REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTRE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

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

N° d'Ordre :……..
Série :……...

MEMOIRE DE MAGISTER

Présenté Par

MR. GUITANI Issam

Option : Contrôle des systèmes

THEME

Commande Adaptative Neuronale par Retour de Sortie des Systèmes Non Linéaires

Soutenu le : .../…/2007

Examiné Par le Jury :

Président :	F. SOLTANI	Professeur	Univ de Mentouri Constantine
Rapporteur :	K. BELARBI	Professeur	Univ de Mentouri Constantine
Examinateurs:	S. FILALI	Professeur	Univ de Mentouri Constantine
:	M^{ed}.Boucherma	Maître de conférence	Univ de Mentouri Constantine

Année 2007

بسم ٱللَّه ٱلرحمن ٱلرحيم

و ما أوتيتم من ٱلعلم إلا قليلا

Remerciements

Je remercie Allah tout puissant qui m'a donné la force et la volonté pour pouvoir finir ce mémoire de magister

Je tiens à remercier profondément mon encadreur : le professeur **Khaled BELARBI** pour la confiance qu'il m'a accordée, ses encouragements, et ses précieux conseils.

J'exprime ma gratitude envers Mr. **Faouzi SOLTANI** professeur à l'université de Mentouri de Constantine de m'avoir fait l'honneur d'accepter de présider le jury,

Je tiens à remercier Mr. **Mohamed BOUCHERMA** maître de conférence à l'université Mentouri de Constantine, d'avoir accepter de juger ce travail en tant qu'examinateur,

Je remercie le Mr. **Salim FILALI**, Professeur à l'université Mentouri de Constantine, d'avoir accepter faire patrie du jury

Je tiens à remercier, tous ceux qui m'ont enseigné durant toutes mes études et en particulier mes enseignants à l'université de Constantine

DEDICACES

Je dédie ce mémoire

A mes chers parents pour tout les sacrifices consentis, pour leur soutient durant toute mes années d'études. Pour leur bonté et leur amour.

A Khaoula

A ma sœurs Ikbel et son mari Chawki et la petite Allae

A mes deux frères Adel et Achour

A toute ma grande famille.

A mes collègues de la promotion et mes amis.

Guitani Issam

SOMMAIRE

Chapitre I

Introduction générale 1

Chapitre II

Les réseaux de neurones dans la commande des systèmes 3

I Introduction……3

II Eléments de base……3

II.1 Structure de base 3

II.2 Fonction d'activation 4

II.3 C'est quoi un réseau de neurones ? 5

II.4 L'apprentissage des réseaux de neurones 6

II.5 Algorithme d'apprentissage 6

III Les réseaux de neurones dans la commande des systèmes……7

III .1 La commande inverse 8

III.2 Commande basée sur l'erreur de sortie 9

III. 3 Commande adaptative neuronale 10

III.3.1 La commande adaptative neuronale indirecte 11

III.3.2 La commande adaptative neuronale directe 12

Chapitre III

Commande adaptative neuronale basée sur un estimateur de l'erreur de commande 13

I Introduction……13

II. Structure de base de la commande adaptative neuronale à retour d'état ……13

II.1 Les réseaux de neurones 14

II.2 Le système à inférence floue 15

III Structure de la commande adaptative neuronale à retour de sortie……18

III.1 La structure de base 18

III.2 Les observateurs non linéaires 19

Chapitre IV

Résultats de simulation 21

I Introduction……21

II les résultats de simulation……21

II.1 Exemple 1: le pendule inverse 21

a) Retour d'état 22

b) Le retour de sortie 23

c) Résultats 23

II 2 Exemple 2: Le CSTR 30

a)Retour de d'état 31

b) Le retour de sortie 32

c) Résultats 32

Chapitre V

Conclusion générale 39

Annexe A

Systèmes d'inférence floue 40

REFERENCES 42

LISTE DES FIGURES

Figure II.1 Le neurone formèle 4
Figure II.2 Différentes fonctions d'activation 5
Figure II.3 Principe de la commande neuronale inverse 8
Figure II.4 Apprentissage de la commande inverse 9
Figure II.5 Commande à modèle de référence 9
Figure II.6 Structure de commande neuronale avec émulateur 10
Figure II.7 Schéma de commande indirecte basée sur la linéarisation par retour 12
Figure III.1 Structure de commande adaptative à retour d'état 14
Figure III.2 Les fonctions d'appartenance utilisées dans le FIS 18
Figure III.3 Structure de commande adaptative à retour de sortie 19
Figure IV.1 le pendule inverse 21
Figure IV.2 la position du pendule par retour d'état 24
Figure IV.3 Le signal de commande de pendule pour le retour d'état 24
Figure IV.4 L'erreur estimée par le FIS pour le retour d'état 25
Figure IV.5 les poids du RN pour le retour d'état 25
Figure IV.6 L'erreur filtrée es pour le retour d'état 26
Figure IV.7 la position du pendule par retour de sortie 27
Figure IV. 8 Le signal de commande de pendule pour le retour de sortie 27
Figure IV.9 L'erreur estimée par le FIS pour le retour de sortie 28
Figure IV.10 Les poids du RN pour le retour de sortie 28
Figure IV.11 L'erreur filtrée pour le retour de sortie 29
Figure IV.12 La sortie de l'observateur non linéaire 29
Figure IV.13 Le CSTR 30
Figure IV.14 concentration d'effluent dans le CSTR par retour d'état 33
Figure IV.15 Le signal de commande (débit du refroidissant) pour le retour d'état 33
Figure IV.16 L'erreur estimée par le FIS pour le retour d'état 34
Figure IV.17 Les poids du RN pour le retour d'état 34
Figure IV.18 L'erreur filtrée es pour le retour d'état 35

Figure IV.19 Concentration d'effluent dans le CSTR par retour de sortie 36
Figure IV.20 Le signal de commande (débit du refroidissant) pour le retour de sortie 36
Figure IV.21 L'erreur estimée par le FIS pour le retour de sortie 37
Figure IV. 22 Les poids du RN pour le retour de sortie 37
Figure IV.23 L'erreur filtrée es pour le retour de sortie 38
Figure IV.24 La sortie de l'observateur non linéaire 38

Liste des tableaux

Tableau III.1 BR1 16
Tableau III.2 BR2 16
Tableau IV.1 paramètres du CSTR 31

CHAPITRE I

INTRODUCTION GENERALE

Chapitre I

Introduction générale

Les réseaux de neurones artificiels ont trouvé une large utilisation dans le domaine de la commande des systèmes non linéaires. Ceci est dû à leur propriété d'approximation universelle qui les rend capables d'approximer, avec un degré de précision arbitraire fixé, n'importe quelle fonction non linéaire [3,6]**.** Les réseaux à base de perceptrons multicouches, Multi Layer Perceptrons, MLP, et les fonctions radiales de base RBF, sont les plus utilisés. Les premières applications des réseaux de neurones en commande n'étaient pas basées sur des analyses de stabilité rigoureuses. Plus tard, cependant, la théorie de Lyapunov a été introduite pour calculer des lois adaptatives garantissant la stabilité du système en boucle fermée. En général, la loi de commande est exprimée en fonction des non linéarités du modèle du système suivant la méthode de linéarisation entrée-sortie. Les réseaux de neurones sont, ensuite, utilisés soit pour approcher directement la loi de commande soit pour approcher les non linéarités formant ainsi une loi de commande neuronale adaptative. Le modèle non linéaire utilisé dans ces études satisfait les conditions de linéarisation entrée sortie. Dans tous ces travaux, le signal erreur utilisé pour l'apprentissage dans les lois d'adaptations est basé sur l'erreur de poursuite.

Dans ce mémoire, une structure de commande neuronale avec des lois d'adaptation basée sur le signal erreur de commande est étudiée. Dans ce cas la fonction à optimiser dépend alors directement des poids. Cependant, le signal de commande idéale ne pouvant être calculé, il est remplacé par une estimée, cette estimée sera obtenue par un système d'inférence flou.

Dans le deuxième chapitre, nous présentons les éléments de base et les principales techniques de la commande à base de réseaux de neurones.

Dans le troisième chapitre, nous introduisons la structure de commande des systèmes non linéaires basée sur un contrôleur neuronal et un estimateur flou pour deux cas : le premier cas est une commande par retour d'état le second une commande à retour de sortie basée sur un observateur. Dans le quatrième chapitre, nous présentons les résultats de simulation de ces deux stratégies appliquées à deux exemples : la commande de la position de pendule inverse et la commande de concentration dans un réacteur chimique.

CHAPITRE II

LES RESEAUX DE NEURONES DANS LA COMMANDE DES SYSTEMES

Chapitre II

Les réseaux de neurones dans la commande des systèmes

I Introduction:

La recherche sur les réseaux de neurones a connu un développement important ces dernières années, tant du coté architecture où plusieurs modèles sont proposés, que du coté algorithmes d'apprentissages utilisés pour entraîner ces réseaux. En effet, ces travaux de recherche ont montré que les réseaux de neurones sont des approximateurs universels, ce qui permet de modéliser n'importe quel système non linéaire, et ils sont principalement, dotés de deux propriétés importantes : l'apprentissage et la généralisation. Ces travaux de recherche ont donné lieu à des applications trés intéressantes des réseaux de neurones dans plusieurs domaines, et en particulier le domaine de la commande des systèmes non linéaires.

II Eléments de base:

II.1 Structure de base:

Le premier modèle du neurone formel date des années quarante. Il été présenté par Mc Culloch et Pitts. S'inspirant de leurs travaux sur les neurones biologiques ils ont proposés le modèle suivant:

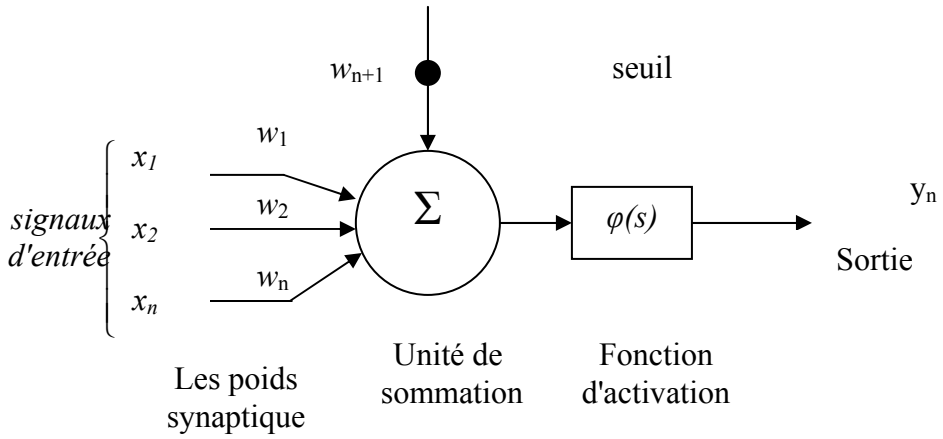

Figure II.1 le neurone formèle

Un neurone formel est une fonction algébrique non linéaire et bornée, dont la valeur dépend des paramètres appelés poids synaptiques ou poids des connexions. D'une façon plus générale, un neurone formel est un élément de traitement (opérateur mathématique) possédant n entées (qui sont les neurones externes ou les sorties des autres neurones), et une seule sortie. Ce modèle est décrit mathématiquement par les équations suivantes:

$$s = \sum_{i=1}^{n+1} w_i x_i \qquad (1)$$

$$y_n = \varphi(s)$$

Où *xi,* w_i, φ *et* y_n sont respectivement, les entrées, les poids synaptiques, la fonction d'activation et la sortie du neurone.

II.2 Fonction d'activation:

Les fonctions d'activations représentent généralement certaines formes de non linéarité. L'une des formes de non linéarité la plus simple, et qui est appropriée aux réseaux discret, est la fonction signe, figure II.2.a. Une autre variante de ce type des non linéarités est la fonction de Heaviside, figure II.2.b.Pour la majorité des algorithmes d'apprentissage il est nécessaire d'utiliser des fonctions sigmoïdes différentiables, telles

que la fonction sigmoïde unipolaire, figure II.2.c et la fonction sigmoïde bipolaire, figure II.2.d. La classe, la plus utilisée des fonctions d'activation, dans le domaine de la modélisation et de la commande des systèmes non linéaires est la fonction sigmoïde bipolaire.

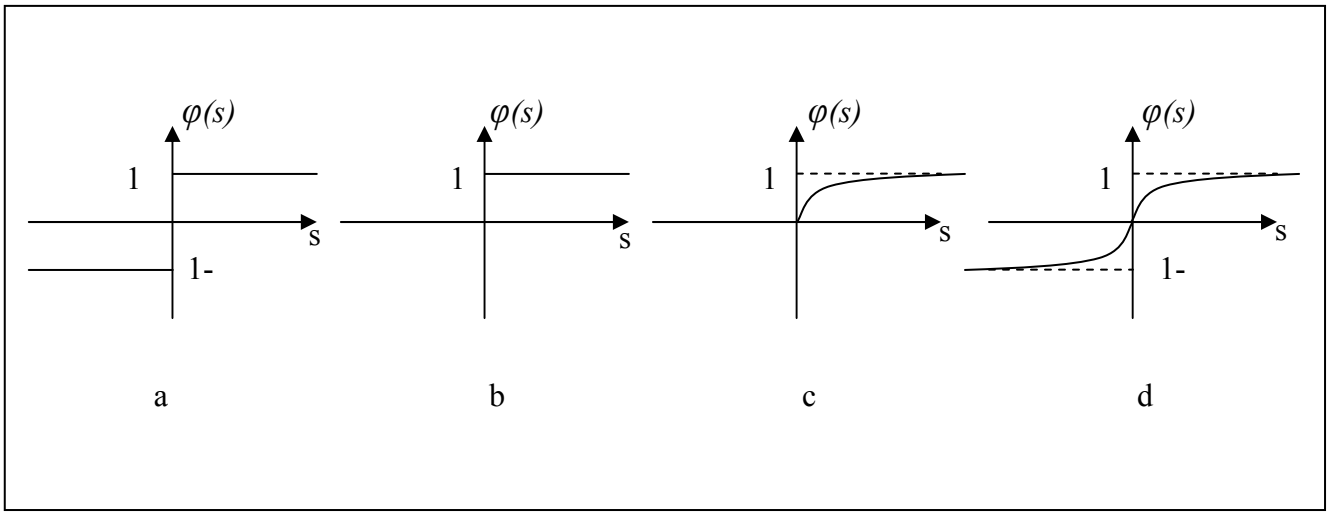

Figure II.2 Différentes fonctions d'activation

II.3 C'est quoi un réseau de neurones ?

Un réseau de neurone est un ensemble d'éléments de traitement de l'information, avec une topologie spécifique d'interconnexions (architecteur du réseau) entre ces éléments et une loi d'apprentissage pour adapter les poids de connexions (poids synaptiques), il est caractérisé par un parallélisme à gain très fin et à forte connectivité. Nous entendons par là que dans un réseau de neurones donné, l'information est traitée par grand nombre de processeurs élémentaires très simples, chacun étant relié à d'autres processus. Ce processus très simple est un neurone formel désigné ainsi son fonctionnement s'inspire d'une modélisation des cellules biologiques. Ils sont dotés de deux propriétés importantes qui sont à l'origine de leur intérêt pratique des domaines très divers:

- Capacité d'adaptation ou d'apprentissage qui permet au réseau de tenir compte des nouvelles contraintes ou de nouvelles données du monde extérieur,

- Capacité de généralisation qui est son aptitude de donner une réponse satisfaisante à une entrée qui ne fait pas partie des exemples à partir desquels il appris.

II.4 L'apprentissage des réseaux de neurones:

L'information que peut acquérir un réseau de neurones est représentée dans les poids des connexions entre les neurones. L'apprentissage consiste donc à ajuster ces poids de telle façon que le réseau présente certains comportements désirés. En d'autres termes, l'apprentissage des réseaux de neurones consiste à ajuster les poids synaptiques de telle manière que les sorties du réseau soient aussi proches que possible des sorties désirées. Il existe trois types d'apprentissage:

- L'apprentissage supervisé: pour lequel on dispose de la sortie désirée et qui consiste à ajuster les poids synaptiques de tel sorte à minimiser l'écart entre la sortie désirée et la sortie du réseau,
- L'apprentissage non supervisé: pour lequel le réseau de neurones organise lui-même les entrées qui lui sont présentées de façon à optimiser un critère de performances donné,
- L'apprentissage par renforcement: pour lequel le réseau de neurones est informé d'une manière indirecte sur l'effet de son action choisie. Cette action est renforcée si elle conduit à une amélioration des performances.

II.5 Algorithme d'apprentissage:

L'apprentissage supervisé pour les réseaux de neurones, consiste à adapter les poids synaptiques de telle manière que l'erreur entre la sortie du réseau et la sortie désirée soit aussi petite que possible. La plupart des algorithmes d'apprentissage des réseaux de neurones est basée sur les méthodes du gradient: ils cherchent à minimiser un critère de la forme suivante:

$$J = \frac{1}{2}\sum_{q=1}^{m}\left(y_{rq}(Xp) - y_q^d(Xp)\right)^2 \qquad (2)$$

Où y_{rq} et y_q^d sont la sortie du réseau et la sortie désirée pour le vecteur d'entrée X_p.

Cette optimisation se fait d'une manière itérative, en modifiant les poids en fonction du gradient de la fonction de coût selon la loi d'adaptation suivante:

$$w_{ji}^{l}(k+1) = w_{ji}^{l}(k) - \eta \frac{\partial J}{\partial w_{ji}^{l}} \quad (3)$$

Où

w_{ji}^{l} est le poids de la connexion entre le $j^{ème}$ neurone de la couche l et le $i^{ème}$ neurone de la couche l-1

i=1,…,n+1 la $i^{ème}$ composante du vecteur d'entrée,

j=1,…,m+1 la $i^{ème}$ composante du vecteur de sortie,

l=1,…,L l'ordre d'une couche dans le réseau de neurone,

η est une constante positive appelée taux d'apprentissage.

