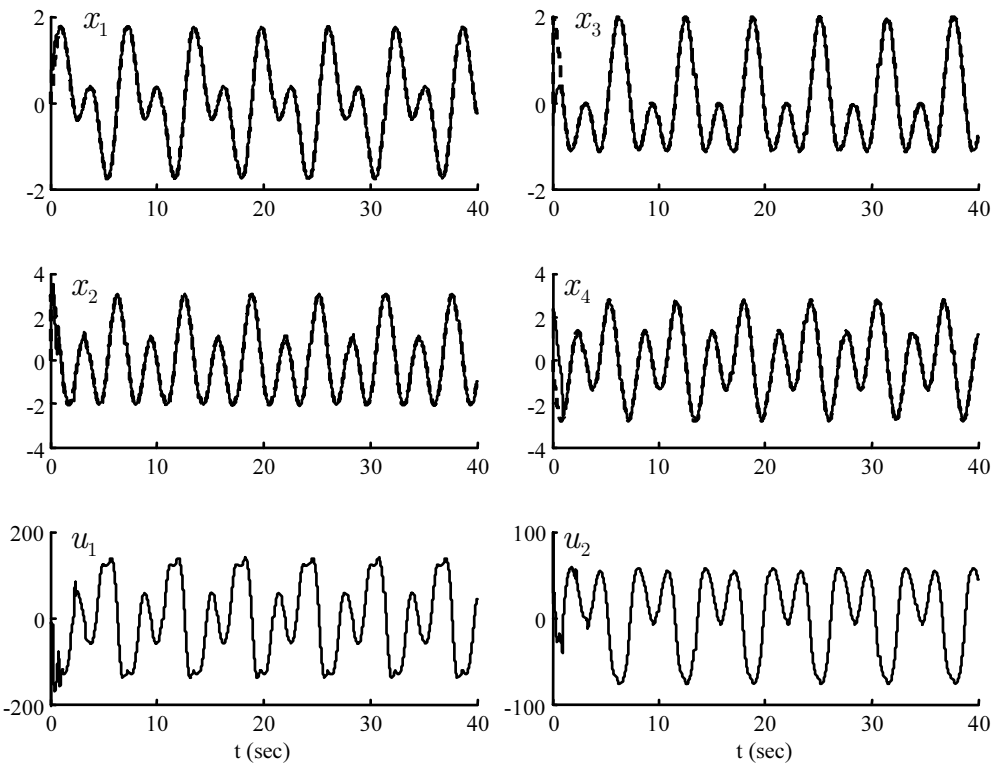


Figure 5.12. Performance de poursuite (sous les effets des frottements).

