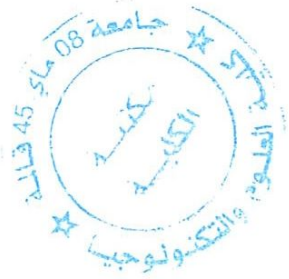
République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université 8Mai 1945 – Guelma
Faculté des Sciences et de la Technologie
Département de Génie Electrotechnique et Automatique



Mémoire de fin d'étude
pour l'obtention du diplôme de Master Académique

Domaine : **Sciences et Techniques**
Filière : Automatique et Informatique Industrielle
Spécialité : Commande et diagnostic des systèmes Industriels

Commande neuro-inverse d'un procédé industriel

Présenté par : ***ZEKRI Farouk***

Sous la direction de : ***M^eme LOUCIF Fatiha***

JUIN 2011

Remerciment

Je tien à remercier mon encadreur Mme Loucif de m'avoir dirigé, et pour avoir accepté de m'encadrer, au enseignants de notre département, à mes collègues, mes amis : hamza, hamdi, nabil, abdeslam, sofiane, anis … et à tout les gens qui étaient derrière moi durant mes études et mes formations.

Sommaire

II-C-9-3-Interprétation du résultat.....18

II-C-9-4-Conclusion.....18

CHAPITRE III. LA COMMANDE PAR RETOUR DETAT

III -A Introduction.....20

III -B Représentation d'état discret du four.....20

III -C- Commande d'un four placements des pôles.....21

III -C-1- DISCRETISATION DU PROCESSUS (FOUR).....21

III -C-2- Schéma global de la simulation.....25

III -C-3 RESULTAT DE SIMULATION.....26

III -C-4- INTERPRETATION DES RESULTATS.....26

III -C-5- CONCLUSION.....27

III -D Commande d'un four par retour d'état avec intégration.....27

III -D-1 SCHEMA GLOBAL DE LA SIMULATION.....27

III -D-2 RESULTAT DE SIMULATION.....28

III -D-3 INTERPRETATION DES RESULTATS.....29

III -D-4 CONCLUSION.....29

III -E- COMMANDE D'UN FOUR ELECTRIQUE VENTILE PAR RETOUR D'ETAT EN UTILISANT LE RECONSTRUCTEUR DE KALMAN.....30

III -E-1-INTRODUCTION.....30

III -E-2-PRINCIPE D'UTILISATION DU FILTRE DE KALMAN.....30

III -E-3- RESULTAT DE LA SIMULATION.....32

III -E-4- CONCLUSION.....33

CHAPITRE IV. RESEAUX DE NEURONES

IV-A- INTRODUCTION.....35

IV-B- RESEAUX DE NEURONES.....35

IV-B-1- NEURONE BIOLOGIQUE.....35

IV-B-2- NEURONE FORMEL.....36

IV-C- ARCHITECTURE DES RESEAUX DE NEURONES. 37

IV-C-1- LES RESEAUX NON BOUCLES. 37

IV-C-2- LES RESEAUX BOUCLES. 38

IV-D- APPRENTISSAGE. 38

IV-D-1- L'APPRENTISSAGE NON- SUPERVISE. 39

IV-D-2- L'APPRENTISSAGE PAR RENFORCEMENT. 40

IV-D-3- PRINCIPE DE L'APPRENTISSAGE SUPERVISE. 40

IV-D-4- Préapprentissage 41

IV-D-5- APPRENTISSAGE CONTINU. 41

IV-D-6- LA REGLE DE HEBB. 41

IV-D-7- LOI DE HOP FIELD. 42

IV-D-8- LOI DE DELTA. 42

IV-D-9- LE PERCEPTRON. 42

IV-D-10- L'ADALINE. ADAPTIVE LINE AR Elément. 43

IV-E- LES RESEAUX MULTICOUCHES ET L'APPROXIMATION DE FONCTIONS. 46

IV-E-1- LE PERCEPTRON MULTICOUCHE (MLP). 46

IV -E-1-1-STRUCTURE DU RESEAU. 46

IV-E-1-2- L'ALGORITHME DE LA RETRO PROPAGATION DU GRADIENT D'ERREUR (BACKPROPAGTION). 46

IV-F- PROPRIETES ET PROBLEMES. 49

IV-F-1- PROPRIETE D'APPROXIMATION UNIVERSELLE. 49

IV-F-2- PROBLEME DES VALEURS INITIALES DES POIDS DU RESEAU. 50

IV-F-3- PROBLEME DU SUR APPRENTISSAGE. 50

IV-F-4- PROBLEME DU REGLAGE DU PAS D'APPRENTISSAGE. 50

IV-G- APPLICATION DES RESEAUX DE NEURONES. 50

IV-G-1- IDENTIFICATION DE SYSTEME. 50

IV-G-2- PRINCIPE ET OBJECTIVE.....50

IV-G-3- LES DIFFERENTS TYPES DE MODELE.....51

IV-G-4- PROCEDURE D'IDENTIFICATION.....51

IV-G-5- IDENTIFICATION NEURONALE. 52

IV-G-6- PRINCIPE DE L'IDENTIFICATION PAR RESEAUX DE NEURONE. 53

IV-G-7- IDENTIFICATION PARALLELE.....53

IV-G-8- IDENTIFICATION SERIE PARALLELE.....53

IV-G-9- QUELQUES DOMAINES D'APPLICATION. 53

IV-G-10- INTERET DU NEURONAL EN IDENTIFICATION. 54

IV-H- COMMANDE NEURONALE. 54

IV-H-1- PRINCIPE. 54

IV-H-2- SOLUTION.....55

IV-H-3- COMMANDE SUPERVISEE. 55

IV-H-4- COMMANDE PAR MODELE DE REFERENCE. 56

IV-H-5- COMMANDE PAR MODELE INTERNE. 56

IV-H-6- COMMANDE PAR INVERSION DIRECTE.....57

IV-I- COMMANDE ET IDENTIFICATION PAR RESEAU DE NEURONE. 57

IV-I-1- IDENTIFICATION DU MODELE DIRECT. 58

IV-I-2- IDENTIFICATION DU MODELE INVERSE. 58

IV-J- CONCLUSION.....59

CHAPITRE V. COMMANDE DU FOUR PAR MODELE INVERSE NEURONAL

V-A- INTRODUCTION. 61

V-B- SYNTHESE D'UNE COMMANDE << NEURONALE>>. 61

V-C- PRINCIPE DE LA COMMANDE DU FOUR PAR MODELE INVERSE NEURONAL. 62

V-C-1- SCHEMA DE LA SIMULATION ENTRES SORTIE POUR L'APPRENTISSAGE DU RESEAU. 63

V-C-2- AFFICHAGE DES SIGNAUX SORTIE ET DE COMMANDE. 63