Le gradient est calculé par une méthode spécifique aux réseaux de neurones, dites méthode de rétro propagation. Dans cette méthode il est possible de calculer le gradient de la fonction coût en retropropageant l'erreur commise en sortie vers les couches cachées.

III Les réseaux de neurones dans la commande des systèmes :

Par leur capacité d'approximation universelle, les réseaux de neurones sont bien adaptés pour la commande des systèmes non linéaires. En effet dans ce cas la fonction commande est une fonction non linéaire, l'objectif est alors d'approximer cette fonction par les RNA (réseaux de neurones artificiels). Cette approximation est réalisée par apprentissage des poids du réseau, l'apprentissage peut se faire hors ligne ou en ligne :

- Dans le cas de hors ligne, l'apprentissage est basé sur un ensemble de données définissant la fonction commande,
- Dans le cas de en ligne, la mise à jour des poids est essentiellement adaptative.

Il existe alors plusieurs algorithmes de commande par RNA basés sur ces deux structures, principalement on peut distinguer:

- La commande inverse (basée sur l'erreur de commande),
- Commande basée sur l'erreur de sortie (d'état),
- Commande adaptative,
- Commande basée sur le critique adaptatif (Adaptif Critic).

Dans ce chapitre nous introduisons les trois premières approches.

III .1 La commande inverse:

Considérons un système non linéaire avec une entrée *u(t)* et une sortie *y(t)*

y(t) peut dépendre de *u(t)* seulement ou de *u(t)* et les états précédents de *y* et/ou *u* CAD

$y(t)=F(u(t))$ (4)

Ou

$y(t)=F(u(t),u(t-1),u(t-2),...,y(t-1),y(t-2),...)$

L'objectif est de faire suivre *y(t)* une trajectoire de référence $y_d(t)$. Le contrôleur idéal réalise l'égalité : $y(t)=y_d(t)$, mathématiquement il doit réaliser l'inverse F^{-1} de la fonction F.

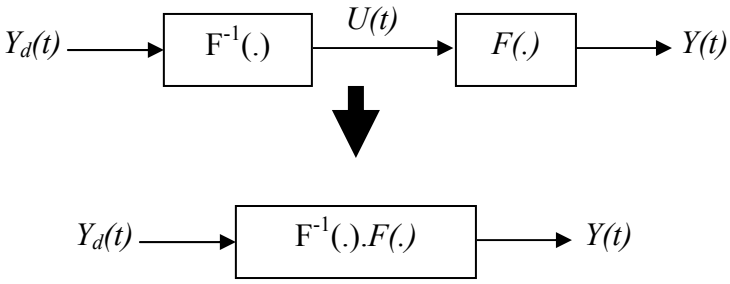

Figure II.3 Principe de la commande neuronale inverse

Le principe de la commande inverse par réseau de neurones consiste à construire le réseau (en générale MLP) qui approxime la fonction F^{-1}, cette approximation se fait par apprentissage hors ligne.

Etant donné la fonction F,il faut construire un RNA pour réaliser l'approximation, on génère une base de données à partir de F. L'objectif de l'apprentissage est de déterminer la fonction inverse à partir de ces données. A l'instant t, on donne au RNA :

y(t), y(t-1),y(t-2) et on obtient à la sortie $u_r(t)$. La mise à jour des poids du réseau est basée sur la minimisation de l'erreur entre $u_r(t)$ et u(t)

$$J = \frac{1}{2}\sum_{i=1}^{T}\left[u(t)-u_r(t)\right]^2 \quad (5)$$

Structurellement le mécanisme d'apprentissage est donné par la figure II.4.

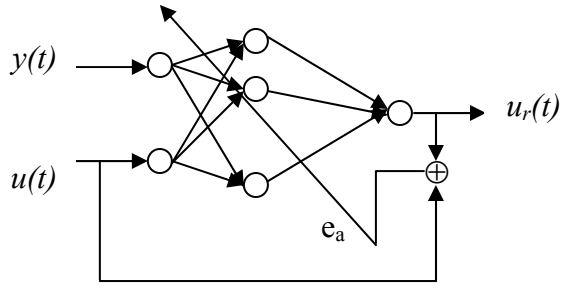

Figure II.4 Apprentissage de la commande inverse

Lorsque l'apprentissage est réalisé le réseau peut être utilisé comme contrôleur inverse. Cette méthode peut être appliquée aux systèmes invariants dans le temps.

III.2 Commande basée sur l'erreur de sortie:

Dans le cas de cette approche, il s'agit de déterminer le contrôleur à RNA qui minimise l'erreur entre la sortie et la référence. Il s'agit alors fondamentalement d'un apprentissage en boucle fermée, alors faut il le réaliser en ligne ou hors ligne?. La structure de l'apprentissage est donnée par la figure II.5.

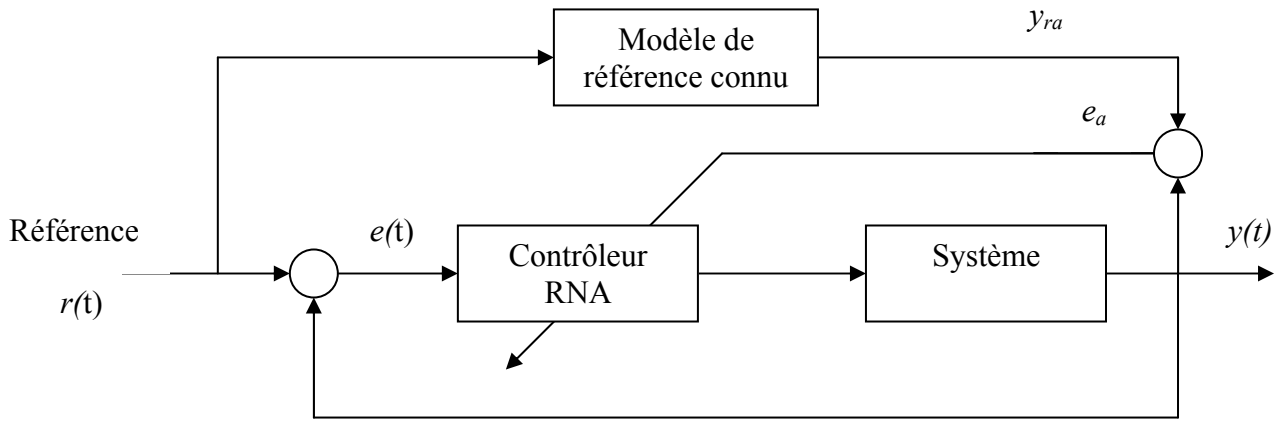

Figure II.5 Commande à modèle de référence

L'objectif est d'entraîner le réseau pour trouver les poids qui minimisent

$$J = \frac{1}{2}\left[y_{ra}(t) - y(t)\right]^2 = \frac{1}{2}\left[e_a(t)\right]^2 \qquad (6)$$

La commande u(t) dépend des poids du réseau, la mise à jour est alors de la forme :

$$v(t+1) = v(t) - \alpha \frac{\partial J}{\partial y} \frac{\partial y}{\partial u} \frac{\partial u}{\partial v} \tag{7}$$

Mais on ne connaît pas le Jacobien $\frac{\partial y}{\partial u}$

Plusieurs propositions ont été faites pour résoudre le problème du Jacobien, une des premières a été de créer un émulateur qui modélise le système.

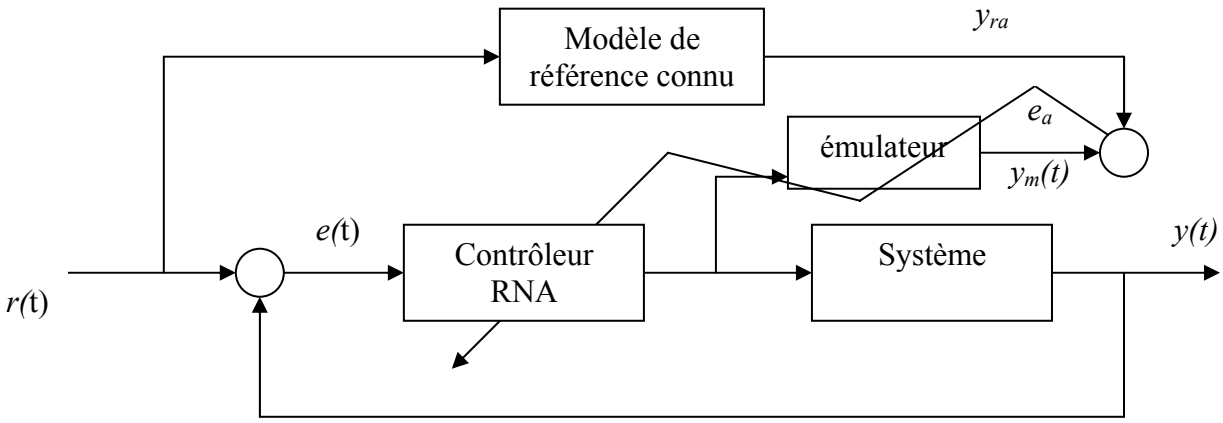

Figure II.6 : Structure de commande neuronale avec émulateur

Cette méthode fonctionne si $y_m(t) = y(t)$. Les étapes pour le développement de cette approche :

- Construire le RNA émulateur (à partir des données),
- Initialiser hors ligne le contrôleur RNA.

III. 3 Commande adaptative neuronale:

Dans cette stratégie de contrôle, les réseaux de neurones sont introduits pour construire des systèmes de commande adaptative stable basés sur la théorie de la stabilité de Lyapunov. Deux classes de méthodes ont été développées: directe et indirecte:

III 3 1 La commande adaptative neuronale indirecte :

Dans la première approche appelée commande adaptative neuronale indirecte, les réseaux de neurones sont utilisés pour approximer les non linéarités du système non linéaire [1,2 ,8,10,11]. Une des méthodes qui a connu le plus de développement [1,2 8 ,9] est basée sur la méthode de linéarisation par retour (feedback linearization) [7] avec un système non linéaire sous forme affine :

$$\begin{cases} \dot{x}(t) = f(x) + g(x)u(t) \\ y(t) = h(x) \end{cases} \tag{8}$$

La loi de commande linéarisante est de la forme :

$$u(t) = \frac{v - f(x)}{g(x)} \tag{9}$$

Lorsque les fonctions non linéaires f(x) et g(x) sont inconnues, deux réseaux de neurones sont utilisés pour obtenir leur approximations $\hat{f}(x)$ et $\hat{g}(x)$ et construire la commande $u(t)$ approximée .

Dans cette approche se pose cependant un problème de singularité de la commande lorsque $\hat{g}(x) = 0$. Plusieurs techniques ont été proposées pour éviter cette singularité, la plus intéressante étant la commande indirecte ci-dessous.

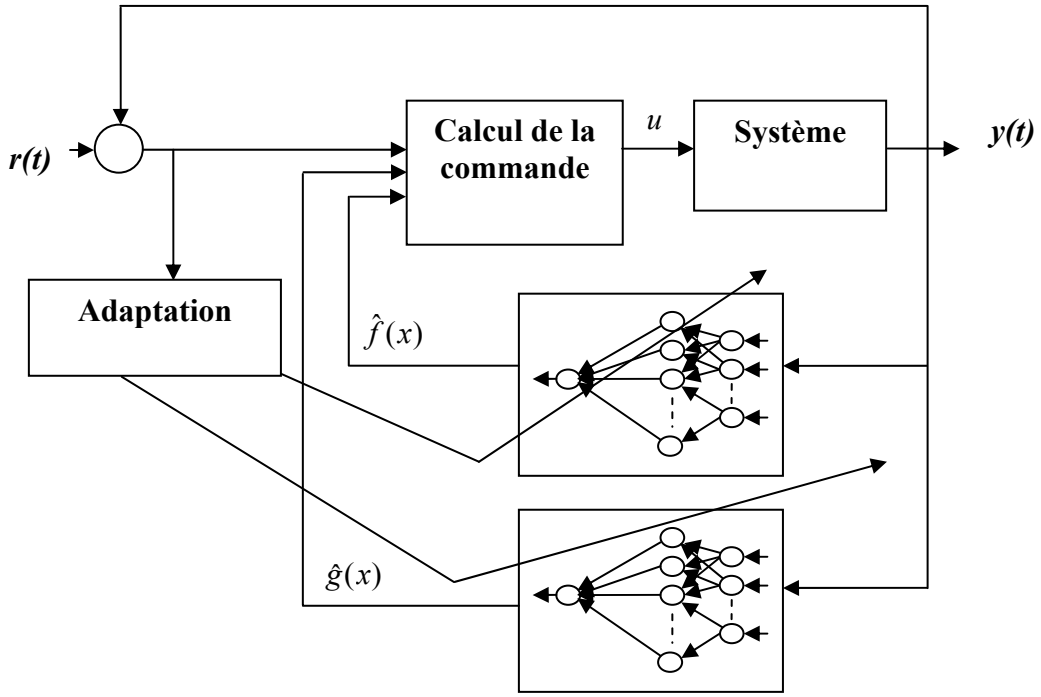

Figure II.7 Schéma de commande indirecte basée sur la linéarisation par retour

III 3 2 La commande adaptative neuronale directe

La deuxième approche appelée commande adaptative directe le seul réseau de neurones est introduit pour approximer directement la loi de commande idéale non constructible $u(t)$ ci-dessus [4,5, 12]. Dans ces méthodes le signal d'adaptation est l'erreur de poursuite. Il faut noter cependant que les développements mathématiques sont plus élaborés pour cette méthode. Par exemple la méthode proposée dans [12] consiste à approximer la loi :

$$u(x) = -\frac{1}{g(x)}(f(x)+v) + 2\frac{\dot{g}(x)}{g(x)^2}e_s \qquad (10)$$

où $e_s = \sum_{i=1}^{n} \lambda_i . e_i$ avec $e = [e_1, e_2, .., e_n]^T$ et $e_i = x_i - x_d^{i-1}$ i=1,…n

et $v = -y_d^{(n)} + \sum_{i=1}^{n-1} \lambda_i e_i$

x_d est un signal de référence borné et supposé n fois dérivable. Les coefficients λ_i sont choisit tel que le polynôme $S^n + \lambda_{n-1} S^{n-1} + ... + \lambda_1$ a tous ses racines strictement dans le demi plan complexe gauche et $\lambda_n = 1$ donc $\boldsymbol{e}(t) \to 0$ tan que $e_s \to 0$.

CHAPITRE III

COMMANDE ADAPTATIVE NEURONALE BASEE SUR UN ESTIMATEUR DE L'ERREUR DE COMMANDE

Chapitre III

Commande adaptative neuronale basée sur un estimateur de l'erreur de commande

I Introduction

Dans ce chapitre, nous introduisons la structure de commande neuronale basée sur un signal d'adaptation utilisant une estimation de l'erreur de commande. Cette estimation est délivrée par un système d'inférence flou. Deux structures de commande seront présentées dans ce chapitre. La première structure est une commande par retour d'état, la deuxième est une commande par retour de sortie utilisant un observateur d'état non linéaire. Ces deux structures sont basées sur deux éléments qui seront détaillés dans ce chapitre : le réseau de neurones contrôleur et l'estimateur flou de l'erreur de commande.

II. Structure de base de la commande adaptative neuronale à retour d'état :

Dans ce paragraphe nous introduisons un contrôleur adaptatif $u(t)$ a retour d'état pour un système non linéaire mono variable sous la forme générale :

$$\begin{cases} \dot{x} = f(x,u) \\ y = h(x) \end{cases} \qquad (1)$$

$x = [x_1, x_2, ..., x_n]^T \in R^n$ est le vecteur d'état supposé complètement disponible.

$u, y \in R$ sont l'entrée et la sortie du système respectivement.

La structure de commande est donnée par la figure suivante :

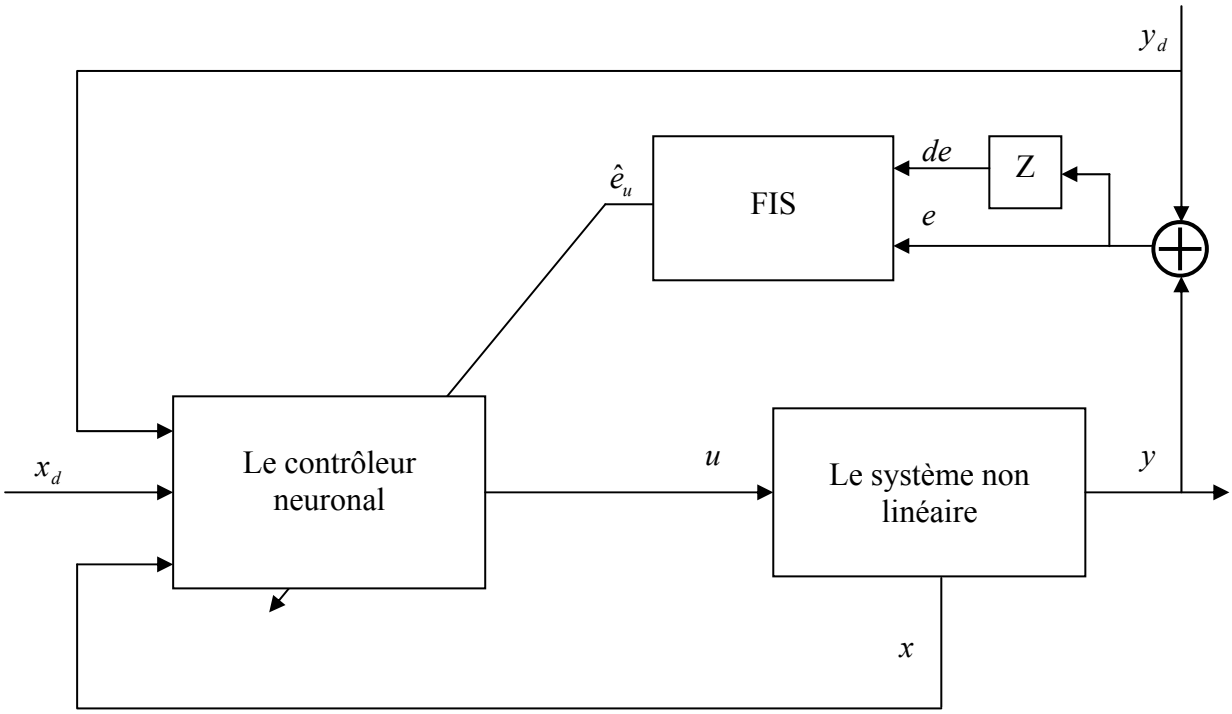

Figure III.1 Structure de commande adaptative à retour d'état

Le système d'inférence flou délivre le signal d'adaptation $\hat{e}_u$ au contrôleur neuronal. Ce signal est alors utilisé pour la mise à jour des poids de connexion du réseau. Dans les paragraphes suivants, nous détaillons la structure et les paramètres de ces deux éléments.