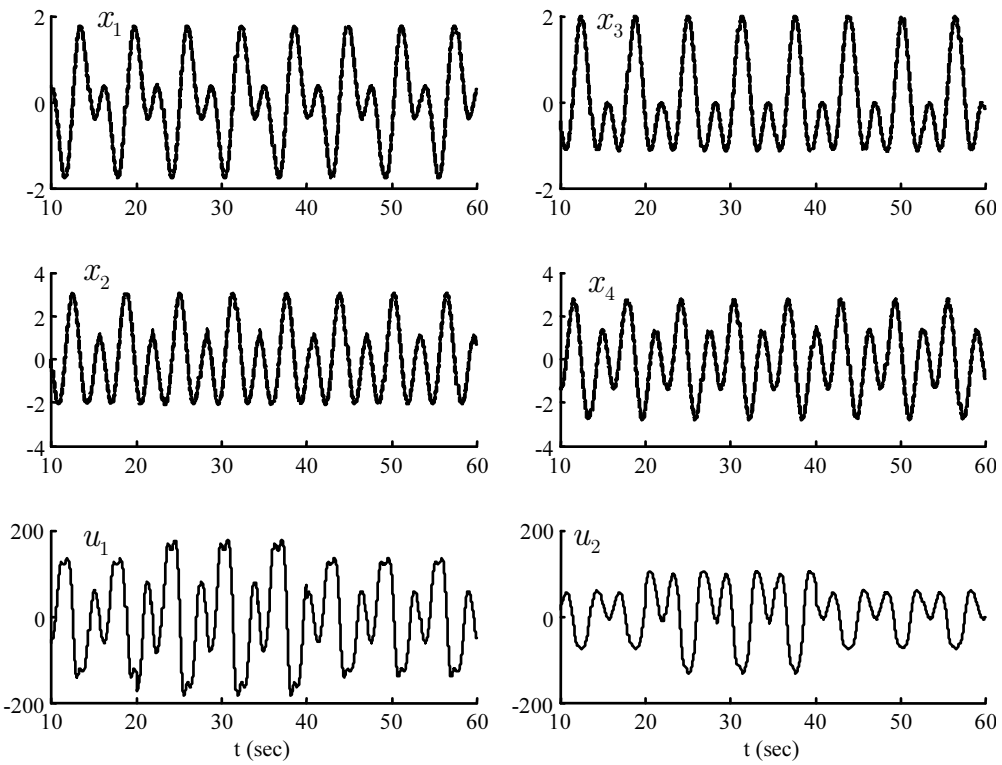


Figure 5.13. Performance de poursuite (sous l'effet de la variation de charge).

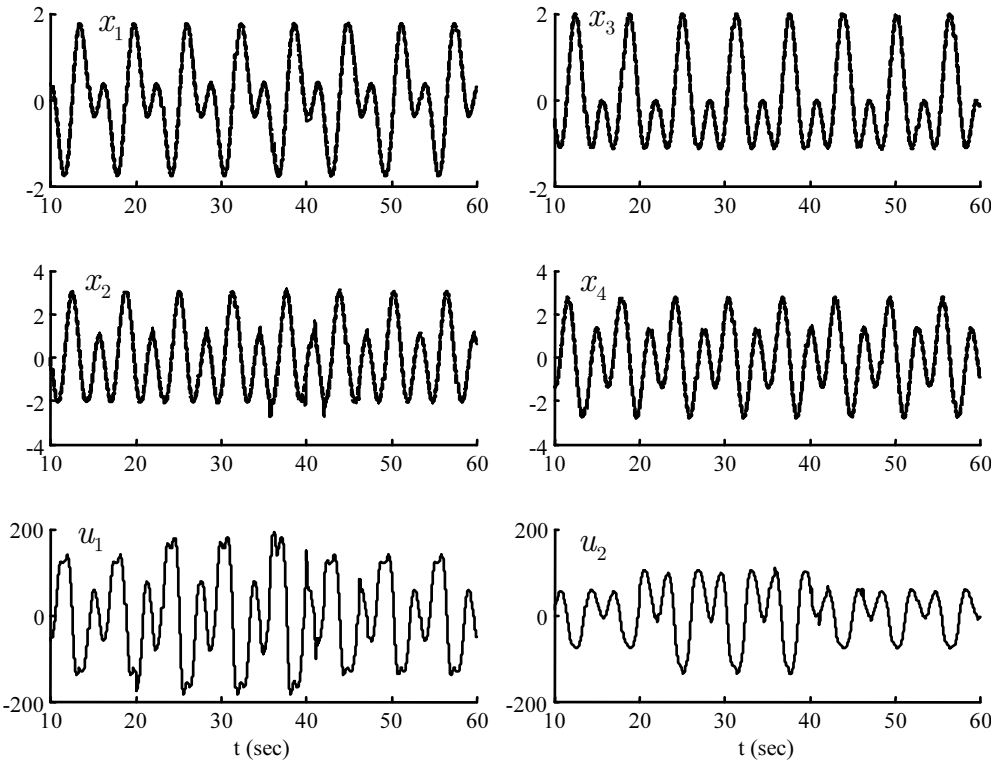


Figure 5.14. Performance de poursuite (sous les effets des frottements et la variation de charge).

5.8 Conclusion

Dans ce chapitre, nous avons développé une approche neuronale pour la commande des systèmes non linéaires multivariables incertains. Cette approche est du type indirect, où une estimation implicite du modèle du système non linéaire est utilisée pour concevoir une loi de commande decoulpée. L'analyse de Lyapunov nous a permis d'établir la stabilité de la boucle de commande. Cette commande neuronale adaptative, bien que simple, a montré son efficacité pour la commande de procédés physiques très complexes, comme démontré par simulation sur le cas du robot manipulateur.