V-C-3- RESULTANTS DE LA SIMULATION. 66

V-C-4- INTERPRETATION DES RESULTATS. 67

V-D-COMMANDE INVERSE NEURONAL. 67

V-D-1- SCHEMA GLOBAL DE LA SIMULATION. 68

V-D-2- RESULTAT DE LA SIMULATION. 68

V-D-3- INTERPRETATION DES RESULTATS. 69

V-E- Conclusion 69

CONCLUSION GENERAL. 71

BIBLIOGRAPHIE. 73

Nomenclature

Q	quantité de chaleur produite
R_a	résistance thermique freinant la circulation de la chaleur du conduit jusqu'à l'enceinte du four,
C_a	capacité calorifique de l'enceinte du four
R_m	résistance thermique freinant la circulation de la chaleur du four à l'intérieur de la cavité de mesure,
C_m	capacité calorifique de la cavité de mesure
R_f	résistance de fuite freinant la circulation de la chaleur vers l'extérieur du four,
θ_a	températures de l'enceinte du four
θ_m	températures de la cavité de mesure
θ_e	Températures de l'extérieur du four
V_m	Température mesure image de la température θ_m
V_c	tension de commande
k_1	(watt/volt)
k_2	(volt/degrés)
Qmax	quantité de chaleur produite maximal
a_0 a_1 b_0	Des variables en fonctions des paramètres du four
Te	période D'échantillonnage
W_n	pulsation propre
ξ	facteur d'amortissement
{A, B, C, D}	Matrice du système continu
A_d, B_d, C_d, D_d	Matrice du système discrétise
Com	la matrice de commandabilité
Det	déterminant de la matrice Com
K	vecteur d'état
p_1 et p_2	Ce son les pôles du système discrétisation
A1, B1, C1,	Matrice d'état du système asservi
K1	Vecteur de retour d'état avec intégration
g	Terme de gain
y_c	La consigne du système
y	La sortie du système
w(k)	est le bruit de modélisation lié à l'incertitude que l'on a sur le modèle de processus
Q	Étant la matrice de variance
v(t)	est le bruit de mesure également considéré comme blanc, gaussien et centré.
R	Étant la matrice de variance
L(k)	matrice de gain d'adaptation du filtre
x_i	représentent le vecteur d'entrée

w_{ij}	sont les poids synaptiques du neurone j .
z(t)	potentiel somatique
σ	La fonction d'activation
α(t)	coefficient d'apprentissage
w_{ij}	le poids synaptique reliant l'entrée xi au neurone j , et yj la sortie du
$\delta_j(t)$	l'erreur faite par le neurone
d(t)	la sortie désirée
σ	l'identité pour l'ADA LINE
n	nombre d'exemples présentés
d(k)	la somme des écarts entre la distance
x(k)	vecteur d'entrée
d(k)	la sortie désirée
y(k)	la sortie réellement fournie par le neurone
ε	la fonction d'erreur
d_1,d_2 et d_3	les sorties désirées
δ_1, δ_2 et δ_3	les erreurs
w_{14} , w_{24}, w_{25} et w_{35}	des poids
δ4 et δ5	les erreurs
α(t)	pas d'apprentissage
$\xi(t)$	un bruit blanc
[w, b]	une commande
(x, t)	solvelin
u (k)	sont la séquence des entres de commande
u(t), y(t), Δu(t)et Δy(t)	les différents signaux
R NA	Réseaux de neurone artificiel
N et DN	normalisation et de dé normalisation des signaux
J (θ)	critère
$\hat{y}$ (t)	la sortie estimée
u (t)	la commande
H(z)	Modèle Arma
W	des poids
b	du biais
W et Z	Les Matrice des Poids
w_0 et z_0	Les vecteurs biais

liste des figures

Fig.I. 1 Processus d'un four ventilé [1]..... 5
Fig.I. 2 Schéma électrique du four [1]..... 5
Fig.I. 3 Schéma fonctionnel du four..... 6
Fig.I. 4 Commande d'un four modèle analogique..... 7
Fig.I. 5 Réponse du four à un échelon de tension..... 7
Fig.I. 6 Réponse du four à un échelon de température..... 8
Fig. II.1 schéma fonctionnel..... 11
Fig. II.2 amélioration la réponse du système..... 11
Fig. II.3 représentation de Black..... 12
Fig. II.4 comportement temporel..... 13
Fig. II.5 Schéma global de simulation de la commande intégral..... 17
Fig. II.6 Réponse du four avec un temps de réponse..... 18
Fig. II.7 Signal de commande..... 18
Fig. III.1 Représentation d'état discret du four..... 20
Fig. III.2 Commande d'un Four Placement de pôles..... 25
Fig. III.3.4 Résultat de simulation..... 26
Fig. III.5 Commande d'un Four par retour d'état avec intégration..... 27
Fig. III.6.7 Résultat de simulation..... 29
Fig. III.8 Estimation de l'état d'un processus..... 31
Fig. III.9 Résultat de simulation..... 32
Fig. IV.B.1 Schéma simplifié d'un neurone..... 36
Fig. IV.B.2 Neurone formel..... 36
Fig. IV.B.3 Fonctions d'activations..... 37
Fig. IV.B.4 mise en correspondance neurone biologique /neurones artificiel..... 37
Fig. IV.B.5 Réseau non bouclé.38
Fig. IV.B.6 Réseau bouclé..... 38
Fig. IV.B.7 Principe d'apprentissage..... 42