II 1 Les réseaux de neurones

La sortie du réseau de neurones est donnée par

$$u(z) = \boldsymbol{W}^{T} \cdot \boldsymbol{S}\left(\boldsymbol{V}^{T} \cdot \boldsymbol{Z}\right) \tag{2}$$

Où $\boldsymbol{V} = \left[v_1, v_2, ..., v_l\right] \in \Re^{(n+1)\times l}$ et $\boldsymbol{W} = \left[w_1, w_2, ..., w_l\right]^T \in \Re^{l}$

Sont les poids du réseau de neurones,

$\mathbf{z}$ est le vecteur d'entrée, $\boldsymbol{x} = \left[x_1, x_2, ..., x_n\right]^T$ est le vecteur d'état du système. La fonction lisse u^* peut être approximée comme suit:

$$u^*(z) = \hat{\boldsymbol{W}}^{T^*} \cdot \boldsymbol{S}\left(\hat{\boldsymbol{V}}^{*T} \cdot \boldsymbol{Z}\right) + \varepsilon(z) \ \forall z \in \Omega_z, \tag{3}$$

Où $\varepsilon(z)$ est l'erreur d'approximation du réseau de neurones.

La sortie du contrôleur neuronal avec les poids $\hat{W}$ et $\hat{V}$ peut être écrite sous la forme :

$$\hat{u}\ (z) = \hat{\boldsymbol{W}}^{T} \cdot \boldsymbol{S}\left(\hat{\boldsymbol{V}}^{T} \cdot \boldsymbol{Z} \right) \tag{4}$$

On définit l'erreur de commande :

$$e_u^* = \hat{u}(z) - u^*(z) \tag{5}$$

Si on veut trouver les poids qui minimisent cette erreur, l'objectif sera

$$J = \frac{1}{2}(e_u^*)^2$$

Les lois de la mise à jour des poids sont obtenues par la dérivation de cette équation par rapport aux ces poids et en utilisant l'algorithme du gradient avec un pas prédéterminé :

$$\dot{\hat{\boldsymbol{W}}} = -\boldsymbol{\Gamma}_{\boldsymbol{w}} (e_u^* \cdot \boldsymbol{S}\left(\hat{\boldsymbol{V}}^{T} \cdot \boldsymbol{Z}\right) + \sigma_w \cdot \left|e_u^*\right| \cdot \hat{\boldsymbol{W}}) \tag{6}$$

$$\dot{\hat{\boldsymbol{V}}} = -\boldsymbol{\Gamma}_{v} (e_u^* \cdot \boldsymbol{Z} \cdot \hat{\boldsymbol{W}}^{T} \cdot \boldsymbol{S}'\left(\hat{\boldsymbol{V}}^{T} \cdot \boldsymbol{Z}\right) + \sigma_v \cdot \left|e_u^*\right| \cdot \hat{\boldsymbol{V}}) \tag{7}$$

Où $\boldsymbol{S}'\left(\hat{\boldsymbol{V}}^{T} \cdot \boldsymbol{Z}\right) = diag\{s_1', s_2', ..., s_l'\}$ et $s_i' = d\left[\boldsymbol{S}(z)\right]/dz\big|_{z=a_i}$, $\hat{\boldsymbol{V}}^{T} \cdot \boldsymbol{Z} = [a_1, a_2, ..., a_l]^T$

Et $\boldsymbol{\Gamma}_w = \delta_w \cdot \mathbf{I} > 0$, $\boldsymbol{\Gamma}_v = \delta_v \cdot \mathbf{I} > 0$ sont les taux de mise à jour de e_u^* l'erreur de commande.

II 2 Le système à inférence floue :

L'erreur de commande e_u^* n'est connue, un système d'inférence flou (FIS, fuzzy inférence system), est utilisé pour fournir un estimé $\hat{e}_u$. Le FIS transforme les informations relatives à l'historique de l'erreur de poursuite $e_y(t)=y_d(t)-y(t)$ en une estimé $\hat{e}_u$ qui est utilisée à la place de e_u^*. Le composant principal du FIS est la base de règles qui est composée des règles sous la forme suivante :

Si X1 est Négatif, et X2 est Zéro ….. alors Eu est positif.(par exemple)

Où X1, X2 sont les variables linguistiques associées avec l'erreur de poursuite de la sortie et sa dérivée, Eu est la variable linguistique associée avec l'erreur de commande estimé ; Négatif, Zéro, Positif (N, Z, P) sont des valeurs linguistiques associées avec ces variables.

On peut distinguer entre deux bases de règles dépendantes à la direction relative du signal d'entrée et la sortie.

La base de règles 1: dans le cas où l'entrée et la sortie ont la même direction CAD si l'entrée augmente (diminue), la sortie aussi augmente (diminue), et une autre base de règles pour les autres cas.

Le raisonnement heuristique pour générer BR1 et BR2 est comme suit :

- Eu est Z: si *e(t)* et *de(t)* sont Z, ou quand l'erreur diminue en gardant le même signe (la sortie se rapproche de la référence de haut ou de bas),
- Eu est N: si *e(t)* est N et *de(t)*=0, ou *de(t)*<0 (la sortie se stabilise en un point supérieur à la référence ou la sortie se diverge de la référence par le haut),
- Eu est P: si *e(t)* est N et *de(t)*=0, ou *de(t)*>0 (la sortie se stabilise en un point inférieur à la référence ou la sortie se diverge de la référence par le bas).

Cette procédure donne une estimée avec un signe correct.

Les deux tableaux suivants résument BR1 et BR2 dans le cas ou les variables sont fuzzifiées avec trois sous ensembles flous, N, Z et P

de \ ***e***	**N**	**Z**	**P**
N	**N**	**N**	**Z**
Z	**N**	**Z**	**P**
P	**Z**	**P**	**P**

Tableau III.1 BR1

de \ ***e***	**N**	**Z**	**P**
N	**P**	**P**	**Z**
Z	**P**	**Z**	**N**
P	**Z**	**N**	**N**

Tableau III.2 BR2

La structure de base de l'estimateur flou est définie par la base de règles RB1 ou RB2. Il comprend deux entrées floues, l'erreur de sortie et sa variation et une sortie floue, l'estimée de l'erreur de commande $\hat{e}_u$. Les trois variables floues prennent trois valeurs linguistiques : NEGATIVE, ZERO, POSITIVE. Pour définir complètement l'estimateur, il reste à choisir les fonctions d'appartenance, associés aux valeurs linguistiques et leurs répartitions sur leurs univers de discours respectifs. Celles-ci sont données par la figure III.2 ou l'on remarque que les fonctions d'appartenance sont uniformément réparties sur l'univers de discours. Les paramètres du FIS sont déterminés de façon à obtenir une estimée $\hat{e}_u$ normalisée comme suit:

$$-1 < \hat{e}_u < +1 \qquad (8)$$

Ceci peut être obtenu par un choix approprié des paramètres c_N, c_Z et c_Z de la figure 3. La valeur numérique de l'estimée de l' e_u est calculée par la formule de défuzzification du centre de gravité donnée par :

$$\hat{e}_u = \frac{\sum_{k=1}^{m} c_k \mu_{B_k}(\hat{e}_{u_k}).}{\sum_{k=1}^{m} \mu_{B_k}(\hat{e}_{u_k})} \tag{9}$$

où c_k est le centre de l'ensemble flou B_k de la sortie. La fonction d'appartenance $\mu_{B_k}(\hat{e}_u)$ est borné par:

$$0 \le \mu_{B_k}(\hat{e}_u) \le 1 \tag{10}$$

La valeur de $\hat{e}_u$ peut être contrôlée par le choix des coefficients c_k . Les valeurs suivantes: $c_N = -1$, $c_Z = 0$, $c_P = +1$ pour l'erreur de commande, permettent de satisfaire (8). Lorsqu'on remplace ces valeurs dans (9), on obtient :

$$\hat{e}_u = \frac{-1 \times \mu_N(\hat{e}_u) + 0 \times \mu_Z(\hat{e}_u) + 1 \times \mu_P(\hat{e}_u)}{\mu_N(\hat{e}_u) + \mu_Z(\hat{e}_u) + \mu_P(\hat{e}_u)} = \frac{-\mu_N(\hat{e}_u) + \mu_P(\hat{e}_u)}{\mu_N(\hat{e}_u) + \mu_Z(\hat{e}_u) + \mu_P(\hat{e}_u)} \tag{11}$$

où $\mu_N(\hat{e}_u)$, $\mu_Z(\hat{e}_u)$ et $\mu_P(\hat{e}_u)$ sont les degrés d'appartenance de $\hat{e}_u$ aux ensembles flous NEGATIF, ZERO ET POSITIVE qui sont calculées à partir des degrés d'appartenance de l'erreur $\mu_N(e)\ \mu_Z(e)\ \mu_P(e)$ et de sa variation $\mu_N(de)\ \mu_Z(de)\ \mu_P(de)$ en utilisant le mécanisme d'inférence du (*min, max)*. I

Les formules suivantes son utilisées pour calculer les degrés d'appartenance:

$$\mu_N(v) = \begin{cases} 1 & if\ v \le c_N \\ \dfrac{c_Z - v}{c_Z - c_N} & if\ c_N < v \le c_Z \\ 0 & if\ v > c_Z \end{cases} \tag{12}$$

$$\mu_Z(v) = \begin{cases} 0 & if\ v \le c_N \\ \dfrac{v - c_N}{c_Z - c_N} & if\ c_N < v \le c_Z \\ \dfrac{c_P - v}{c_p - c_Z} & if\ c < v < c_P \\ 0 & if\ v \ge c_P \end{cases} \tag{13}$$

$$\mu_P(v) = \begin{cases} 0 & if\ v \le c_Z \\ \dfrac{v - c_Z}{c_P - c_Z} & if\ c_Z < v < c_P \\ 1 & if\ v \ge c_P \end{cases} \tag{14}$$

v représente l'erreur de sortie $e_y(t)$ ou sa variation $de_y(t)$.

Les paramètres sont donnés par $c_N = -1, c_Z = 0, c_P = 1$ pour $e_y(t)$ et $c_N = -0.1$, $c_Z = 0$, $c_P = 0.1$ pour $de_y(t)$.

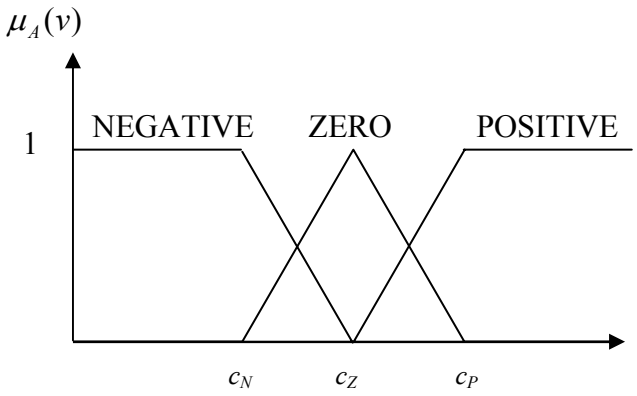

Figure III.2 Les fonctions d'appartenance utilisées dans le FIS

III. Structure de la commande adaptative neuronale à retour de sortie:

III 1 La structure de base

Dans la section précédente, nous avons introduit la structure de commande neuronale en supposant que les états du système sont disponibles. Ceci n'est généralement pas le cas, pour cela nous introduisons maintenant une structure à retour de sortie basée sur un observateur non linéaire. La figure III.3 décrit le schéma de principe de la commande. Dans ce qui suit nous rappelons brièvement la théorie des observateurs non linéaires à gain élevé.

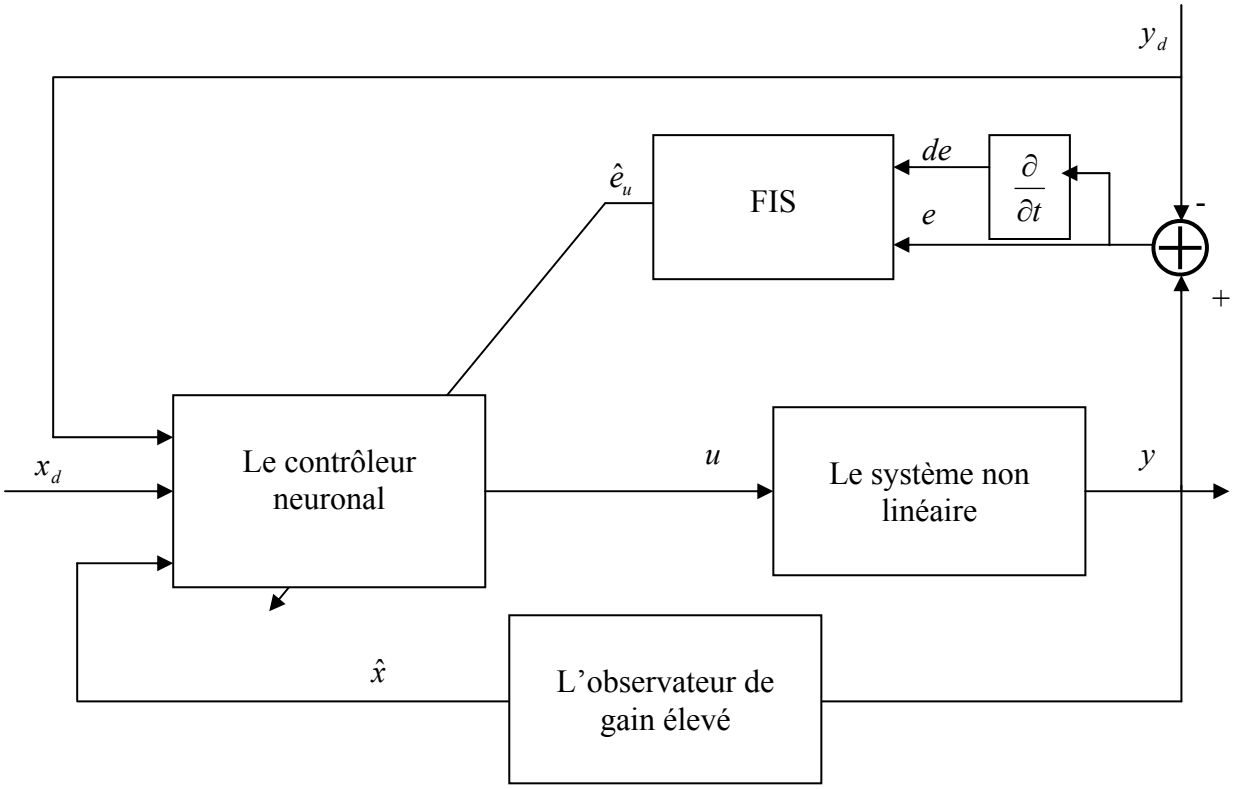

Figure III.3 Structure de commande adaptative à retour de sortie

III 2 Les observateurs non linéaires:

Dans les systèmes de commande à retour d'état la présence des variables d'états inconnues et non mesurables devient une difficulté qui peut être maîtrisée avec l'introduction d'un estimateur d'état approprié. Le développement d'algorithme pour permettre l'estimation a attiré l'attention de plusieurs chercheurs. Plusieurs techniques ont été introduites pour estimer les variables d'état en connaissant juste les valeurs mesurables du système. Il existe plusieurs formes d'estimateur qui peuvent être utilisées dépendant à la structure mathématique du modèle du processus et l'information du système.