Chapitre 6

Conclusion

6.1 Conclusions

La contribution de la recherche présentée dans cette thèse a été le développement d'algorithmes de commande adaptative pour le contrôle des systèmes dynamiques non linéaires sujets à des incertitudes et des perturbations externes. Ces algorithmes sont basés sur des réseaux de neurones de type MLP et RBF, avec une seule couche cachée, pour réaliser les approximations nécessaires. Les lois d'ajustement utilisées ont permis d'assurer la bornitude des paramètres ajustables. L'analyse de Lyapunov a permis de garantir la stabilité de la boucle de commande, sous certaines hypothèses, même en présence des incertitudes et des perturbations externes.

La première contribution a été l'extension de la commande MRAC aux systèmes non linaires perturbés. Il a été montré que pour une large classe de systèmes non linéaires, il est possible d'améliorer les performances de précision et de robustesse avec cette version neuronale de la commande MRAC. L'algorithme adaptatif proposé enlève la nécessité d'un terme de robustification, qui peut éventuellement entraîner une contre réaction à gain élevé. Cette approche a été implémentée à la fois avec un réseau MLP à paramètrisation non linéaire et avec un réseau RBF à paramètrisation linéaire. De plus, cette technique peut être aisément étendue au cas des systèmes non linéaires multivariables decouplés.

La deuxième contribution de cette thèse était d'étendre la commande MRAC neuronale développée au chapitre 3 au cas des systèmes non linéaires dont le vecteur d'état n'est pas accessible à la mesure. L'architecture de commande proposée combine un retour d'état de type MRAC et un observateur linéaire pour estimer les états non mesurables. L'utilisation de l'observateur linéaire permet d'alléger la structure de commande et de simplifier l'étude de la stabilité. Un filtrage approprié a permis de rendre le transfert erreur – incertitudes SPR. L'analyse de Lyapunov nous a permis d'établir la stabilité de la boucle de commande augmentée par l'observateur. Les tests de simulation avec des MLP et RBF ont permis d'évaluer l'efficacité de cette technique. L'extension aux systèmes multivariables decouplés est éventuellement possible.

La troisième contribution a été l'élaboration d'une commande neuronale decouplée pour le contrôle des systèmes non linéaires multivariables incertains. La technique proposée est du type indirect. Les paramètres du contrôleur adaptatif sont calculés à travers une estimation implicite d'un modèle d'identification pour le système non linéaire. La boucle de commande est stable sans aucune action de commande supplémentaire. Un modèle d'un bras manipulateur a permis d'évaluer les aptitudes de cette approche en termes de régulation et de poursuite, et ceci sous l'effet des perturbations dues aux frottements et à la variation des paramètres.

6.2 Recommandations pour de futures recherches

Ce qui suit présente des recommandations pour de futures recherches.

1. A travers les travaux présentés dans cette thèse, nous avons montré qu'il est possible de contrôler l'amplitude du signal de commande en ajustant un certain nombre de paramètres de conception. Cependant, dans les systèmes physiques, il peut être question de contraintes sur les valeurs que peuvent prendre les états du système ; et ceci pour des questions de sécurité ou de bon fonctionnement des systèmes complexes. Alors, il peut être utile d'inclure ces contraintes sur les états du système dans les hypothèses de conception, et d'en prendre compte lors de l'analyse de la stabilité et des performances.
2. Nous avons considéré, dans ce travail, la classe des systèmes non linéaires continus ; et ceci est justifié par le fait que la plupart des procédés physiques possèdent une dynamique continue. Cependant, du point de vue implémentation, nous avons toujours fait appel à des procédures numériques et à des calculateurs. Il est souvent suffisant, de discrétiser les lois de commande continues, et les performances obtenues sont satisfaisantes pour des périodes d'échantillonnage assez petites. Dans d'autres applications, il sera nécessaire de reconsidérer le problème dans le cadre de la commande discrète ; ce qui peut changer certaines étapes de la conception, comme les lois d'ajustement ou les observateurs.
3. Un autre type de systèmes rencontrés beaucoup plus dans les environnements de production sont les systèmes hybrides. Ces systèmes sont caractérisés par des variables d'états continues et des variables d'états discrètes. Les variables d'états discrètes gouvernent le changement de dynamique que subit le système lors du passage d'un type de fonctionnement à un autre. Donc, pour les systèmes hybrides non linéaires, il est envisageable d'utiliser des techniques de commande multi-contrôleurs ou multi-modèles pour gérer toutes les dynamiques possibles. Ceci entraîne aussi de nouvelles difficultés du point de vue conception et analyse de stabilité.