Fig. IV.B.8 Ou Exclusif. 43

Fig. IV.B.9 Présentation des écarts entre d(k) et y(k). 44

Fig. IV.B.10 Réalisation d'un séparateur linéaire. 45

Fig. IV.B.11 Algorithme de Rétro propagation. 46

Fig. IV.B.12 Procédure d'identification. 52

Fig. IV.B.13 identification parallèle est série parallèle. 53

Fig. IV.B.14 commande de processus par un réseau de neurones. 54

Fig. IV.B.15 La commande supervisée. 55

Fig. IV.B.16 La commande neurone adaptatif. 56

Fig. IV.B.17 Apprentissage d'un contrôleur avec modèle de référence. 56

Fig. IV.B.18 Commande avec modèle interne. 57

Fig. IV.B.19 Commande par inversion directe. 57

Fig. IV.B.20 La commande par modèle inverse. 57

Fig. IV.B.21 Identification du modèle direct par RNA. 58

Fig. IV.B.22 Identification du modèle inverse par R NA. 59

Fig. V.1. Principe de l'apprentissage et d'une commande neuronale. 61

Fig. V.2. Contrôleur neuronal. 62

Fig. V.3. Entrées sorties pour l'apprentissage du réseau. 63

Fig. V.4. Affichage des signaux (de sortie et de commande). 64

Fig. V.5. Commande par modèle inverse neuronal du four [1]. 64

Fig. V.6. Principe de réduction du critère $j(\theta)$. 64

Fig. V.7. Evolution des poids et du biais. 66

Fig. V.8. Schéma global de la simulation. 68

Fig. V.9. Sortie du système avec un gain k = 0.1.et k=0.2. 68

Fig. V.10. Sortie du système avec un gain k = 0.1. 68

INTRODUCTION GENERALE

INTRODUCTION GENERALE

Les fours électriques à résistances constituent les équipements **[1][2][3][4]** Électrothermiques industriels les plus connus et les plus répandus Le principe de ce type de four est extrêmement simple : il est constitué d'une enceinte chauffé à l'aide de résistances électriques, très bien calorifugée pour réduire le plus possible les déperditions thermiques. La charge à chauffer est placée dans cette enceinte.

Le four à résistances est un équipement à chauffage indirect, la chaleur produite par effet joule par les résistances est transmise à la charge par rayonnement .et convection. Ils continuent à faire l'objet de nombreux perfectionnements, pour les faire évoluer afin de mieux répondre aux besoins exprimés par les industriels utilisateurs

La mise en ouvre des résistances, le transfert thermique **[5]** entre les éléments chauffants et la charges et l'exploitation et la conduite de ces fours .la robustesse et les fiabilités de ces équipements leur facilite de mise en ouvres la simplicité de maintenances et leur efficacité énergétiques élèves font que les applications industrielle sont fort nombreuses il n'est guère de secteur industriel qui y fasse appel

On peut contrôler la température d'un four il est nécessaire d'analyser le système" four" et le modéliser sous forme d'équations .on pourra ainsi précisément relier l'entrée du système (une tension commandant la température) à la sortie (la température voulue du four).

Cette relation peut ce faire sous forme d'une équation différentielle ou d'une fonction de transfert ou bien on modifiant le schéma fonctionnel du four afin de donner une représentation d'état .On détermine aussi les conditions de stabilité du système (on ne veut pas que le four se mette à augmenter la température sans s'arrêter).O n va ensuite synthétiser un nouveau système, le "régulateur " ,celui –ci –aura pour entrées la consigne(c'est-à-dire la température souhaiter à l'intérieur du four) ainsi que la température réelle du four fourni par un capteur et pour sortie la commande du four .les deux systèmes "régulateur "et" four" sont mis en cascade. L'ensemble forme ce qu'on appelle un système asservi .celui ci doit répondre à un certain nombre d'exigence :

- La stabilité (le régulateur ne doit pas rendre le système instable).
- La poursuite (la température du four doit atteindre la température en consigne).
- Le rejet des perturbations.

Les régulateur classique ne sont pas adapté aux processus thermique ils sont conçu pour des processus linéaire sans retard .or un four est un système non linéaire. On est également souvent en présence d'un retard pur entre le chauffage des résistances et la montée en température consécutive de la charge pour ces déférentes raison avec les nouvelles méthodes récemment apparus en automatiques des recherches sur la régulation des fours sont entreprises recherches qui s'orientent vers l'identification et la régulation en modes neuronal.

A cet effet, ce travail est diviser en cinq chapitres tel que :

- chapitre I: modélisation du four ventilé : ce chapitre traite le modèle mathématique du four ainsi que son schéma fonctionnelle.
- chapitre II: la régulation des fours électrique a résistance il traite la régulation en utilisant le correcteur proportionnelle intégral.
- chapitre III: la commande par retour d'état .il traite le calcul du vecteur d'état complet pour prendre connaissance du comportement du système.
- chapitre IV : les réseaux de neurones .ce chapitre traite la résolution des problèmes avec une grande précision, de modélisation ou d'identification linéaire et non linéaire. . On dispose d'un système dont on connaît les entrées et les sorties quantitatives. A partir d'un certain nombre d'exemples fournis aux réseaux ceux-ci apprennent à se comporter comme le système.
- chapitre V : commande par modèle inverse neuronal. ce chapitre traite le modèle inverse qui est utiliser directement comme contrôleur il commande le processus en boucle fermée.
- Enfin nous terminons notre travail par une conclusion générale.