La forme non linéaire de certains processus a nécessité le développement d'observateurs non linéaire. Ces observateurs sont conçus de sorte qu'ils peuvent prendre en compte la non linéarité intrinsèque du processus dynamique. Nous considérons ici un modèle d'observateur appelé l'observateur à gain élevé (high-gain observer) ;

Pour un système non linéaire de la forme générale

$$\begin{aligned} \dot{x} &= f(x) \\ y &= h(x) \end{aligned} \qquad (15)$$

$$x \in R^n$$

Supposons que la fonction *y(t)* et ses n dérivées sont bornées. Considérons le système linéaire suivant :

$$\begin{aligned} \varepsilon\dot{\xi}_i &= \xi_{i+1} \\ \varepsilon\xi_n &= -\sum_{i=1}^{n} b_i \cdot \xi_{n-i+1} + y(t) \end{aligned} \tag{16}$$

Où les paramètres $b_1,\ldots b_n$ sont choisis tel que le polynôme $S^n + b_1 S^{n-1} + \ldots b_{n-1} S + 1$ a tous ses racines strictement dans le demi plan complexe gauche et avec b_n=1

Alors il existe des constantes positives h_k, k=2,3,…,n, et t* telles que pour tout t>t* on a:

$$\frac{\xi_{k+1}}{\varepsilon^k} - y^{(k)} = -\varepsilon\psi^{(k+1)} \quad k=1,\ldots,n\text{-}1 \tag{17}$$

$$\left|\frac{\xi_{k+1}}{\varepsilon^k} - y^{(k)}\right| \le -\varepsilon h_{k+1} \quad k=1,\ldots,n\text{-}1 \tag{18}$$

Avec ε un petit nombre positif, $\psi = \sum_{i=1}^{n} b_i \cdot \xi_{n-i+1}$

Et $\left|\psi^k\right| \le h_k$, ψ^k la dérivée d'ordre k de ψ

Alors l'estimé $\hat{x}$ du vecteur x est donné par :

$$\hat{x} = \left[\hat{x}_1, \quad \hat{x}_2, \quad \ldots \quad \hat{x}_n\right]^T = \left[x_1, \quad \frac{\xi_2}{\varepsilon}, \quad \frac{\xi_3}{\varepsilon^2}, \quad \ldots \quad \frac{\xi_n}{\varepsilon^{n-1}}\right]^T \tag{19}$$

CHAPITRE IV

RESULTATS DE SIMULATION

Chapitre IV

Résultats de simulation

I Introduction

Dans ce chapitre, nous présentons deux études en simulation du contrôleur neuronal adaptatif. Le premier exemple représente la commande en mode de poursuite du système pendule inversé et le deuxième représente un problème de régulation d'un réacteur chimique Pour les deux exemples on appliquera les deux structures de commande par retour d'état et retour de sortie.

II les résultats de simulation:

II 1 Exemple 1: le pendule inverse

Le pendule inverse avec une longueur variable $l(x_1)$ donnée par:

$l(x_1) = l_0 + l_1 \cos(x_1), l_1 = 0.5l_0$, g/l$_0$=10, $ml_0^2 = 1$.

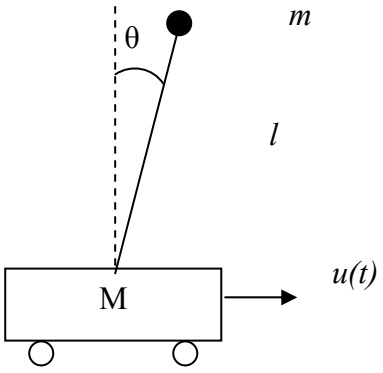

Figure IV.1 le pendule inverse

Le modèle mathématique non linéaire du pendule est :

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = \frac{(0.5\sin x_1(1+0.5\cos x_1)x_2^2 - 10\sin x_1(1+\cos x_1)) + u(t)}{0.25(2+\cos x_1)^2} \quad (1)$$

x_1 et x_2 sont respectivement l'angle du pendule avec la normale et la vitesse angulaire. Il y a deux groupes de paramètres qu'on va choisir pour contrôler ce système, ceux associés avec l'estimateur flou et les autres associés avec le réseau de neurones. Le FIS peut être construit tel que décrit dans le troisième chapitre pour fournir un estimé borné de l'erreur de commande $\hat{e}_u$. Tous les ensembles flous ont des formes triangulaires uniformément distribuées dans les univers de discours. Il ne reste qu'à choisir les paramètres du réseau de neurones.

a) Retour d'état:

Le vecteur d'entrée du réseau de neurones est:

$z = [x_1, \dot{x}_1 m, ke_s]^T$ est le vecteur d'entrée, (2)

$m = \lambda(\dot{x}_1 - \dot{y}_d) - \dot{y}_d$

$e_s = \lambda(x_1 - y_d) + (\dot{x}_1 - \dot{y}_d)$

$x_1, \dot{x}_1$ la sortie du système et sa dérivée respectivement.

$y_d, \dot{y}_d$ la sortie désirée et sa dérivée respectivement

et λ=10 $K = 5$

la fonction d'activation du contrôleur neuronal est choisie comme :

$$f(x) = \frac{1}{1+e^{-x}} \quad (3)$$

Les poids initiaux sont pris aléatoirement dan l'intervalle [0,1]

L'objectif de commande est de faire suivre au le pendule le signal de référence y_d

$y_d = (\pi/30)\sin(t)$ pour $0 \le t \le 25s$,

$y_d = (\pi/60)\sin(t)$ pour $25s \le t \le 60s$

$y_d=0$ pour $t \ge 60s$

Dans cette simulation la base de règles **BR1** est utilisée pour l'estimateur flou. Les taux d'apprentissage sont: $\delta_w = \delta_v = 1$ et $\sigma_w = \sigma_v = 0.01$.

b) Le retour de sortie

Le réseau de neurones contrôleur est le même que pour le retour d'état avec les mêmes valeurs de paramètres. Comme il est indiqué dans le paragraphe III.2 du chapitre précédent on choisit les paramètres de l'observateur non linéaire comme suit : supposons que la sortie *y(t)* et ses n dérivées sont bornées et considérons le système linéaire suivant :

$$\begin{aligned} \varepsilon\dot{\xi}_1 &= \xi_2 \\ \varepsilon\dot{\xi}_2 &= \xi_3 \\ \varepsilon\xi_3 &= -b_1 \cdot \xi_3 - b_2 \cdot \xi_2 - b_3 \cdot \xi_1 + y(t) \end{aligned} \qquad (4)$$

les paramètres b_1, b_2, b_3 sont choisis tels que le polynôme $H = S^3 + b_2 S^2 + b_3 S + 1$ ait toutes ses racines strictement dans le demi plan complexe gauche. Ici on choisit ces paramètres comme suit : *b1=1*, *b2= 2*, *b3=1*

Les pôles du polynôme H sont alors dans le demi plan complexe gauche et sont :

P_1= -1.7549 P_2= -0.1226 + 0.7449 i P_2= -0.1226 - 0.7449 i

c) Résultats

Les résultats de simulation pour le pendule inverse sont présentés par les figures IV.2 à IV.6 pour la commande en retour d'état et par les figures IV.7 à IV.12 pour le retour de sortie.

La figure IV.2 présente la sortie du système, la sortie désirée et l'erreur entre les deux, on voit bien que la sortie suit la sortie désirée et l'erreur est presque nulle dans l'intervalle $0 \le t \le 60s$ et elle s'annule après 60*s*, même pour le cas de contrôleur par retour de sortie figure IV.7 a poursuite est bien satisfaite est l'erreur est acceptable,

Le signal de commande pour les deux cas, figure IV.3 et figure IV.8 devient lisse rapidement et il est borné, l'erreur de commande estimée par le FIS et pratiquement nulle soit pour le cas de contrôleur par retour d'état, figure IV.4 ou par retour de sortie, figure IV.9,

Les poids synaptiques du contrôleur neuronal convergent vers des valeurs constantes dans les deux cas, figure IV.5 et figure IV.10 et ces constantes dépendent du signal de référence choisi. Enfin l'erreur filtrée est pratiquement nulle pour le retour d'état, figure IV.6et acceptable pour le retour de sortie, figure IV.11. La sortie de l'observateur suit bien le signal à observer, figure IV.12.