Bibliographie

[1] Slotine, J. and Li, W., Applied Nonlinear Control, chap. 4, Prentice Hall, New Jersey, 1991.

[2] M. Krstic, I. Kanellakopoulos, P. Kokotovic, Nonlinear and Adaptive Control Design, Wiley, New York, 1995.

[3] Isidori, A., Nonlinear Control Systems, Springer, Berlin, 1995.

[4] Khalil, H., Nonlinear Systems, Prentice Hall, New Jersey, 3rd ed., 2002.

[5] S. Devasia, D. Chen and B. Paden, "Nonlinear inversion-based output tracking", IEEE Trans. Automatic Control, vol. 41, no. 7, pp. 930-942, July, 1996.

[6] J. Huang, "Asymptotic tracking of a nonminimum phase nonlinear system with nonhyperbolic zero dynamics", IEEE Trans. Automatic Control, vol. 45, no. 3, pp. 542-546, March, 2000.

[7] A. Astolfi, D. Karagiannis and R. Ortega, Nonlinear and Adaptive Control with Applications, Springer, Berlin, 2008.

[8] S. S. Sastry and A. Isidori, "Adaptive control of linearizable systems", IEEE Trans. Automatic Control, vol. 34, no. 11, pp1123-1131, Nov. 1989.

[9] G. Cybenko, Approximations by superpositions of a sigmoidal function, Math. Signals Syst., vol. 2, pp. 303–314, 1989.

[10] K. Hornik, M. Stinchcombe, and H. White, Multilayer Feedforward Networks are Universal Approximators, Neural Networks 2, 359-366, 1989.

[11] M. E. Cotter, The Stone-Weierstrass theorem and its applications to neural nets, IEEE Trans. Neural Networks, vol. 1, pp. 290-295, 1990.

[12] J. Park and I. W. Sandberg, Universal approximation using radial basis function networks, Neural Comput., vol. 3, no. 2, pp. 246–257, 1991.

[13] A. R. Barron, Universal approximation bounds for superpositions of a sigmoid function, IEEE Trans. Inform. Theory 39, no. 3, 930-945, 1993.

[14] S. Ferrari and R. F. Stengel, Smooth function approximation using neural networks, IEEE Trans. Neural Networks, vol. 16, no. 1, pp. 24-38, Jan. 2005.

[15] F. Girosi and T. Poggio, Networks and the best approximation property, Artificial Intelligence Lab. Memo no. 1164, 1989.

[16] K.S. Narendra and K. Parthasarathy, Identification and control of dynamical systems using neural networks, IEEE Transactions on Neural Networks 1, 4-27, 1990.

[17] M.M. Polycarpou and P.A. Ioannou, Modelling, Identification and Stable Adaptive Control of Continuous-Time Nonlinear Dynamical Systems Using Neural Networks, Proceedings of the American Control Conference, 36-40, 1992.

[18] R. Sanner and J.J. Slotine, Gaussian networks for direct adaptive control, IEEE Transactions on Neural Networks 3, no. 6, 837-864, 1992.

[19] N. Sadegh, A perceptron network for functional identification and control of nonlinear systems, IEEE Transactions on Neural Networks 4, no. 6, 982-988, 1993.

[20] F.C. Chen and H.K. Khalil, Adaptive Control of Nonlinear Systems using Neural Networks, International Journal of Control 55, no. 6, 1299-1317, 1992.

[21] F.C. Chen and C.C. Liu, Adaptively controlling nonlinear continuous-time systems using multilayer neural networks, IEEE Trans. Autom. Contr. 39, no. 6, 1306-1310, 1994.

[22] F.L. Lewis, A. Yesildirek, and K. Liu, Multilayer Neural-Net Robot Controller with Guaranteed Tracking Performance, IEEE Transactions on Neural Networks 7, no. 2, 1-12, 1996.

[23] F.L. Lewis, A. Yesildirek, and K. Liu, Neural net robot controller : structure and stability proofs, Journal of Intelligent and Robotic System 12, 277-299, 1996.