Chapitre

I

Modélisation Du Four Ventilé

I-A- INTRODUCTION

Le four électrique à résistances est sans doute le plus connu des appareils électrothermiques. Son utilisation industrielle remonte en effet au début des années **1920** et sa technique n'a cessé de se perfectionner depuis. Comme il est illustré à la **figure.I.1**, la chaleur à l'intérieur du four électrique ventilé, à modéliser, est produite par une résistance chauffante, commandé en tension Vc par un amplificateur de puissance.

La mesure de la température se fait à partir d'un thermocouple placé dans une cavité de mesure et d'un amplificateur d'instrumentation produisant une tension Vm, image de la température θm. L'ensemble capteur+amplificateur d'instrumentation sont supposés linéaires dans la gamme de température du four.

Les différents paramètres constituant ce processus sont **[11]:**

- $Q = k_1 V_c$: quantité de chaleur produite,
- R_a : résistance thermique freinant la circulation de la chaleur du conduit jusqu'à l'enceinte du four,
- C_a : capacité calorifique de l'enceinte du four,
- R_m : résistance thermique freinant la circulation de la chaleur du four à l'intérieur de la cavité de mesure,
- C_m : capacité calorifique de la cavité de mesure,
- R_f : résistance de fuite freinant la circulation de la chaleur vers l'extérieur du four,
- calorifique extérieur du four est considéré comme infini,
- θ_a, θ_m, θ_e : température respectives de l'enceinte du four, de la cavité de mesure, et e l'extérieur du four,
- θ_m : température mesurée son image est la tension V_m ($V_m=k_2\ \theta_m$).

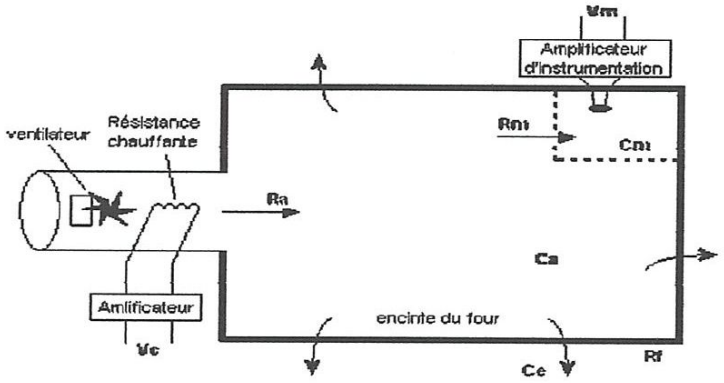


Fig.I. 1. Processus d'un four ventilé [1]

I-B- MODELISATION DU FOUR

La figure **(I.2)** montre le schéma électrique d'un four ventilé. Ce dernier est résistif- capacitif.

Afin de donner un aspect plus explicite à notre étude nous avons trouvé nécessaire de donner le modèle mathématique du four ventile

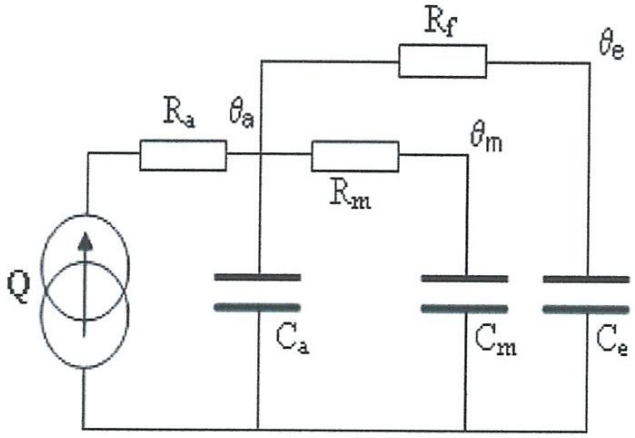


Fig.I. 2. Schéma électrique du four [1]

Par identification entre les paramètres du four et le schéma électrique et en considérant la quantité de chaleur Q comme étant le courant, on obtient les équations suivantes :

$$Q = C_a \frac{d\theta_a}{dt} + C\frac{d\theta_m}{dt} + \frac{\theta_a - \theta_e}{R_f} \qquad \textbf{(I.1)}$$

$C(d\theta)/dt$ Étant le courant qui traverse la capacité la température θ_m est exprimée par

(2) comme suite :

$$\theta_m = \theta_a - R_m C_m \frac{d\theta_m}{dt} \quad \textbf{(I.2)}$$

Les paramètres θ_a ,θ_m et θ_e ont été choisis comme étant respectivement les tensions (images des températures) des capacités C_a, C_m et C_e.

En utilisant la transformé de LAPLACE des expressions suivantes (1) et (2), nous obtenons les équations suivantes :

$$\theta_a = \left(Q + \frac{\theta_e}{R_f}\right).\left(\frac{R_f(1+R_m C_m p)}{1+(R_m C_m + R_f C_m + R_f C_a)p + R_f R_m C_m C_a p^2}\right) \quad \textbf{(I.3)}$$

$$\frac{\theta_m}{\theta_a} = \frac{1}{1+R_m C_m p} \quad \textbf{(I.4)}$$

Compte tenu des équations (3) et (4), le schéma fonctionnel du processus complet peut être comme suite :