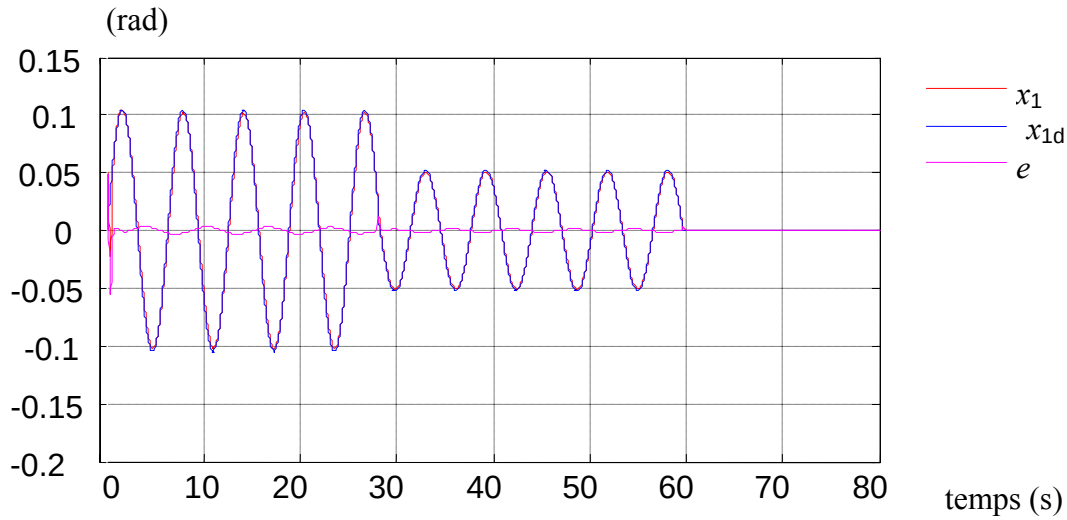

Figure IV.2 la position du pendule par retour d'état

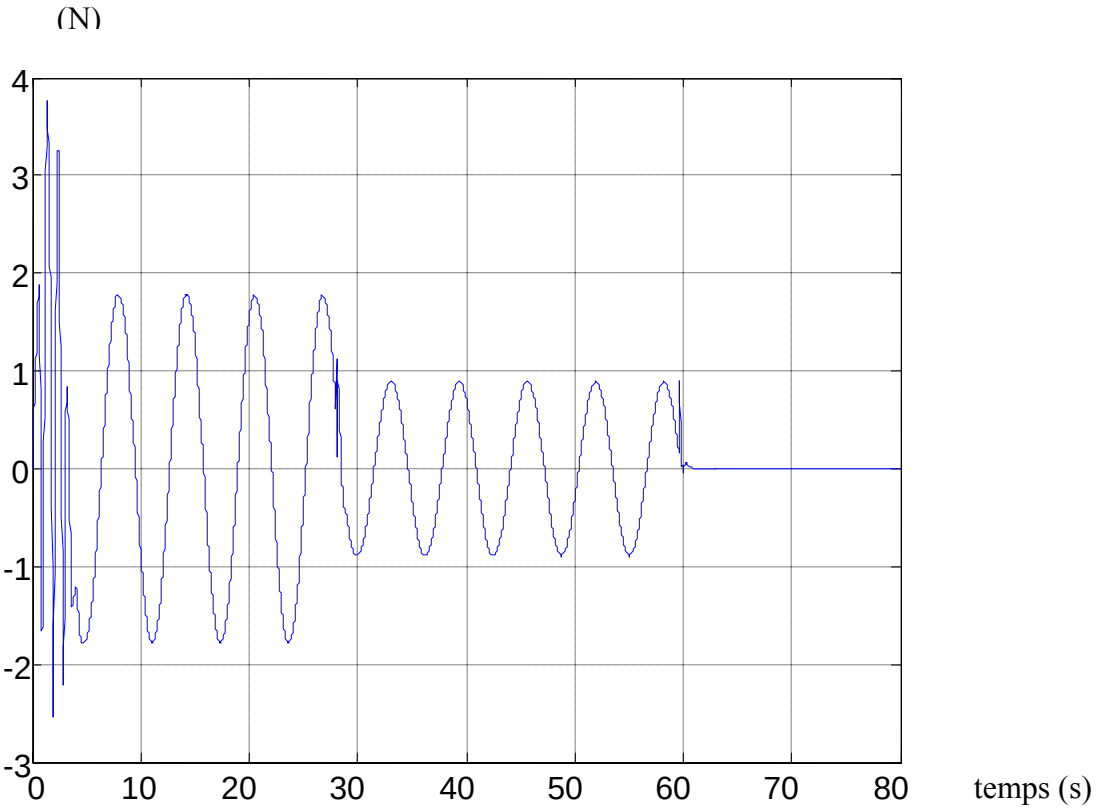

Figure IV.3 Le signal de commande de pendule pour le retour d'état

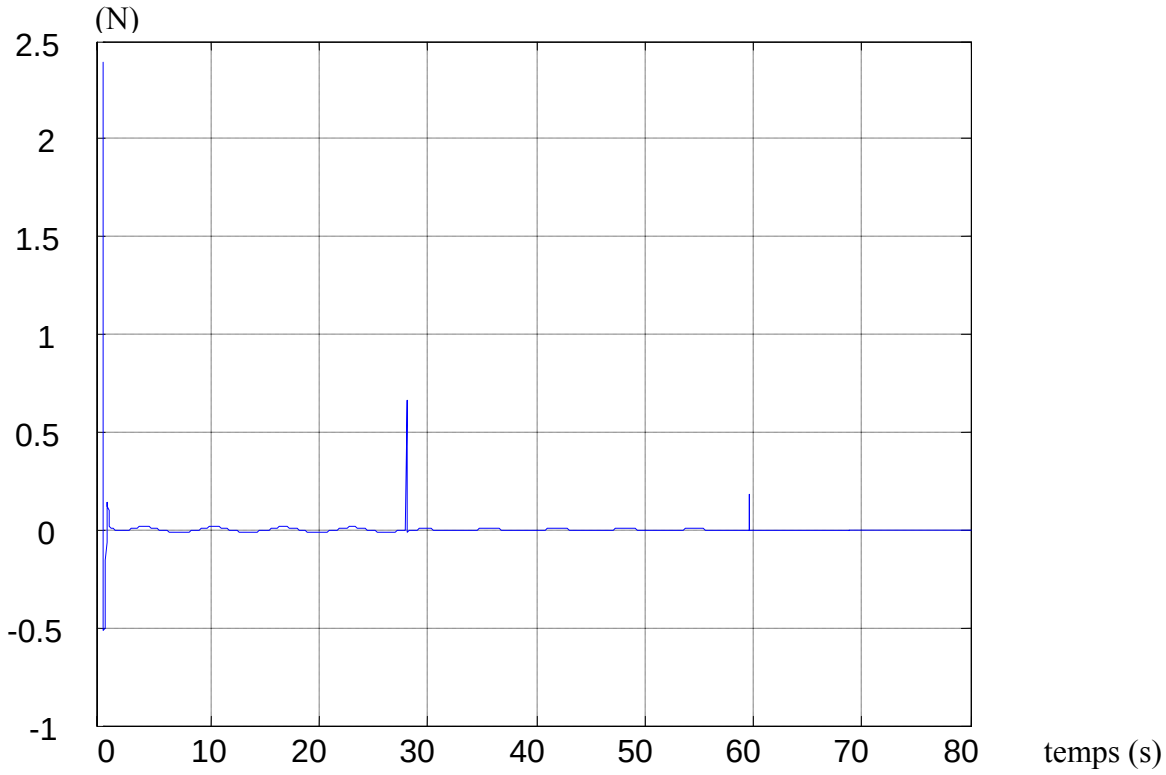

Figure IV.4 L'erreur estimée par le FIS pour le retour d'état

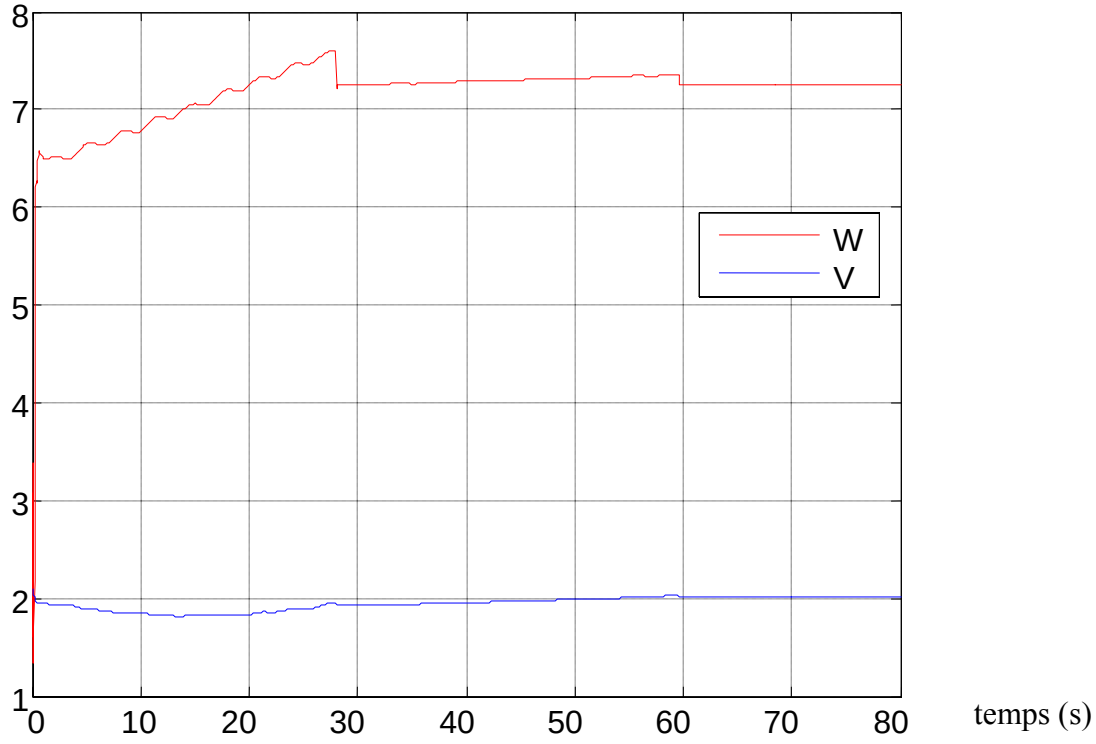

Figure IV.5 les poids du RN pour le retour d'état

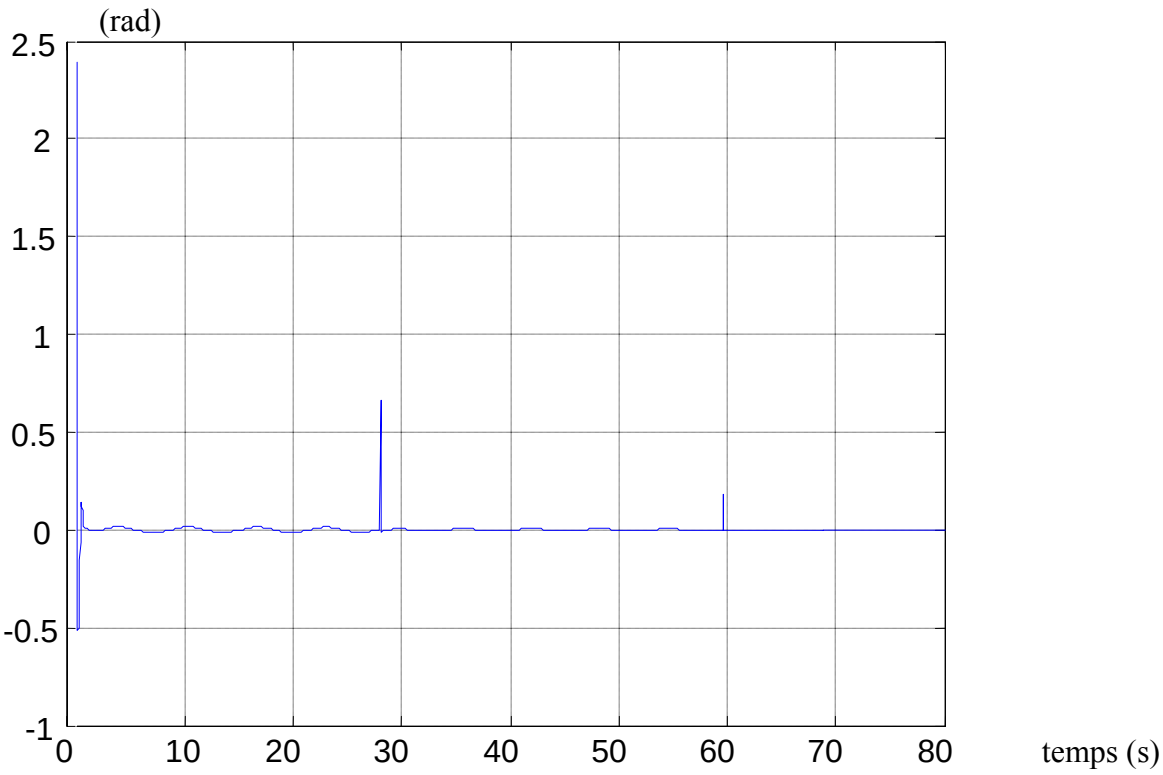

Figure IV.6 L'erreur filtrée es pour le retour d'état

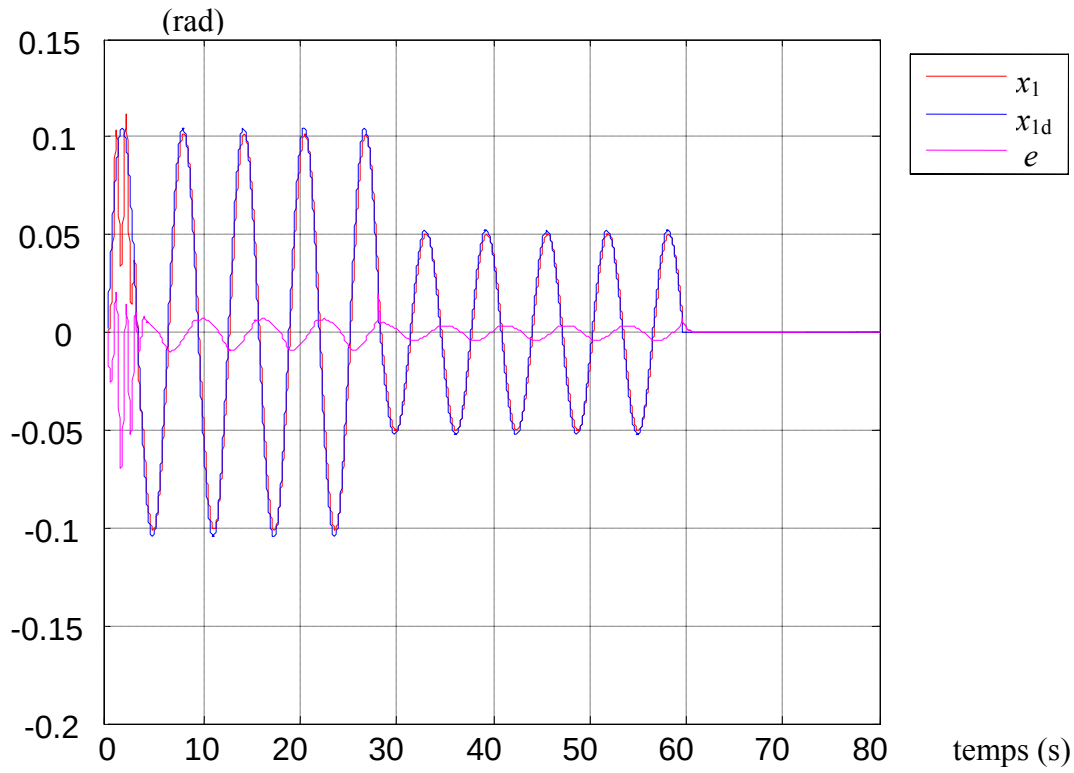

Figure IV.7 la position du pendule par retour de sortie

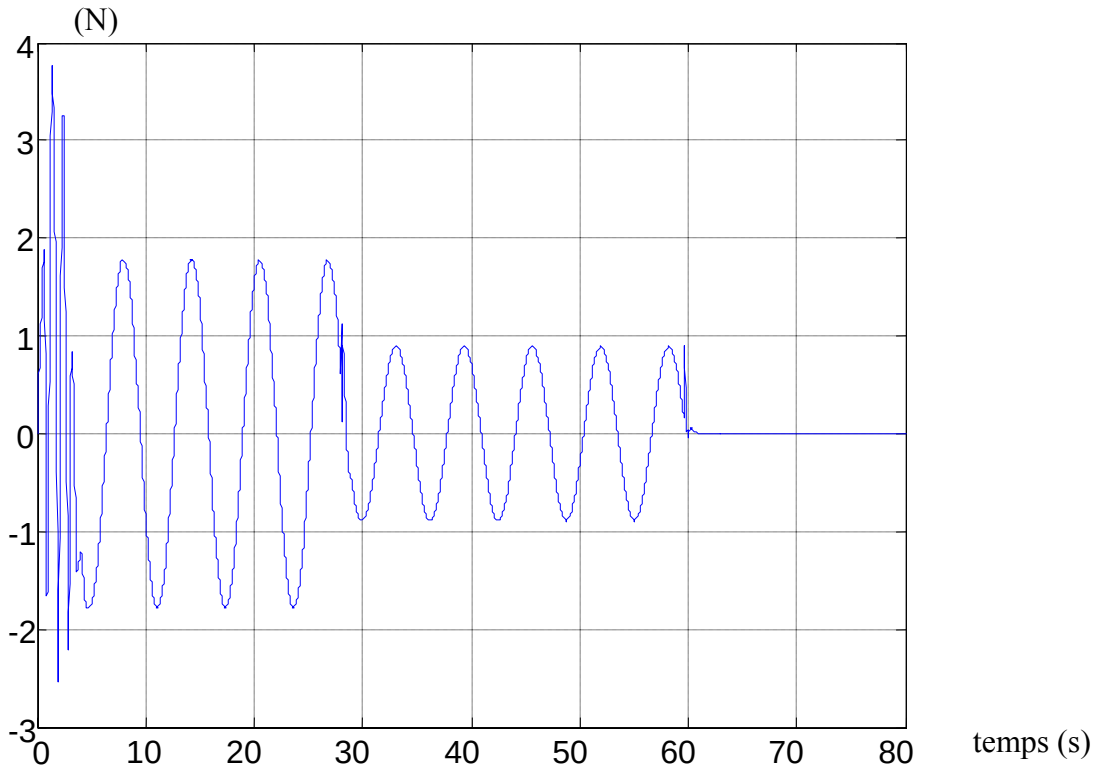

Figure IV. 8 Le signal de commande de pendule pour le retour de sortie

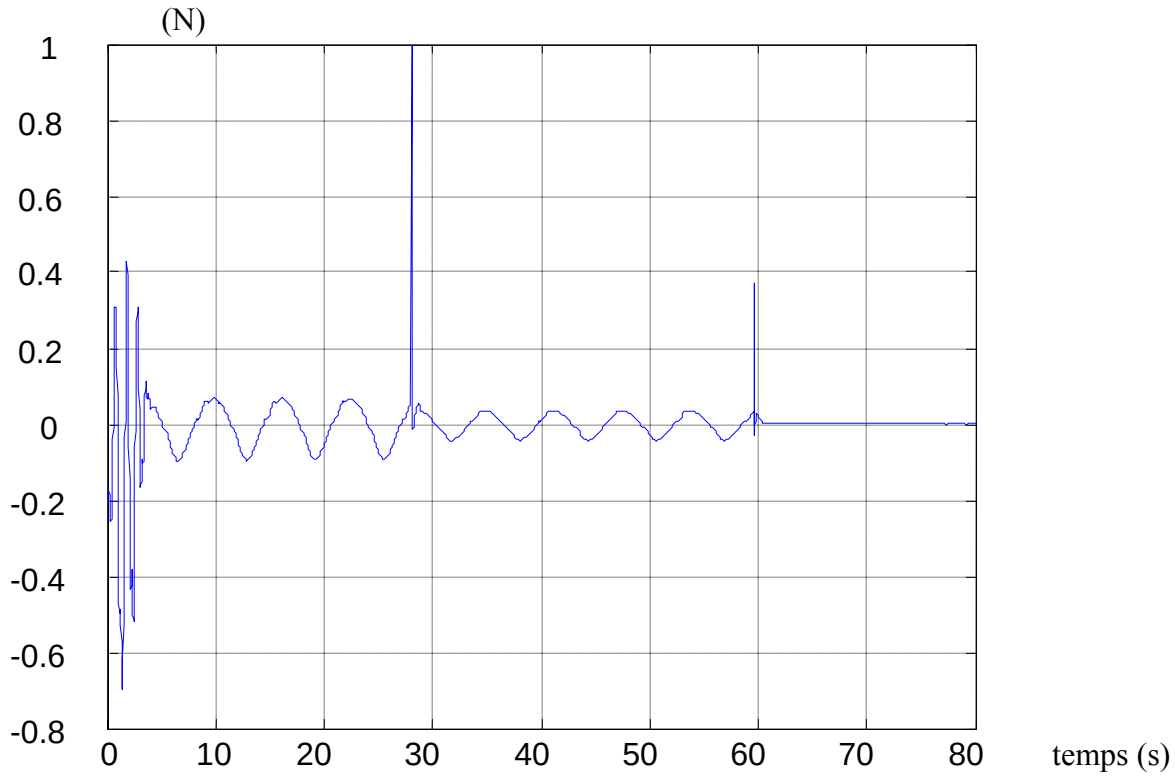

Figure IV.9 L'erreur estimée par le FIS pour le retour de sortie

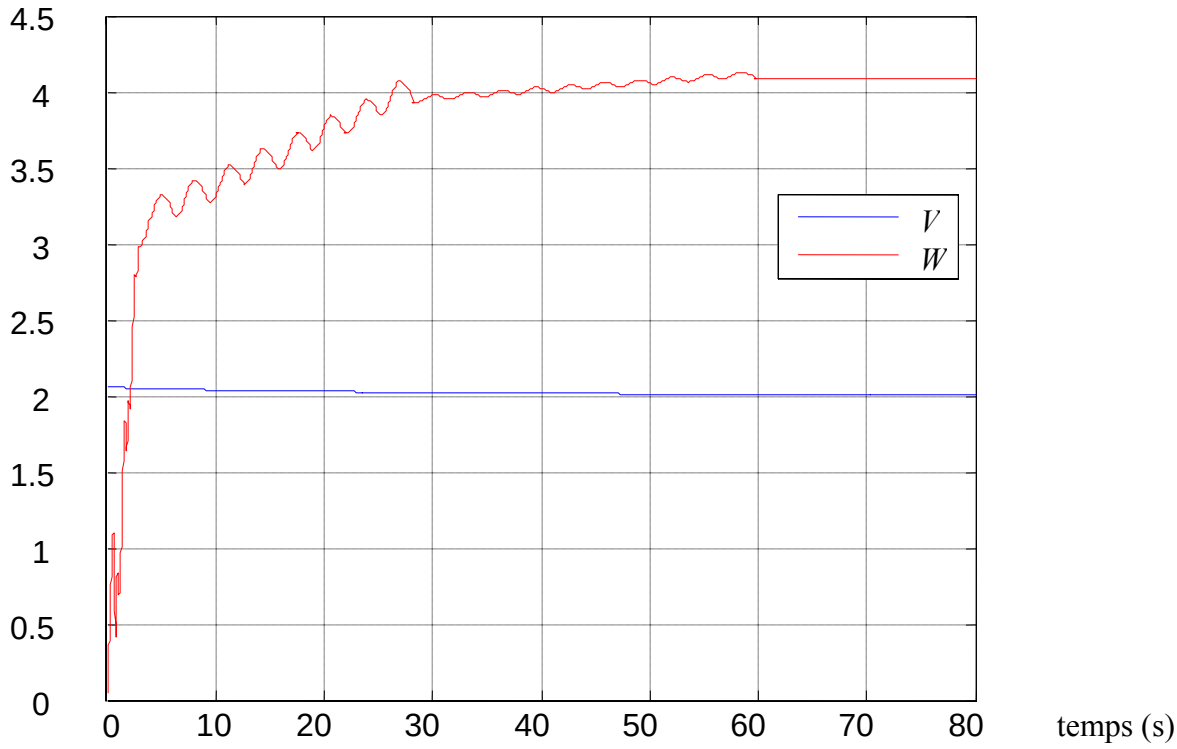

Figure IV.10 Les poids du RN pour le retour de sortie

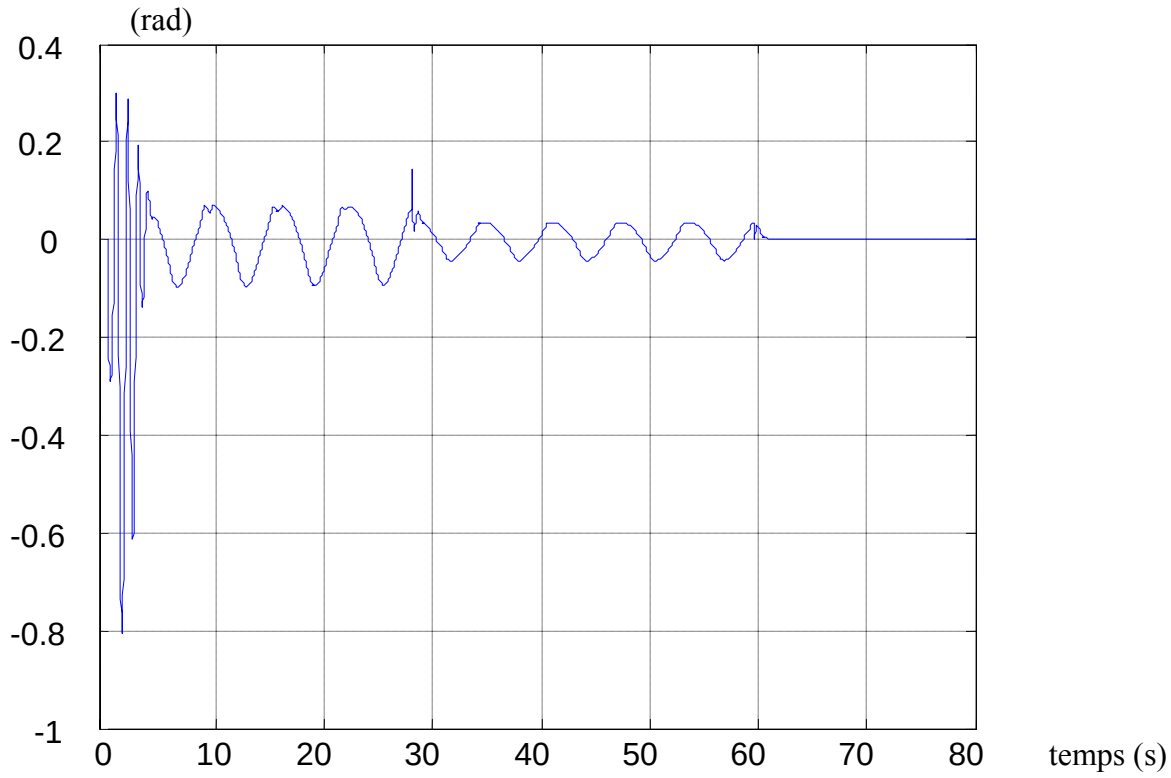

Figure IV.11 L'erreur filtrée par retour de sortie

Les sorties de l'observateur sont montrées dans la figure IV.12

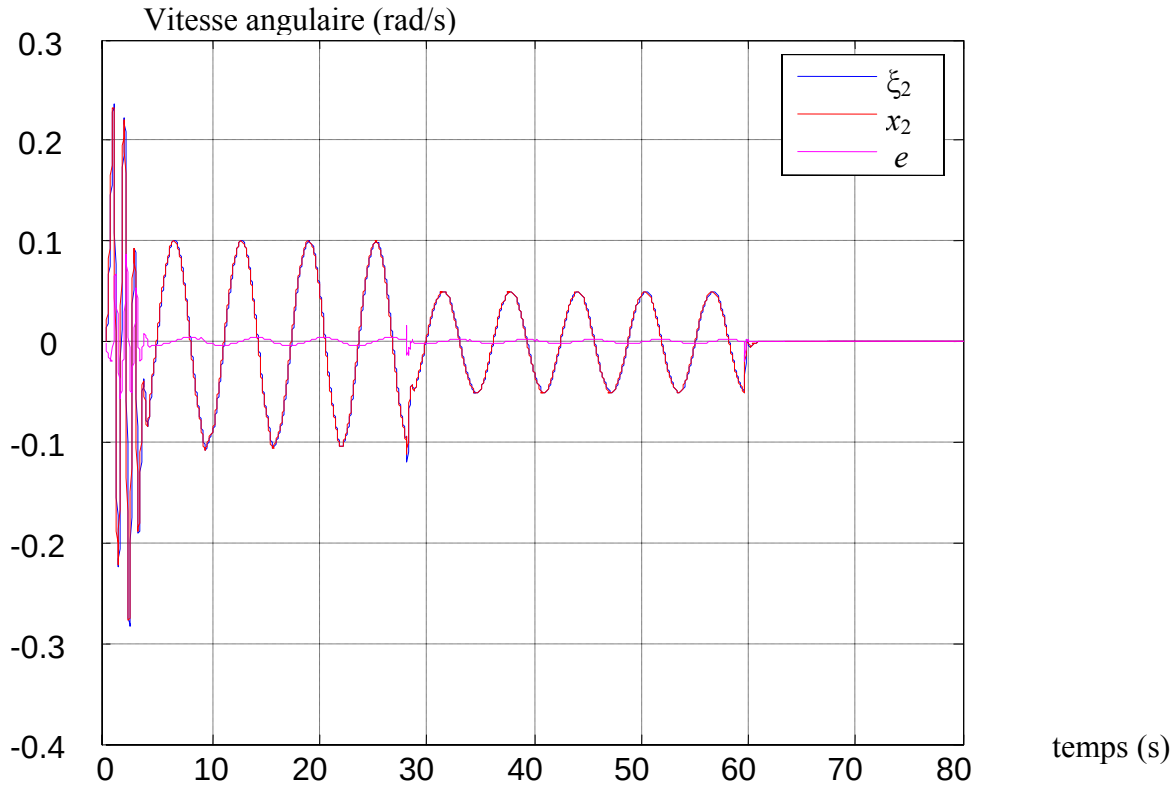

Figure IV.12 La sortie de l'observateur non linéaire

II 2 Exemple 2: Le CSTR

Le système du CSTR (Continuously Stirred Tank Reactor) consiste en un réacteur de volume constant. Une réaction exothermique, irréversible $A \rightarrow B$ se fait dans le réservoir; L'objectif est de contrôler la concentration d'effluent *Ca* en manipulant le débit du refroidissant. Le processus est modélisé par les équations différentielles suivantes:

$$\begin{aligned} \dot{C}_a &= \frac{q}{V}(C_{a0} - C_a) - a_0 C_a e^{-\frac{E}{RT_a}} \\ T_a &= \frac{q}{V}(T_f - T_a) + a_1 C_a e^{-\frac{E}{RT_a}} + a_3 q_c \left[1 - e^{-\frac{a_2}{q_c}}\right](T_{cf} - T_a) \end{aligned} \tag{3}$$

Où *Ca*, *Ta* sont la concentration et la température du réservoir, respectivement,
Le débit du refroidissant q_c est le signal de commande.

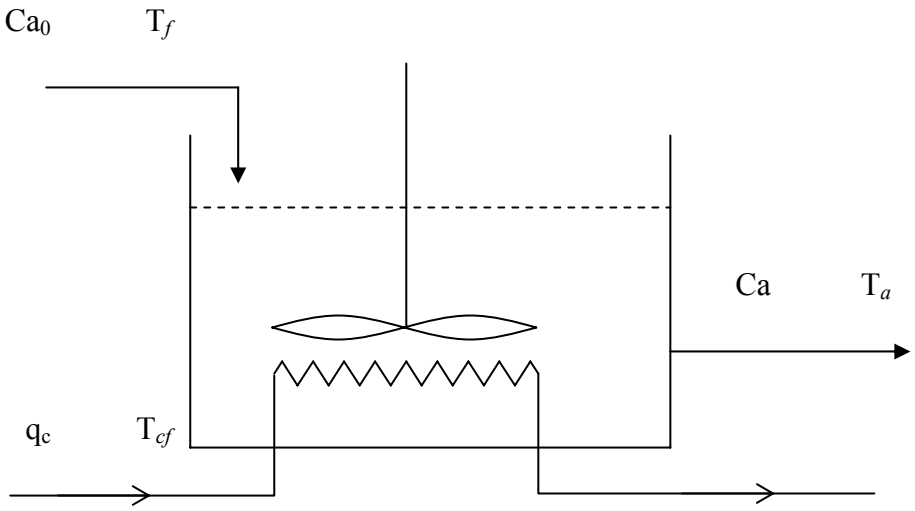

Figure IV.13 Le CSTR

On définit les variables d'état, l'entrée et la sortie comme suit:

$x = [x_1 \quad x_2] = [C_a \quad T_a]$

$u = q_c$

$y = C_a$

Donc le système peut être écrit sous la forme:

$$\dot{x} = f(x,u) = \begin{bmatrix} 1 - x_1 - a_0 x_1 e^{-\frac{10^4}{x_2}} \\ 350 - x_2 + a_0 x_1 e^{-\frac{10^4}{x_2}} + a_3 u \left(1 - e^{-\frac{a_2}{u}}\right)(350 - x_2) \end{bmatrix}$$

$$y = h(x) = x_1$$

Le tableau suivant nous montre les paramètres du CSTR

paramètre	description	
q	débit du processus	100 mol/l
C_{a0}	concentration du composant A	1 mol/l
T_f	température d'alimentation	350 K
T_{cf}	température du vapeur de refroidissement	350 K
V	Volume du réacteur	100 l