[24] J. Leitner, A. J. Calise and J.V. R. Prasad, "Analysis of Adaptive Neural Networks for Helicopter Flight Controls," Journal of Guidance, Control, and Dynamics, Vol. 20, No. 5, pp. 972–979, 1997.

[25] A.J. Calise and R.T. Rysdyk, Nonlinear Adaptive Flight Control using Neural Networks, IEEE Control System Magazine 18, no. 6, 14-25, 1998.

[26] Z. Zeng and J.Wang, Advances in Neural Network Research and Applications, , Springer, Berlin, 2010.

[27] K. Diamantaras, W. Duch and L. S. Iliadis (Eds.), Artificial Neural Networks – ICANN 2010, 20th International Conference Proceedings, Thessaloniki, Greece, September 15-18, 2010.

[28] P. A. Ioannou and B. Fidan, Adaptive Control – Tutorial, Philadelphia, SIAM, 1996.

[29] J. T. Spooner, M. Maggiore, R. Ordonez, K. M. Passino, Stable Adaptive Control and Estimation for Nonlinear Systems : Neural and Fuzzy Approximator Techniques, Wiley, New York, 2002.

[30] M. Powell. Radial basis functions for multivariable interpolation : A review. In J. Mason and M. Cox, editors, Algorithms for Approximation of Functions and Data, Oxford University, Oxford, UK, 1987.

[31] S. Chen, S. Billings, C. Cowan, and P. Grant. Practical identification of NARMAX models using radial basis functions. International Journal of Control, 52(6) : 1327-1350, 1990.

[32] S. Chen, S. Billings, and P. Grant. Non-linear system identification using neural networks. International Journal of Control, 51 :1191-1214, 1990.

[33] S. Chen, S. Billings, and P. Grant. Recursive hybrid algorithm for non-linear system identification using radial basis function networks. International Journal of Contml, 55(5) : 105 1-1070, 1992.

[34] D. G. Luenberger, Linear and nonlinear programming, Addison- Wesley, Reading, MA, 2008.

[35] H.P. Whitaker, J. Yamron, and A. Kezer, Design of model reference adaptive control systems for aircraft, Instrumentation Laboratory, M.I.T., 1958.

[36] L.P. Grayson, Design via Lyapunov's Second Method, Proc. of the 4th JACC, 1963.

[37] R.L. Butchart and B. Shackcloth, Synthesis of Model Reference Adaptive Systems by Lyapunov's Second Method, Proc. IFAC Symposium on Adaptive Control, 145-152, 1965.

[38] P.C. Parks, Lyapunov redesign of model reference adaptive control systems, IEEE Trans. Autom. Contr. 11, 362-367, 1966.

[39] R.M. Dressler, An Approach to Model-Reference Adaptive Control Systems, IEEE Trans. Autom. Contr. 12, no. 1, 75-80, 1967.

[40] R.V. Monopoli, Model reference adaptive control with an augmented error signal, IEEE Trans. Autom. Contr. 19, 474-484, 1974.

[41] R.V. Monopoli and C.C. Hsing, Parameter Adaptive Control of Multivariable Systems, International Journal of Control 22, no. 3, 313-327, 1975.

[42] M. Cannon and J. J. E. Slotine, "Space-frequency localized basis function networks for nonlinear estimation and control," Neurocomputing, vol. 9, no 3, pp293-342, Dec. 1995.

[43] G.A. Rovithakis, Nonlinear Adaptive Control in the Presence of Unmodelled Dynamics using Neural Networks, Proceedings of IEEE Conf. Dec. and Contr., 2150-2155, 1999.

[44] J.T. Spooner and K.M. Passino, Stable adaptive control using fuzzy systems and neural networks, IEEE Trans. Neural Net., Vol. 4, No. 3, pp. 339–359, 1996..

[45] C-T Chen, Linear system theory and design, Oxford University Press, 3rd ed, UK, 1999.

[46] G. Besançon (Ed.), Nonlinear Observers and Applications, Springer-Verlag Berlin Heidelberg 2007.

[47] J.Y. Choi and J. Farrell, Adaptive Observer for a Class of Nonlinear Systems Using Neural Networks, Proceedings of IEEE Intl. Symposium on Intelligent Control/Intelligent Systems and Semiotics, 114-119, 1999.