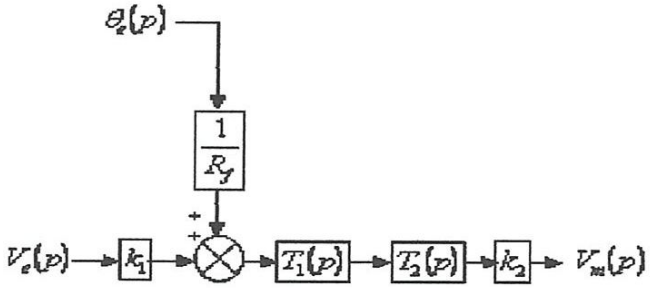


Fig. (I. 3.) Schéma fonctionnel du four

Avec $$T_1(p) = \frac{R_f(1+R_m C_m p)}{1+(R_m C_m + R_f C_m + R_f C_a)p + R_f R_m C_m C_a p^2} \quad \textbf{(I.5)}$$

Et $$T_2(p) = \frac{1}{1+R_m C_m p} \quad \textbf{(I.6)}$$

Les paramètres du modèle du four **[1]** sont dressés sur le tableau 1. La simulation a été réalisée sous l'environnement MATALAB 7.5/simulink6

R_a	0.01 °C / W	C_a	5000 J / °C
R_m	3 °C / W	C_m	10 J / °C
R_f	0.1 °C / W	C_e	∞
Qmax	5000 W	k_1	100 W / V
θe	20 °C	k_2	0.1 V / °C

Soit après calculs :

$$T_1(p) = \frac{0.1 + 3p}{1 + 531p + 1500p^2}$$

$$T_2(p) = \frac{1}{1 + 30p}$$

I-C- MODELE ANALOGIQUE DU FOUR :

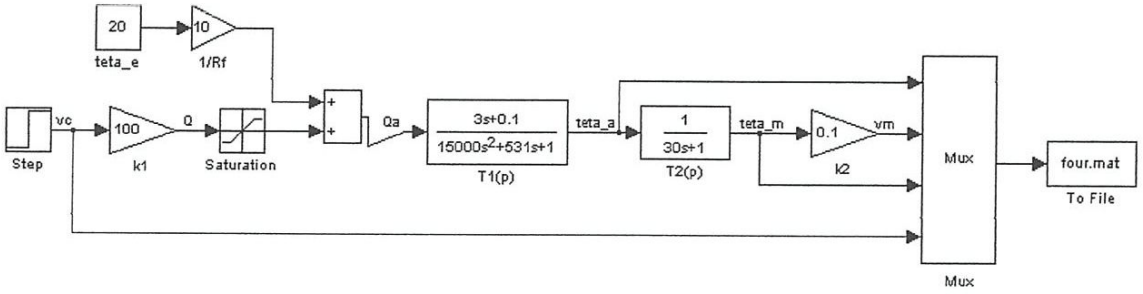


Fig. (I.4) Commande d'un four modèle analogique

I-D- RESULTAT DE SIMULATION :

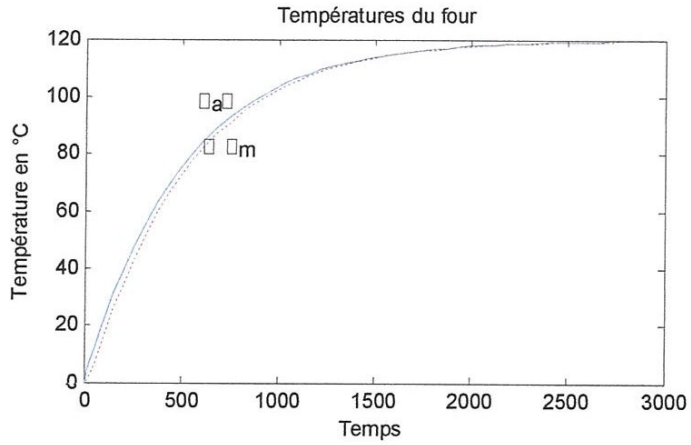


Fig. (I. 5.) Réponse du four à un échelon de tension.

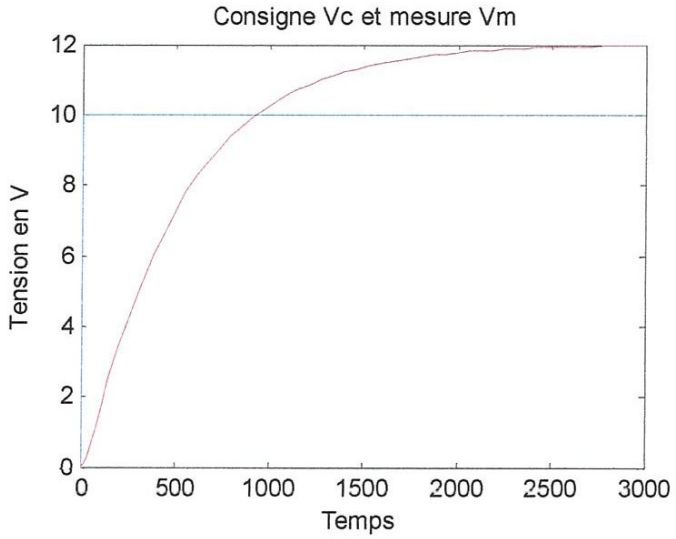


Fig. (I. 6.) Réponse du four à un échelon de température.

I-E- INTERPRETATION DE RESULTAT :

Les figures **(I.4)** et **(I.5)** montrent les réponses Vm et θm aux échelons de la tension de référence Vc et de la température du four θa respectivement.

La commande analogique montre bien que la tension mesurée Vm est l'image de la température du four θm Figures **(I. 4)** et **(I. 5).**

On remarque aussi que la réponse du four est assez lente **(figure I.4)** et que la réponse du système s'amortit vers une valeur autre que l'excitation figure **(I.5).**

I-F- CONCLUSION :

Le four se comporte comme un processus de premier ordre de constante de temps proche de 500 secondes. La température mesurée θm est très proche de la température de four θa lors du régime transitoire. Elles sont, bien sur confondues en régime permanant.