h_a	Taux de transfert de la température	7.10^5 J/min K
a_0	Facteur pré exponentiel	$7{,}2.10^{10}$ min^{-1}
E/R		2.10^4cal/mol
a_1		$1{,}44.10^{13}$
a_2		$6{,}987.10^2$
a_3		0.01

Tableau IV.1 paramètres du CSTR

a)Retour de d'état :

Pour ce système la base de règles **BR2** est utilisée. La fonction d'activation:

$$f(x) = \frac{1}{1+e^{-x}}$$

Les taux d'apprentissage: $\delta_w = \delta_v = 1$ et $\sigma_w = \sigma_v = 0.01$.

$e_s = \lambda(x_1 - y_d) + (x_2 - \dot{y}_d)$ et λ=10

L'objectif est de faire suivre à la concentration une référence lisse qui change autour du point de fonctionnent 0.1mol/l La référence $y_d(t)$ suit la loi suivante:

$\frac{yd(s)}{r(s)} = \frac{w_n^2}{s^2 + 2\zeta_n w_n s + w_n^2}$ avec *r(s)* prend les valeurs 0.12 et 0.08

b) Le retour de sortie

Comme on a fait pour le pendule inverse on choisit les paramètres de l'observateur non linéaire comme suit

$\varepsilon\dot{\xi}_1 = \xi_2$

$\varepsilon\dot{\xi}_2 = \xi_3$

$\varepsilon\xi_3 = -b_1 \cdot \xi_3 - b_2 \cdot \xi_2 - b_3 \cdot \xi_1 + y(t)$

Où les paramètres b_1, b_2, b_3 sont choisis tel que le polynôme $H = S^3 + b_2 S^2 + b_3 S + 1$ ait toutes ses racines strictement dans le demi plan complexe gauche. On choisit alors

b1=1 ; b2=2.5 ; b3=2.8;

Les pôles du polynôme H sont alors :

P_1= -1.6573 P_2= -0.4214 + 0.6526i P_3= -0.4214 - 0.6526i

c) Résultats

La figure IV.14 présente la sortie du système, la sortie désirée et l'erreur entre les deux. On voit bien que la sortie suit la sortie désirée et l'erreur est presque nulle dans l'intervalle $0 \le t \le 60s$ et elle s'annule après 60*s*, même pour le cas de contrôleur par retour de sortie, figure IV.7 la poursuite est bien satisfaite est l'erreur est acceptable,

Le signal de commande est borné pour les deux cas, figure IV.15 et figure IV.20 et devient lisse rapidement. L'erreur de commande estimée par le FIS et pratiquement nulle soit pour le cas de contrôleur par retour d'état, figure IV.16 ou par retour de sortie, figure IV.21.

Les poids synaptiques du contrôleur neuronal convergent vers des valeurs constantes dans les deux cas, figure IV.17 et figure IV.22. Ces constantes dépendent du signal de référence choisi. Enfin l'erreur filtrée est pratiquement nulle pour le retour d'état, figure IV.18 et acceptable pour le retour de sortie, figure IV.23. La figure IV.24 montre que la sortie de l'observateur suit bien le signal à observer.

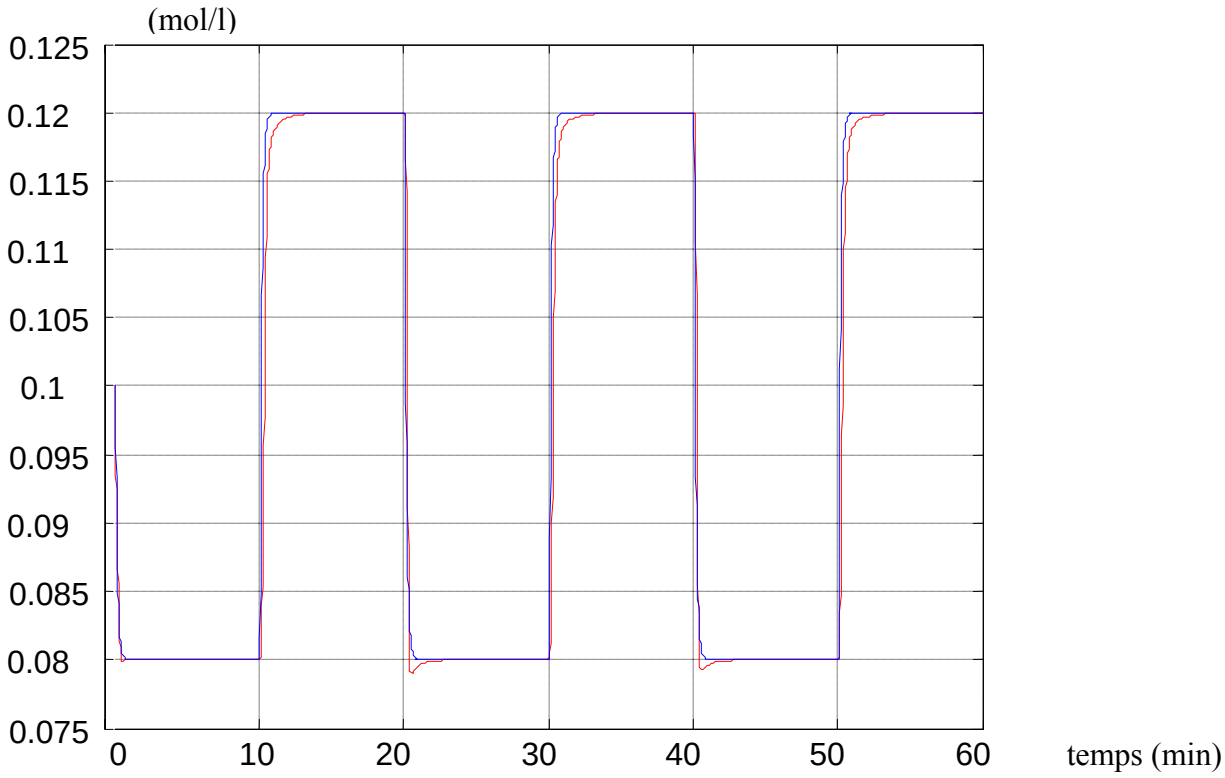

Figure IV.14 concentration d'effluent dans le CSTR par retour d'état

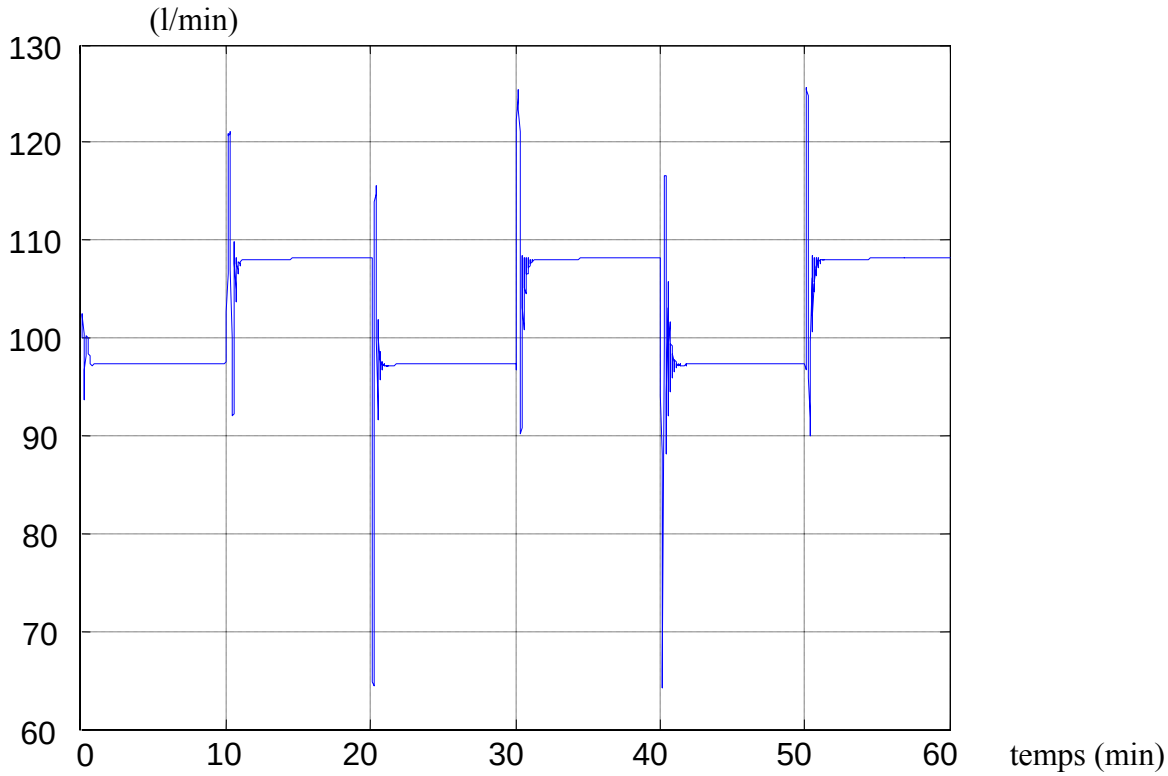

Figure IV.15 Le signal de commande (débit du refroidissant) pour le retour d'état