[48] S. S. Ge, C. C. Hang and T. Zhang, "Adaptive Neural Network Control of Nonlinear Systems by State and Output Feedback", IEEE Trans. on Systems, Man and Cybernetics (Part B), vol. 29, no. 6, pp. 818-828, Dec. 1999.

[49] H. A. Talebi, F. Abdollahi, R. V. Patel, K. Khorasani, Neural Network-Based State Estimation of Nonlinear Systems, Springer, New York, 2010.

[50] S. Seshagiri and H.K. Khalil, Output Feedback Control of Nonlinear Systems Using RBF Neural Networks, IEEE Trans. Neural Networks 11, no. 1, 69-79, 2000.

[51] F. Nardi, Neural Network based Adaptive Algorithms for Nonlinear Control , Ph.D. thesis, School of Aerospace Engineering, Georgia Institute of Technology, 2000.

[52] N. Hovakimyan, F. Nardi, and A. Calise, "A Novel Error Observer based Adaptive Output Feedback Approach for Control of Uncertain Systems," IEEE Transactions on Automatic Control , Vol. 47, No. 8, pp. 1310–1314, 2002.

[53] Boyd, S., L. El Ghaoui, E. Feron, V. Balakrishnan, Linear Matrix Inequalities in Systems and Control Theory, SIAM books, 1994.

[54] D. Psichogios, L Ungar, Direct and indirect model based control using neural networks, Industrial and Engineering Chemistry Research, 30, 12, 2564-2573, 1991.

[55] R. Ortega and M. Spong, Adaptive motion control of rigid robots : A tutorial, Proc. 27th Conf. on Decision and Control CDC, Austin, Texas, , pp. 1575-1584, Dec 1988.

Annexe

Les définitions suivantes peuvent être retrouvées dans la plupart des livres de commande (voir p. ex. [1-4, 28]).

Vecteurs :
Soit le vecteur $\underline{x}^T = \begin{bmatrix} x_1 & x_2 & .. & x_n \end{bmatrix}$, alors sa norme Euclidienne est définie par :

$$|\underline{x}| = \sqrt{\underline{x}^T \underline{x}} = \sqrt{x_1^2 + x_2^2 + .. + x_n^2}$$

La norme Euclidienne possède les propriétés suivantes :

1. $|\underline{x}| \geq 0$
2. Soient le vecteur $\underline{x}$ et le scalaire a alors $|a\underline{x}| \leq |a|\,|\underline{x}|$.
3. Soient les vecteurs $\underline{x}$ et $\underline{y}$ alors $\left|\underline{x} + \underline{y}\right| \leq |\underline{x}| + \left|\underline{y}\right|$.

Matrices :
Soit la matrice $A \in \Re^{n \times n}$ tel que :

$$A = \begin{bmatrix} a_{11} & a_{12} & .. & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

Alors nous avons les propriétés suivantes :

1. Si $A^T = A$ alors A est dite symétrique.
2. La matrice $A \in \Re^{n \times n}$ est dite de Hurwitz (stable) si les racines de son polynôme caractéristique (ses valeurs propres) sont toutes à partie réelle négative, soit : $\mathrm{Re}\,[\lambda_{iA}] < 0$.
3. La trace de la matrice A est définie par :

$$\mathrm{tr}(A) = \sum_{i=1}^{n} a_{ii}$$

4. Soient les matrices $A, B \in \Re^{n \times n}$ alors $\mathrm{tr}(AB) = \mathrm{tr}(BA)$

5. Soient les vecteurs $\underline{x}$ et $\underline{y} \in \Re^{n\times 1}$ alors $\mathrm{tr}(\underline{y}\underline{x}^T) = \underline{x}^T\underline{y}$.
6. La norme-2 ou norme de Frobenius de la matrice A est définie par :

$$|A| = \sqrt{\lambda_{\max(A^TA)}}$$

où $\lambda_{\max(A^TA)}$ est la plus grande valeur propre de A^TA.