Chapitre

II

Régulation Des Fours Electriques A Résistances

II-A – GENERALITES

Le schéma fonctionnel d'un système asservi peut se représenter globalement par :

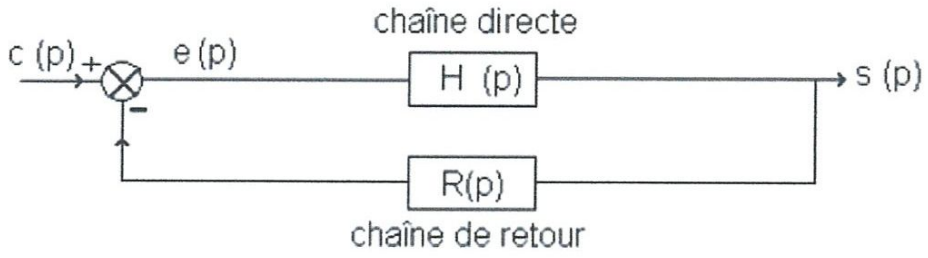


fig.II.1.schéma fonctionnel

La fonction de transfert en boucle ouverte est T (p) = H (p) R (p)

La fonction de transfert en boucle fermée est alors :

$$T'(p) = \frac{T(p)}{1 + H(p)R(p)}$$

Pour faire apparaître plus complètement les propriétés du système, on le représente par

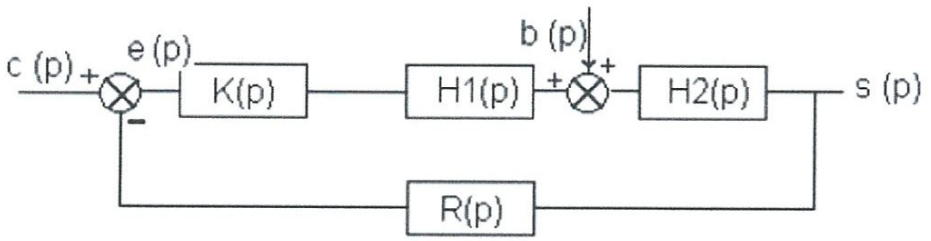


Fig.II.2. amélioration la réponse du système

H (p) = H1 (p) H2 (p) est la fonction de transfert du système lui-même et *b (p)* représente les perturbations que l'on peut introduire au système. Au contraire de *c(p)* qui représente la consigne à atteindre et que l'on connaît et maîtrise, *b(p)* est subie et pas toujours bien connue et on cherche à rendre son influence négligeable sur *s(p)*.

K (p) est la fonction de transfert du correcteur que l'on ajoute sur la chaîne directe afin d'améliorer la réponse du système.

II-B - STABILITE D'UN SYSTEME ASSERVI

Un système est stable au sens mathématique si, pour une entrée bornée, la grandeur de sortie reste bornée pour tout $t > 0$.

II-B-1-Critères de stabilité :

Un système bouclé est stable si et seulement si tous les pôles de sa FT en boucle fermée ont une partie réelle négative.

Il existe des critères algébriques qui permettent de déterminer le signe de la partie réelle des pôles à partir de la FT en boucle ouverte (critère de Routh –Hurwitz).

Nous nous contenterons de critères graphiques. On peut les faire en représentation de Bode (deux courbes $G(\omega)=20\log|T(\omega)|$, $\phi(\omega)$), de Nyquist (représentation polaire de $T(\omega)$) ou de Black Nichols ($G(\omega)=f(\phi(\omega))$ en coordonnées cartésiennes).

La représentation de Black s'avère commode en particulier pour voir les marges de stabilité et l'effet des correcteurs. Dans cette représentation, le point $\{\phi=-180°\ ;\ G=0\ dB\}$ représente le point critique

II-B-2-Marges de stabilité

Il ne suffit pas que le système soit stable au sens mathématique, il faut encore qu'il soit suffisamment stable (par exemple que son comportement transitoire ne comporte pas trop d'oscillations). Pour cela, on se fixe une marge de stabilité :

- ✓ Marge de phase $m\phi = 180° - \phi$ (pour ϕ correspondant à $G = 0\ dB$)
- ✓ Marge de gain ΔG par rapport à 0 pour $\phi = -180°$

Ces marges apparaissent simplement en représentation de Black **(fig.II.3).**

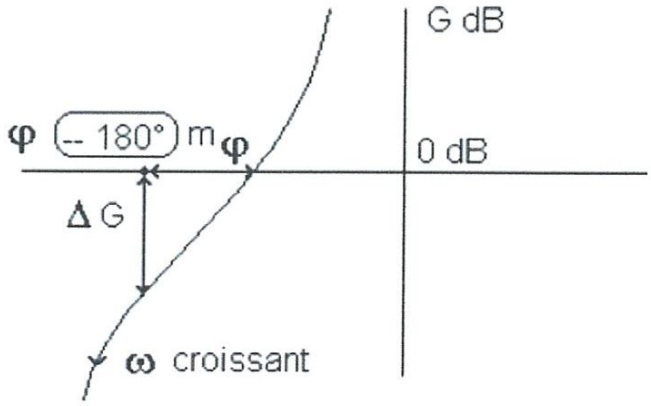


Fig.II.3. représentation de Black

On voit en particulier qu'une augmentation de gain (qui améliore la précision) – décalage de la courbe vers le haut – se fait au détriment de la marge de phase donc une détérioration de la marge de stabilité (et même une instabilité).

II-B-3-Précision d'un système asservi :

C'est l'écart entre la valeur de la consigne et la valeur de sortie. C'est donc la valeur à la sortie du soustracteur. On la détermine en régime permanent *($t \rightarrow \infty$ donc $p \rightarrow 0$).*

On peut montrer que l'écart est :

$$E(t \rightarrow \infty) = \lim_{p \rightarrow 0} \frac{p\, C(p)}{1 + T(p)}$$

On s'intéresse principalement à la réponse à un échelon. Alors, si E0 est la valeur de la consigne appliquée en t = 0, l'erreur est :

Nulle pour tout système de type 1 (ou plus) lorsqu'il y a au moins un intégrateur.