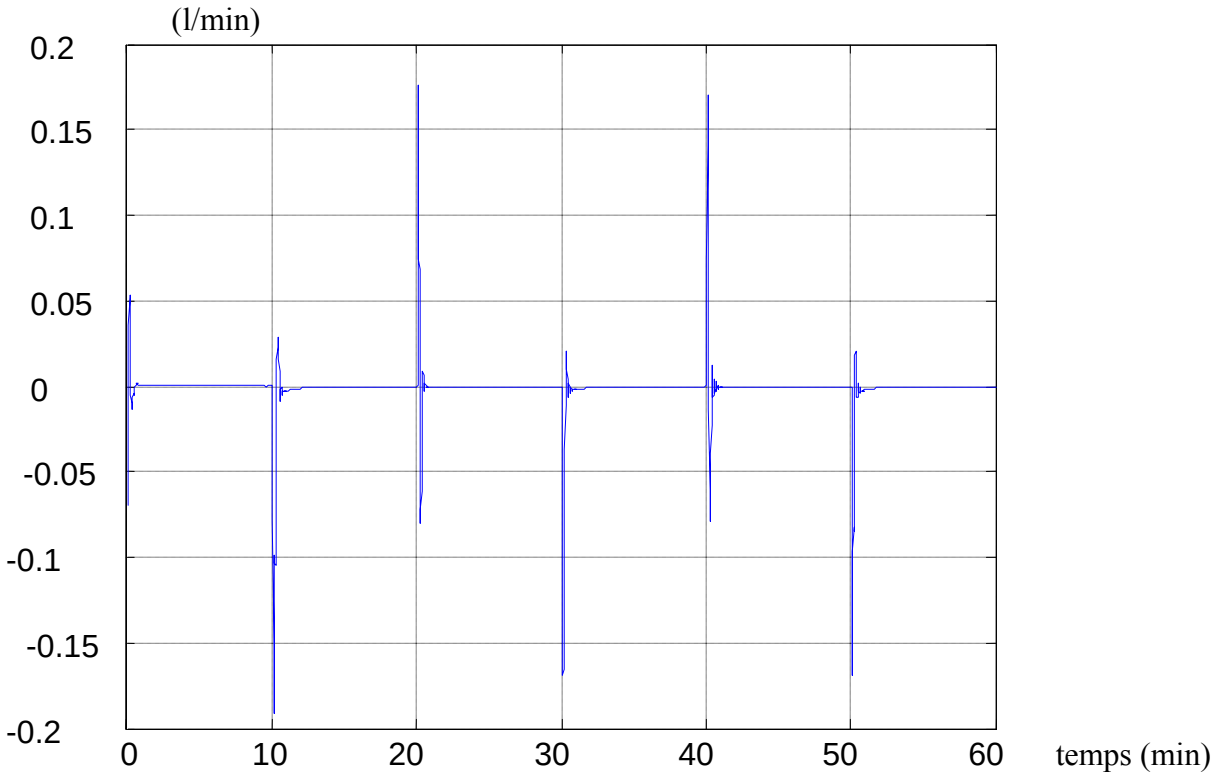

Figure IV.16 L'erreur estimée par le FIS pour le retour d'état

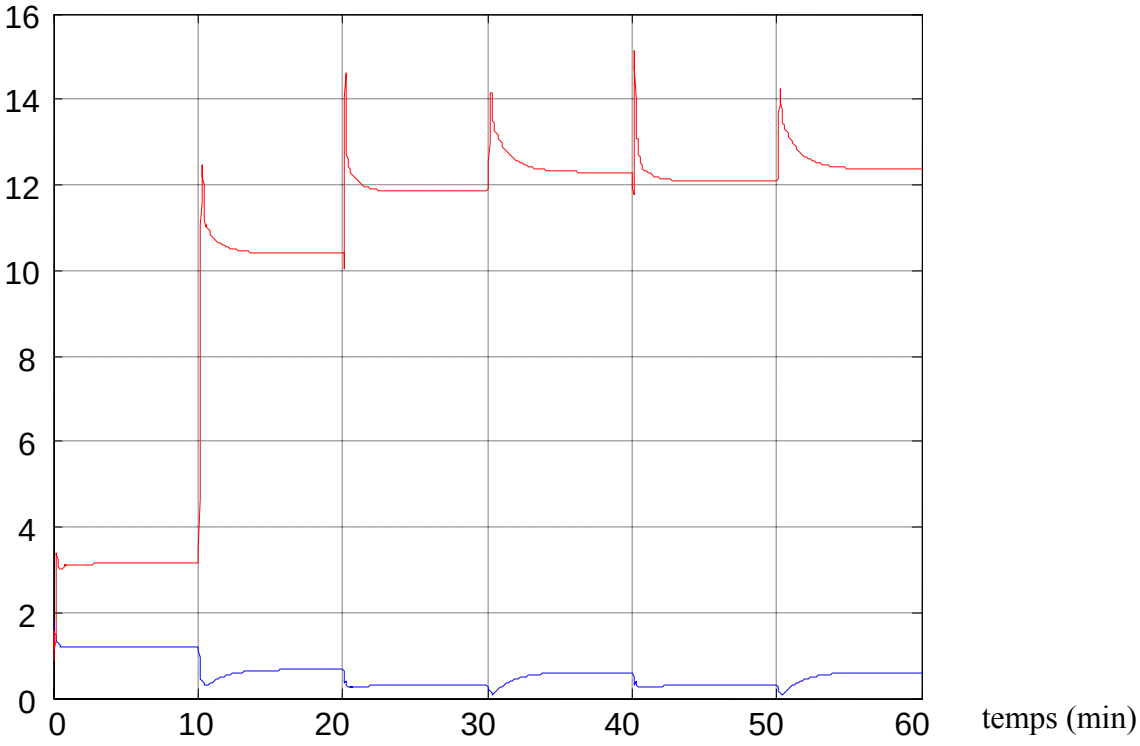

Figure IV.17 Les poids du RN pour le retour d'état

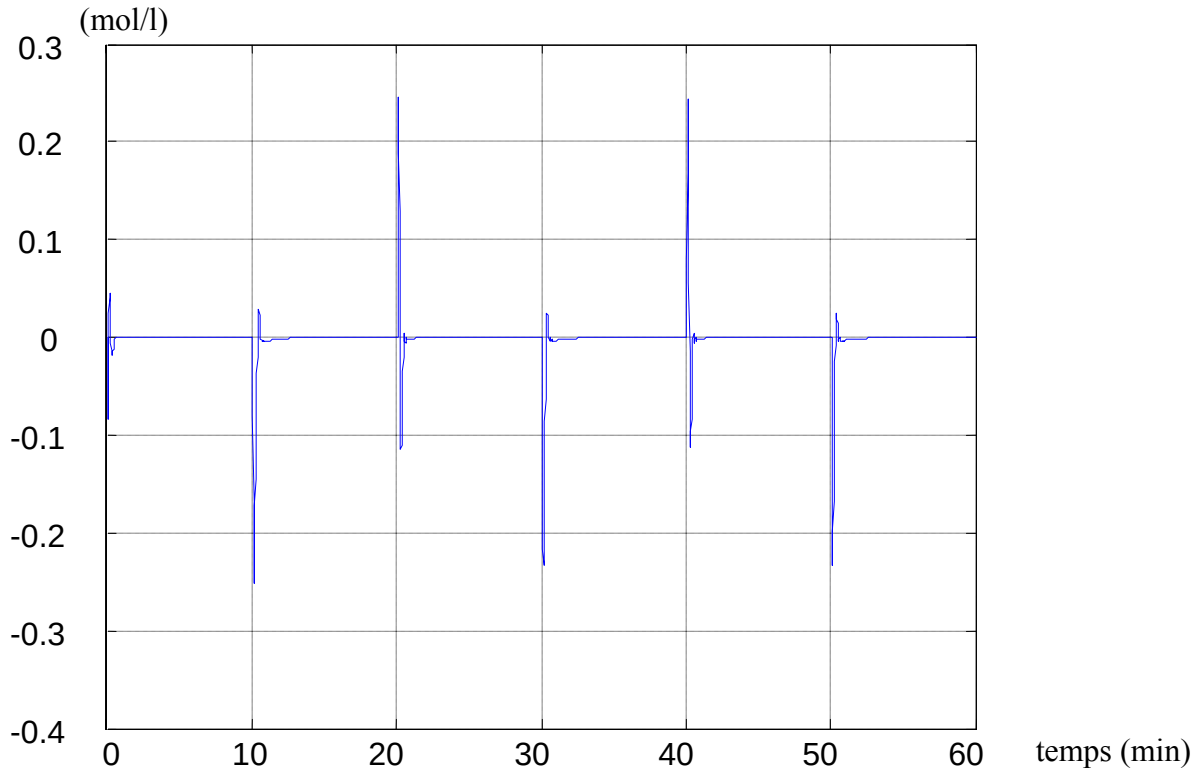

Figure IV.18 L'erreur filtrée es pour le retour d'état

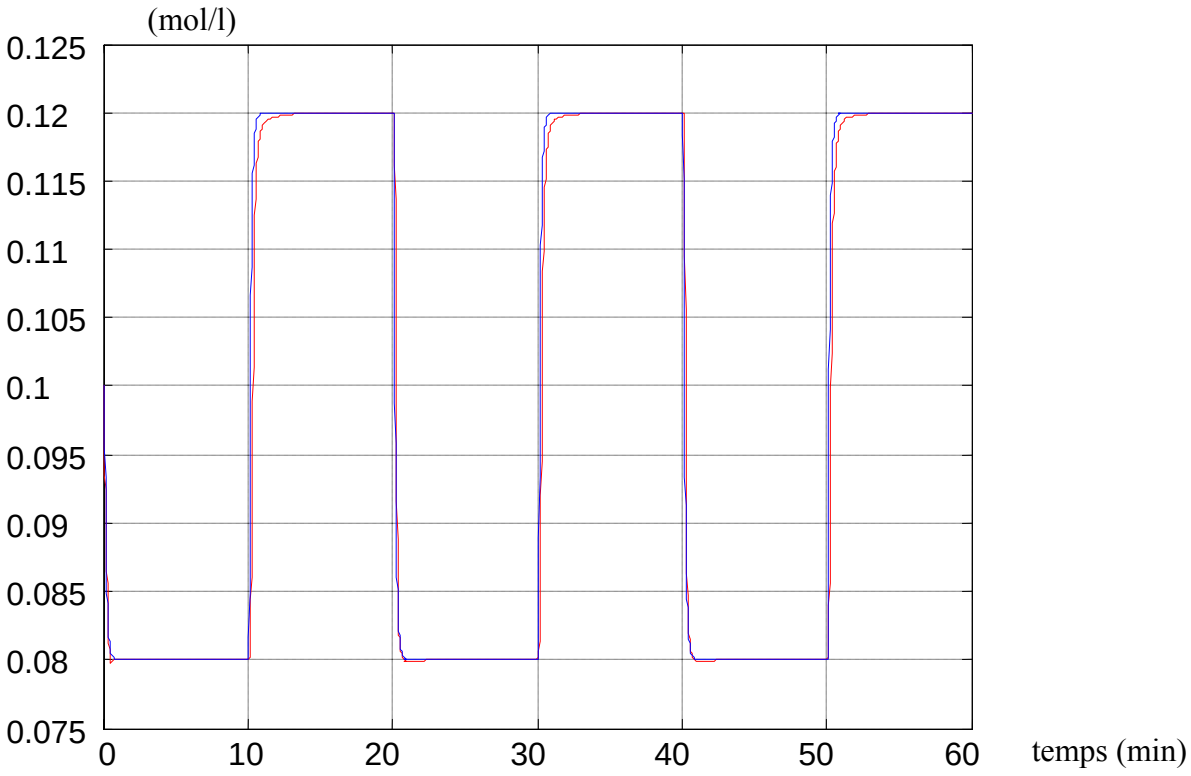

Figure IV.19 concentration d'effluent dans le CSTR par retour de sortie

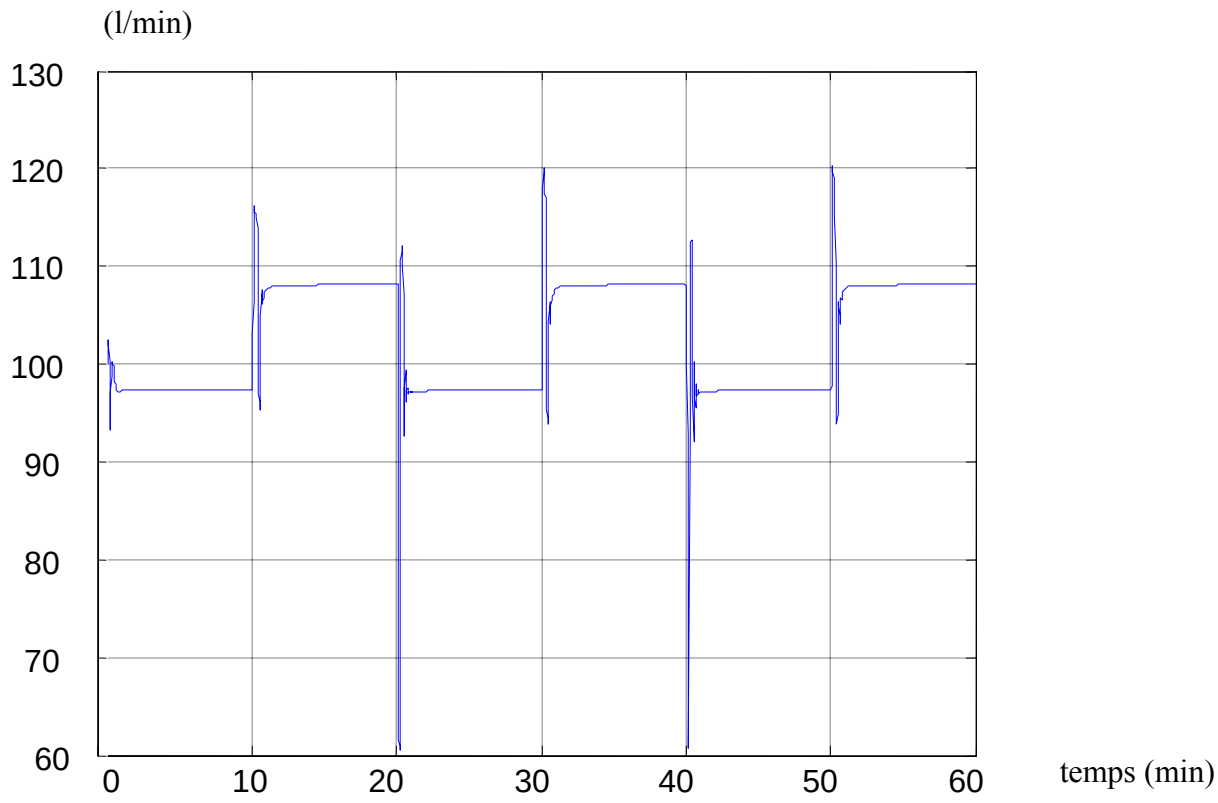

Figure IV.20 Le signal de commande (débit du refroidissant) pour le retour de sortie