7. Soient les matrices $A, B \in \Re^{n\times n}$ et le vecteur $\underline{x} \in \Re^{n\times 1}$ alors.
   (a) $|A+B| \leq |A| + |B|$
   (b) $|AB| \leq |A|\,|B|$
   (c) $|A\underline{x}| \leq |A|\,|\underline{x}|$
8. Soient la matrice $P \in \Re^{n\times n}$ et le vecteur $\underline{x} \in \Re^{n\times 1}$ alors.
   (a) P est dite définie positive ($P > 0$) si $\underline{x}^TP\underline{x} > 0, \forall \underline{x} \neq \underline{0}$, ou d'une manière équivalente si tous ses mineurs principaux sont positifs.
   (b) Si $P > 0$ alors, nous avons l'inégalité de Rayleigh-Ritz :

$$\lambda_{\min P}\,|\underline{x}|^2 \leq \underline{x}^TP\underline{x} \leq \lambda_{\max P}\,|\underline{x}|^2$$

   où $\lambda_{\min P}$, $\lambda_{\max P}$ sont, respectivement, la plus grande et la plus petite valeurs propres de P.
   (c) La matrice $Q \in \Re^{n\times n}$ est dite définie négative ($Q < 0$) si $\underline{x}^TQ\underline{x} < 0, \forall \underline{x} \neq \underline{0}$, ou d'une manière équivalente si tous ses mineurs principaux sont négatifs.

Continuité

Soit un sous-ensemble $S \in \Re^n$, une fonction $f(\underline{x}) :\ S \to \Re$ est continue en $\underline{x}_0 \in S$.

1. Si pour chaque $\sigma > 0$ il existe $\rho(\sigma, \underline{x}_0) > 0$ tel que $|\underline{x} - \underline{x}_0| < \rho(\sigma, \underline{x}_0)$ implique $|f(\underline{x}) - f(\underline{x}_0)| < \sigma$.
2. Si $\rho(\sigma)$ est indépendante de $\underline{x}_0$ alors $f(\underline{x})$ est dite uniformément continue.
3. Une autre manière de définir la continuité uniforme est : Si $f(\underline{x})$ est continue et si sa dérivée $\dot{f}(\underline{x})$ est bornée alors $f(\underline{x})$ est uniformément continue.
4. Une fonction $f(\underline{x}) :\ S \to \Re$ est différentiable si sa dérivée $\dot{f}(\underline{x})$ existe.
5. Une fonction $f(\underline{x}) :\ S \to \Re$ est continûment différentiable si sa dérivée $\dot{f}(\underline{x})$ existe et est continue.
6. Une fonction $f(\underline{x}) :\ S \to \Re$ est localement Lipschitzienne, si pour $\underline{x}, \underline{y} \in S$ nous avons $\left|f(\underline{x}) - f(\underline{y})\right| < L(S)\left|\underline{x} - \underline{y}\right|$ où $L(S)$ est dite constante de Lipschitz. Si $S = \Re^n$ alors $f(\underline{x})$ est globalement Lipschitzienne.
7. Si $f(\underline{x})$ est globalement Lipschitzienne, alors $f(\underline{x})$ est uniformément continue.
8. Si $f(\underline{x})$ est continûment différentiable alors $f(\underline{x})$ est localement Lipschitzienne.

Normes des signaux

Soit le vecteur $\underline{x}(t) \in \Re^{n\times 1}$, alors sa norme L_p est donnée par :

$$|\underline{x}(t)|_p = \left(\int_0^\infty |\underline{x}(t)|^p \, dt\right)^{\frac{1}{p}}$$

Si $p = \infty$ alors :

$$|\underline{x}(t)|_\infty = \sup_t |\underline{x}(t)|$$

Si L_p est finie, on dit que $\underline{x}(t) \in L_p$. Notons qu'un signal $\underline{x}(t) \in L_\infty$ s'il est borné ; et $\underline{x}(t) \in L_2$ s'il est à énergie finie.

Lemme de Barbalat :

Soit le vecteur $\underline{e}(t) \in \Re^{n\times 1}$, si $\underline{e}(t) \in L_2$ et $\dot{\underline{e}}(t) \in L_\infty$ alors $\underline{e}(t) \to 0$.

Fonction de Lyapnunov :

Une fonction scalaire continue $V(\underline{x})$ est dite localement définie positive si $V(0) = 0$, et dans $S \in \Re^n$, $V(\underline{x}) > 0, \forall \underline{x} \neq 0$.

Si $S = \Re^n$, alors $V(\underline{x})$ est dite globalement définie positive.