Égale à *E0 / (1+T0)* pour un système de type 0. On augmente la précision en augmentant le gain de la chaîne (ce qui détériore la stabilité).

C'est une règle générale des systèmes bouclés : il faut faire un compromis entre la précision et la stabilité.

Un résumé du comportement temporel **[12]** à une entrée échelon ou à une perturbation échelon est donné dans le tableau ci-dessous selon l'existence et la place d'un intégrateur dans le système.

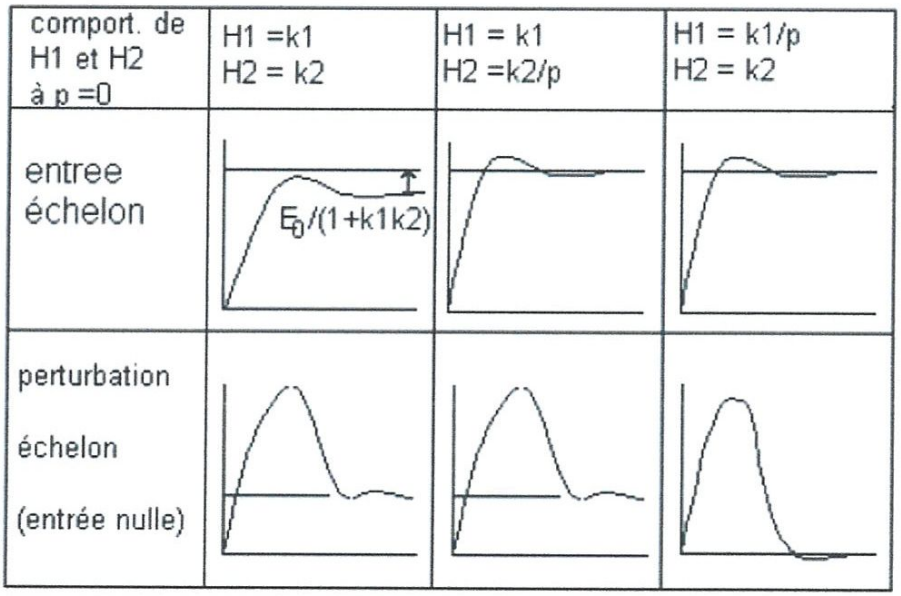


Fig.II.4comportement temporel

II-C - EFFET DES CORRECTEURS

Dans la fonction de transfert du système (en boucle ouverte), on peut distinguer trois domaines **[13]** de fréquence :

- ✓ En basse fréquence (et éventuellement à la fréquence nulle), un gain important va diminuer l'erreur dite statique (entre la consigne et la sortie en régime permanent). S'il y a un intégrateur – type 1– (ou plus), cette erreur sera nulle.
- ✓ Dans la zone de passage à *0 dB* (bande passante), il faut imposer une certaine marge de phase afin d'éviter des oscillations peu amorties et donc des dépassements importants de consigne. En général, une marge de phase de 45° limite le dépassement à 20%.
- ✓ En haute fréquence, il faut limiter le gain pour limiter l'influence des bruits de mesure (défauts des capteurs) qui se superposent au signal de retour injecté sur le soustracteur.

Le rôle des correcteurs sera alors de modifier la fonction de transfert afin de respecter au mieux ces différentes contraintes.

On se restreint ici aux correcteurs série

II-C-1 - CORRECTEUR PROPORTIONNEL (P) :

Le correcteur proportionnel P (ou à action proportionnelle P) est le correcteur le plus simple il agit directement sur l'erreur à l'instant courant pour générer l'action requise pour corriger le comportement dynamique de la grandeur à commander sa fonction de transfert est donne par la relation suivante :

$$K(p) = K$$

Avec K est le gain de l'action Proportionnelle

Sur le plan temporel cela correspond à *(Vs = K Ve)*.

L'avantage principal de ce type de correcteur est sa simplicité d'implantation à l'aide d'amplificateur opérationnel .son in convient majeur est qui il n'offre aucune possibilité d'annuler l'erreur du système en régime permanent d'un système de type zéro(ne possédant aucun pole à l'origine en boucle ouverte) lorsque la grandeur d'entre est en forme d'échelon.

II-C-2 - CORRECTEUR INTEGRAL (I):

Pour pallier l'inconvénient mentionné dans le cas du système de type zéros le correcteur du type intégral est d'une grande importance, car il permet d'augmenter le type du système d'une unité et, par conséquent, d'améliorer le type du système par unité et par conséquence d'améliorer le régime permanent .l'inconvénient majeur de ce type de correcteur se traduit par l'ajout d'un pole à l'origine du plan p, ce qui cause un effet de dégradation de stabilité intégral ayant un effet déstabilisateur, son utilisation est combiné en général à l'action proportionnel sa fonction de transfert est donne par la relation suivantes :

$$K(p) = 1/Tip$$

Avec *Ti* est la constante de temps de l'action d'intégral.

Sur le plan temporel cela correspond à $Vs = Td\, dVe/dt$

II-C-3 - CORRECTEUR DERIVE D :

Le correcteur intégral produit toujours le signal de commande même si le signal a atteint le régime permanant ce qui l'in convient majeur de ce correcteur une possibiliste pour éviter ceci consiste à concevoir un correcteur qui réagit à la dérive du signal d'erreur. Le principal avantage du correcteur dérivé est qui' il prédit l'évolution future de l'erreur et agit en conséquence pour contrer cette évolution .ce type de correcteur n'as aucun effet en régime permanent sa fonction de transfert est donne par la relation suivantes :

$$K(p) = Td\, p$$

Td est la constante de temps de l'action Dérivé

II-C-4 - CORRECTEUR PROPORTIONNEL – DERIVE (PD) :

$$K(p) = K(1 + Td\, p)$$

K est le gain de l'action Proportionnelle

Td est la constante de temps de l'action Dérivée

Sur le plan temporel cela correspond à $(Vs = K\, Ve + K/Ti \int Ve\, dt)$

L'action dérivée

- ✓ n'augmente pas la précision (pas d'effet à basse fréquence)
- ✓ augmente la stabilité (par apport de phase, la marge de phase est plus grande)
- ✓ améliore la rapidité de réponse (par augmentation de la bande passante)
- ✓ augmente la sensibilité au bruit.