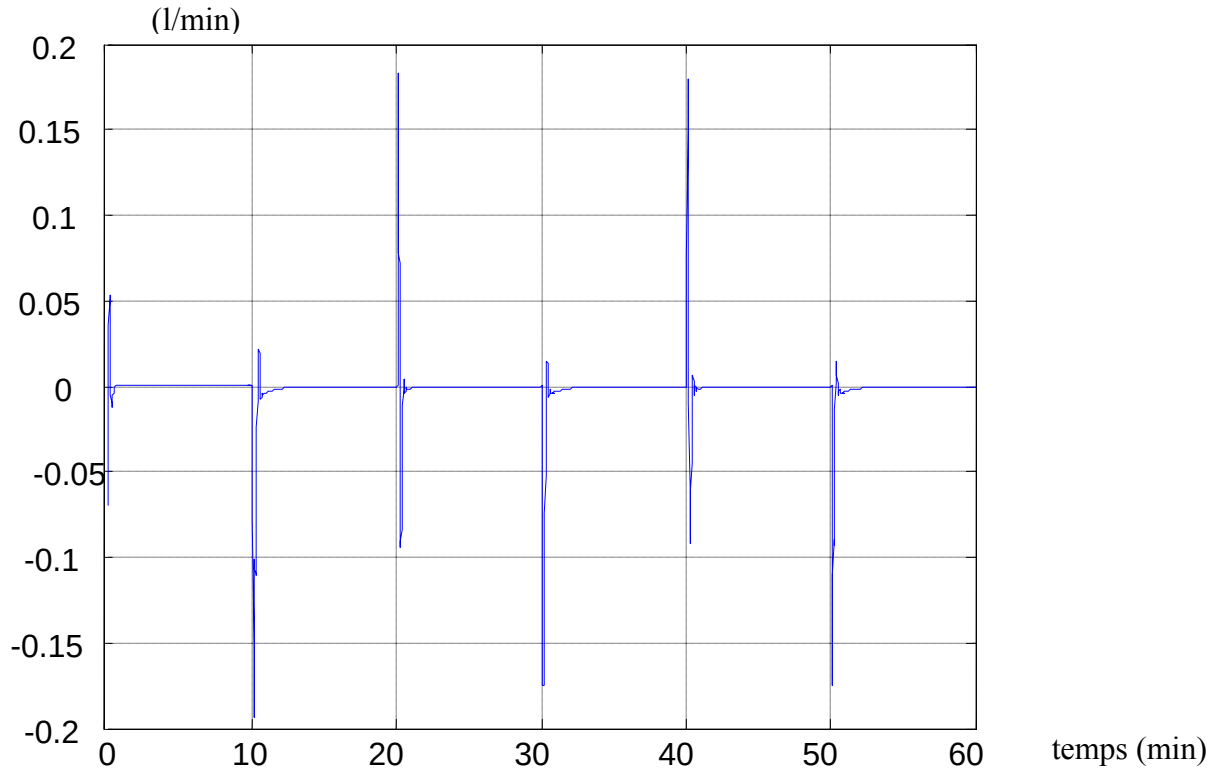

Figure IV.21 L'erreur estimée par le FIS pour le retour de sortie

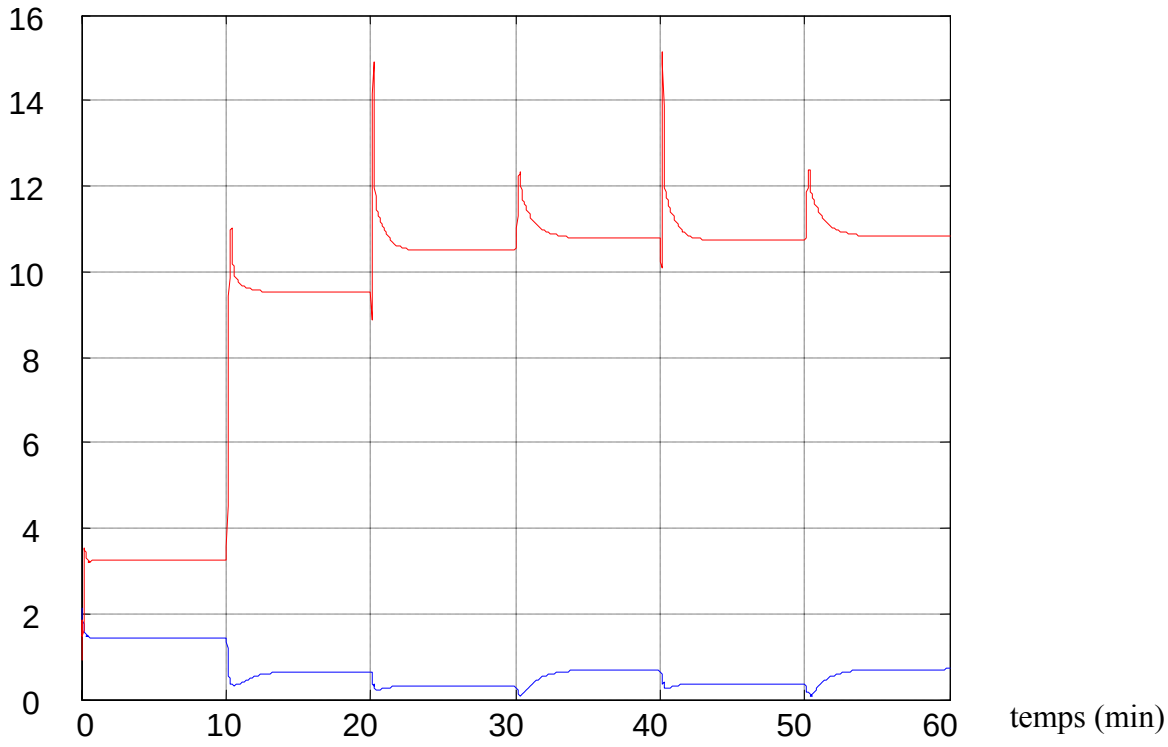

Figure IV. 22 Les poids du RN pour le retour de sortie

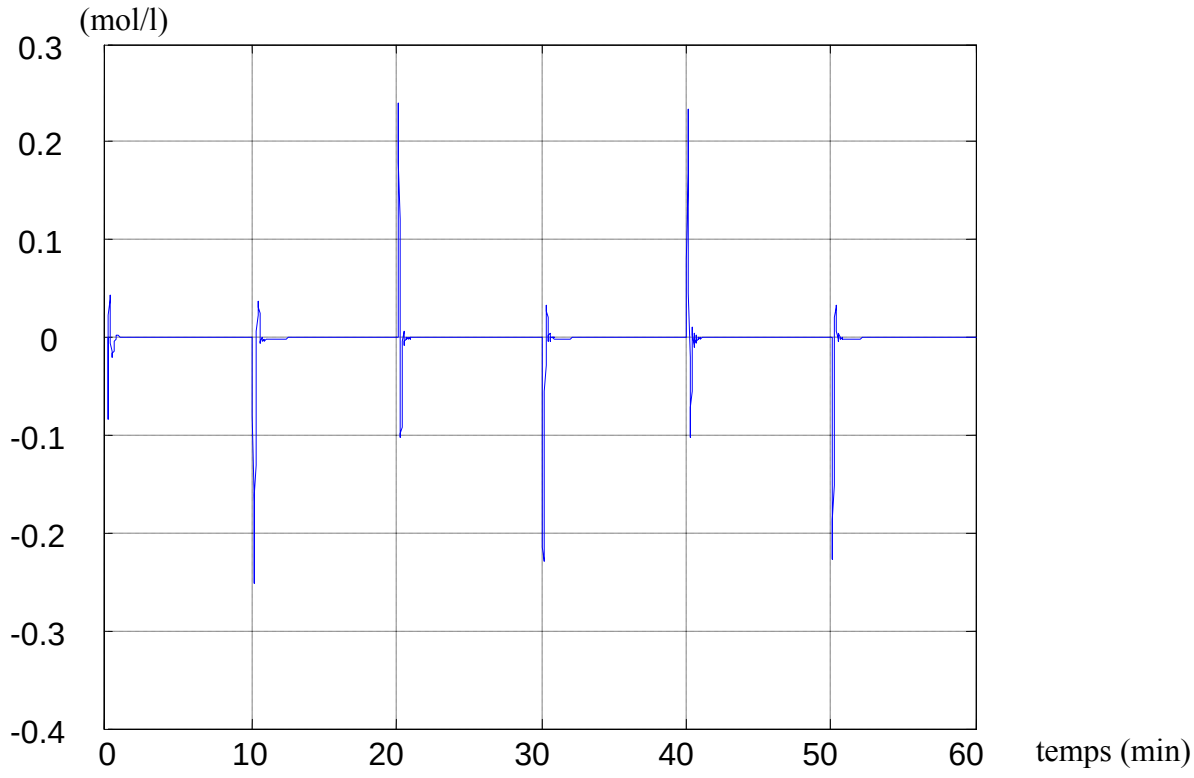

Figure IV.23 L'erreur filtrée es pour le retour de sortie

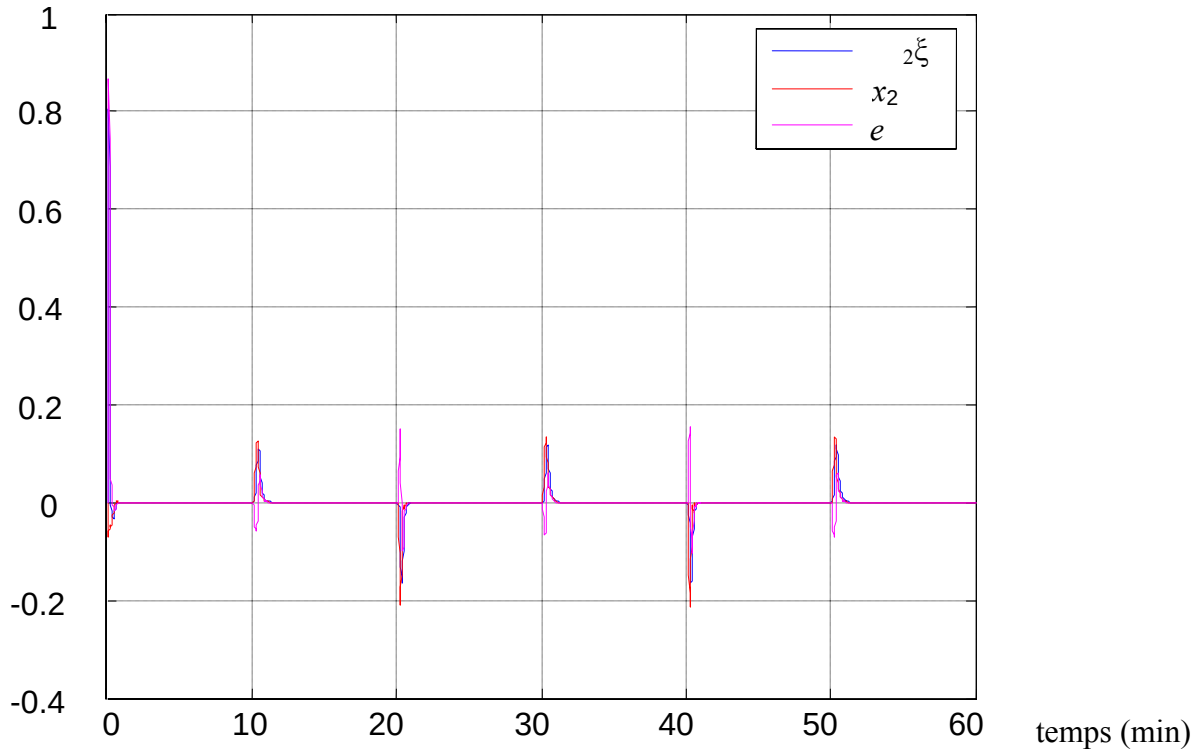

Figure IV.24 La sortie de l'observateur non linéaire

CHAPITRE V

CONCLUSION GENERALE

Chapitre V

Conclusion générale

Dans ce travail nous avons analysé un algorithme de commande adaptative neuronale permettant une adaptation directe des poids du réseau de neurones. L'algorithme est basé sur la minimisation du signal erreur sur la commande. La fonction à optimiser dépend alors directement des poids. Le signal de commande idéale ne pouvant être calculé, il est remplacé par une estimée obtenue par un système d'inférence flou. L'estimée ayant un signe correct, elle n'influe que sur le pas d'apprentissage. Une étude en simulation da été réalisée pour deux cas : une commande à retour d'état et une commande à retour de sortie basée sur un observateur non linéaire à gain élevée. Les résultats de cette étude ont montré les performances de la méthode pour les deux cas. Une perspective à ce travail qui sera entamée dans la thèse de doctorat sera d'étudier la stabilité de la méthode pour le cas de systèmes non linéaires sous forme générale.

ANNEXE

Annexe A

Systèmes d'inférence floue

Les systèmes à logique floue opèrent sur des variables linguistiques au lieu des variables numériques, ils fournissent un algorithme de conversion d'une stratégie d'inférence linguistique basée sur l'expertise humaine en une stratégie de contrôle automatique, il est décrit par un ensemble de règles de contrôle flou du type :

$$R_i : Si\ x\ est\ A_i\ et\ y\ est\ B_{i1}\ alors\ z\ est\ C_i$$

La structure classique d'un système d'inférence floue FIS (Fuzzy Inference system) est montrée par figure (A.1).

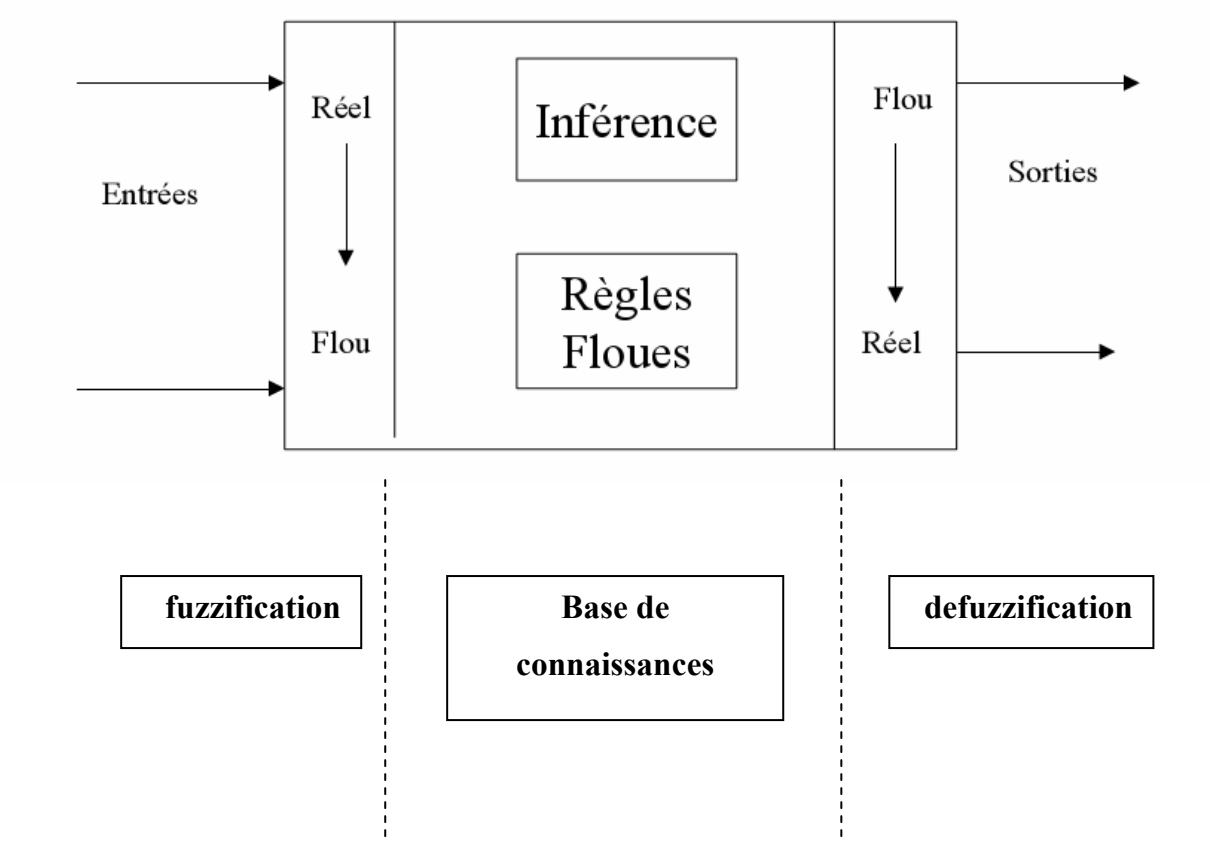

Figure (A.1). structure classique d'un FIS

Le FIS fonctionne selon trois étapes:

Interface fuzzification

L'interface fuzzification consiste à:

- ✓ Mesurer les valeurs numériques des variables d'entrées,
- ✓ Les projetées dans l'univers de discours en utilisant un facteur d'échelle,
- ✓ Transformer ces valeurs numériques en valeurs linguistiques convenable en utilisant des fonctions d'appartenances qui servent à subdiviser l'espace d'entrée, univers du discours en sous ensembles flous.

Base de connaissances

Elle contient les informations du domaine d'application et le but de contrôle, elle est définie par les deux principales bases suivantes:

base de données: Elle fournit des informations nécessaires utilisées pour l'exploration des règles d'inférence floue et la manipulation des données dans un FIS.

base de règles: Elle caractérise le but et la politique du contrôle flou via un ensemble de règles de contrôle flou.

Logique de décision

C'est le noyau du FIS; elle est capable de simuler les décisions humaines basées sur le concept flou et d'inférer des actions de contrôle flou par l'intervention de l'implication floue et des règles d'inférence dans la logique floue.

Défuzzification

Elle consiste à:

- ✓ Transformer les valeurs de la sortie comprissent dans l'univers du discours en valeurs réelles comprissent dans le domaine de variation,
- ✓ Extraire de la sortie de vérité la valeur numérique de la sortie.

BIBLIOGRAPHIE

REFERENCES

[1] Chen F.C. & Lin C.C. (1994). Adaptively controlling nonlinear continuous time systems using multilayer neural networks. *IEEE Transactions on Automatic Control, 39*, 1306-1310.

[2] Chen F.C. & Khalil H. K. (1995). Adaptive control of a class of non linear discrete-time systems using neural networks. *IEEE Transactions on Automatic Control, 40*, 791-801.

[3] Funahashi, K. L. (1989). On the approximate realization of continuous mapping by neural netorks. *Neural Networks,* 2, 183-192.

[4] Ge S. S. , Hang C. C. and Zhang T., A direct method for robust adaptive nonlinear control with guaranteed transient performance, Systems and Control Letters, 37, (1999), 275-284;

[5] Ge, S. S. & Wang, J. (2002). Robust adaptive neural control for a class of perturbed strict feedback nonlinear systems. *IEEE Transactions on Neural Network*, 13, 1409-1419.

[6] Hornik, K. Stinchombe M. & White H. (1989). Multilayer feedforward networks are universal approximators. *Neural netwoks* , *2*, 1083-1112.

[7] Isidori A. (1989). *Non linear control systems.* (2nd edition) Berlin:Springer.

[8] Lewis, F. L., Yesildirek, A. & Liu, K. (1996). Multilayer neural net robot controller with guaranteed tracking performance. *IEEE Transactions on Neural Network*, *7,* 388-399.

[9] Passino K. M. & Yurkovich, S. (1998). *Fuzzy Control.* Reading, MA: Addison-Wesley.

[10] Polycarpou, M. M. (1996). Stable adaptive neural control scheme for nonlinear systems. *IEEE Transactions on Automatic Contr*ol, *41,* 447-451.

[11] Sanner R. M. and Slotine J. E. (1992). Gaussian networks for direct adaptive control. *IEEE Transactions on Neural Networks, 3*, 837-863.

[12] Zhang, T. Ge, SS, & Hang, C. C. (1999). Design and performance analysis of a direct adaptive controller for nonlinear systems. *Automatica*, *35,* 1809-1817.

Résumé

Dans ce travail nous avons analysé un algorithme de commande adaptative neuronale permettant une adaptation directe des poids du réseau de neurones. L'algorithme est basé sur la minimisation du signal erreur sur la commande. Le signal de commande idéale ne pouvant être calculé, il est remplacé par une estimée obtenue par un système d'inférence flou.

L'estimée ayant un signe correct, elle n'influe que sur le pas d'apprentissage. Une étude en simulation da été réalisée pour deux cas : une commande à retour d'état et une commande à retour de sortie basée sur un observateur non linéaire à gain élevée. Les résultats de cette étude appliquée sur deux systèmes non linéaires le pendule inverse et le CSTR, ont montré les performances de la méthode pour les deux cas.

ملخص

في هذا العمل قمنا بدراسة خوارزمية للتحكم التكيفي المستند على الشبكات العصبونية التي تسمح لنا بالتحديث المباشر للمعاملات، هذه الخوارزمية تعتمد على تقليص الخطأ على إشارة التحكم. وحيث أننا لا نستطيع حساب إشارة التحكم المثالية فإن الخطأ على إشارة التحكم يقدر فقط عن طريق نظام يعتمد على المنطق المبهم.

حاملا لإشارة صحيحة فإن الخطأ المقدر لا يؤثر إلا على خطوة التمرن. تم تحقيق محاكاة من أجل حالتين للتحكم: الأولى تعتمد على رجوع حالات النظام، والثانية على رجوع مخارج النظام مع ملاحظ غير خطي ذي معامل كبير. نتائج الدراسة المطبقة على نظامين غير خطيين هما النواس المعكوس والمفاعل الكيميائي CSTR أثبتت فاعلية الطريقة المتبعة للتحكم.

Abstract

In this work we have analyzed an output feedback adaptive neural control allowing direct update of neural network weights. This algorithm is based on control error minimization. The ideal control signal cannot be calculated, so the error control is estimated by a fuzzy inference system. The estimate having the correct sign, it influences only the leaning rate. A simulation study is carried out for two cases: a state feedback control and output feedback control based on a nonlinear high gain observer. We applied this approach to two nonlinear systems the inverted pendulum and a CSTR (Continuously Stirred Tank Reactor). The results of simulation showed the performances of the strategy in the two cases.

.