Soit le système $\dot{\underline{x}} = f(\underline{x})$ alors :

Si la fonction $V(\underline{x})$ est définie positive et ses dérivées partielles sont continues, et si sa dérivée $\dot{V}(\underline{x})$ par rapport au temps le long de toutes les trajectoires d'état du système est semi-définie négative, alors $V(\underline{x})$ est dite une fonction de Lyapunov pour ce système.

Stabilité :

Soit le système $\dot{\underline{x}} = f(\underline{x})$ alors :

S'il existe une fonction scalaire $V(\underline{x})$ avec une première dérivée continue telle que :

1. $V(\underline{x})$ est définie positive,
2. $\dot{V}(\underline{x})$ est semi-définie négative,

alors le point d'équilibre du système est stable.

Si, de plus, $\dot{V}(\underline{x})$ est définie négative, alors la stabilité est asymptotique.

S'il existe une fonction scalaire $V(\underline{x})$, avec une dérivée continue telle que :

1. $V(\underline{x})$ est définie positive,
2. $\dot{V}(\underline{x})$ est définie négative,
3. $V(\underline{x}) \to \infty$ lorsque $|\underline{x}| \to \infty$,

Alors le point d'équilibre est globalement asymptotiquement stable.

Abstract:

The nonlinear control methods rely on the existence of an analytical model of the system. However, for complex nonlinear systems, the model may be unusable, inaccurate or simply nonexistent. On the other hand, the adaptive control provides a powerful tool for the control of linear systems and for reducing parametric uncertainty. Neural networks have emerged recently as adjustable approximators able to reproduce the behaviour of complex nonlinear systems. In this thesis, we combine adaptive control and neural networks to take advantage of their common features in the solution of problems of uncertain nonlinear systems control. This thesis describes three contributions to the neural control of nonlinear systems. The first contribution extends the model reference adaptive control to a class of uncertain nonlinear systems. The second contribution eliminates the need to measure all system states by introducing an observer to estimate the unmeasurable states. The third contribution presents an indirect adaptive neural control, applicable to a wide class of nonlinear multivariable systems. The theoretical results are verified by simulation on unstable and chaotic nonlinear systems.

Keywords: Nonlinear systems, adaptive control, neural networks, universal approximation, MRAC, Observer, uncertainties, Stability.

ملخص:

تعتمد طرق التحكم غير الخطية على وجود نموذج رياضي للنظام. و لكن، بالنسبة للأنظمة غير الخطية المعقدة، قد يكون النموذج غير قابل للاستخدام، غير دقيق أو غير موجود بكل بساطة. من ناحية أخرى، فإن التحكم المتكيف يوفر أداة قوية للسيطرة على الأنظمة الخطية وللتخفيض من عدم الدقة الحدودية. ظهرت، في الآونة الأخيرة، الشبكات العصبية كمقربات شاملة قابلة للتكيف وقادرة على إنتاج سلوك الأنظمة غير الخطية المعقدة. في هذه الرسالة، نقوم بالجمع بين التحكم المتكيف والشبكات العصبية للاستفادة من الميزات المشتركة في حل مشاكل السيطرة على الأنظمة غير الخطية غير الدقيقة. هذه الرسالة تقدم ثلاث مساهمات في التحكم المتكيف في الأنظمة غير الخطية باستعمال الشبكات العصبية. المساهمة الأولى تعمم التحكم المتكيف مع نموذج مرجعي إلى الأنظمة غير الخطية غير الدقيقة. المساهمة الثانية تلغي الحاجة لقياس جميع متغيرات النظام غير الخطي عن طريق إدخال مراقب لحساب المتغيرات غير المقاسة. وتناولت المساهمة الثالثة التحكم المتكيف غير المباشر باستعمال الشبكات العصبية، و الذي يمكن تطبيقه على فئة واسعة من الأنظمة غير الخطية متعددة المتغيرات. تم التحقق من النتائج النظرية بالتطبيق، عبر المحاكاة، على أنظمة غير خطية غير مستقرة وفوضوية.

كلمات مفتاحية : أنظمة غير الخطية، التحكم المتكيف، الشبكات العصبية، التقريب الشامل، MRAC ، المراقب، عدم الدقة، الاستقرار.