II-C-5 - CORRECTEUR PROPORTIONNEL – INTEGRAL (PI) :

$$K(p) = K(1 + 1/Tip)$$

K est le gain de l'action Proportionnelle

Ti est la constante de temps de l'action d'intégral

Sur le plan temporel cela correspond à $Vs = K\, Ve + K\, Td\, dVe/dt$

L'action Intégrale

- ✓ améliore la précision (augmentation du gain en BF)
- ✓ diminue la stabilité (par perte de phase) et ralentit le système

II-C-6- CORRECTEUR PID :

Une combinaison judicieuse de ces effets permet de résoudre partiellement le dilemme Précision - Stabilité

On voit sur la fig.4 le diagramme de Black d'un système du 2ème ordre : le gain est faible à basse fréquence (ici 0 dB) donc peu de précision statique.

L'effet du correcteur PD est d'éloigner le diagramme du point critique : amélioration de la stabilité.

L'effet du correcteur PI est d'augmenter le gain à basse fréquence (amélioration de la précision) mais instabilité car le point critique est du mauvais côté.

L'effet du PID permet de combiner les effets positifs.

Loi de commande	Avantage	Inconvénient
Tout ou rien	▪ simplicité ▪ montée rapide	▪ dépassement de la consigne ▪ oscillations permanente
Proportionnelle P	▪ dépassement de la consigne ▪ Simplicité ▪ Meilleure précision	▪ Ecart de statisme la mesure ne joint pas la consigne ▪ Risque d'instabilité
Proportionnelle intégrale PI	▪ l'action intégrale permet de rattraper la consigne	▪ Instabilités si intégrale mal réglée ▪ Erreur statique nulle
Proportionnelle intégrale dérive PID	▪ Efficacité La dérive s'oppose aux fluctuations rapides de la mesure ▪ Très utilisé en industrie Action PI + PD	▪ Réglage délicat ▪ Réglage des paramètres plus difficile

Le tableau 1 : résume la principale loi de commandes possible

II-C-7- METHODE POUR LE CHOIX DES CORRECTEURS :

- ✓ Analyse du système (identification, performances dynamiques, réponse fréquentielle).
- ✓ Analyse du cahier de charges (traduction en termes d'erreur, de rapidité, de marge de phase, de pulsation ωco).
- ✓ Choix de la structure du correcteur compte tenu du cahier des charges et des caractéristiques du système.
- ✓ Calcul des paramètres du correcteur.

- ✓ Vérification des performances du système corrigé. Si le cahier des charges n'est pas satisfait, retour à 3.
- ✓ Réalisation de l'asservissement et tests.

II-C-8-PRINCIPE GENERAL :

La régulation d'un four à résistances a pour rôle de contrôler **[8]** la température de l'ambiances ou de la charge éventuellement de la faire évoluer suivant un cycle prédétermine (programmation)de s'assurer que les résistances ne dépassent pas une températures maximale limite

Sur le plan de la régulation on distingue deux catégories de fours :

- ✓ Les fours à faibles puissances spécifiques inférieures à environ15kw/m^2 de paroi chauffantes qui nécessitent une régulation classique
- ✓ Les fours à puissances élever et faible inertie pour il faut passer à une solution plus élabore (P I D numérique)

II-C-9- REGULATEUR INTEGRAL AVEC COMPENSATION DES POLES ET ZEROS :

II-C-9-1-Schéma global de la simulation :

On réalise une commande intégrale avec compensation des pôles et zéros du modèle.

Le modèle est défini par le sous-système suivant, dans laquelle on décrit la fonction de transfert analogique T (p)

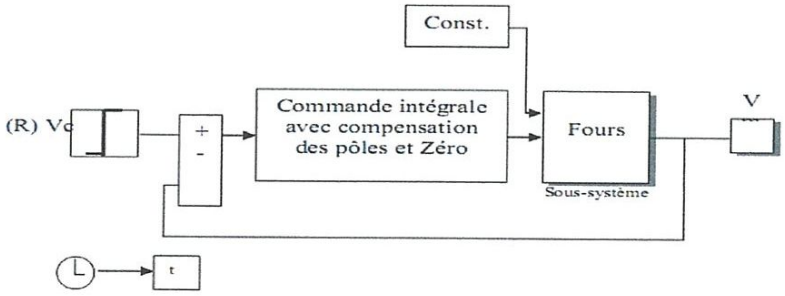


Fig.II :5 Schéma global de simulation de la commande intégral avec compensation des Pôles et Zéro

II-C-9-2-Résulta de simulation :

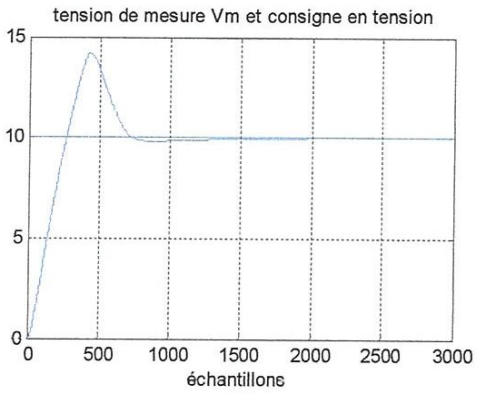


Fig. II.6. Réponse du four avec un temps de réponse

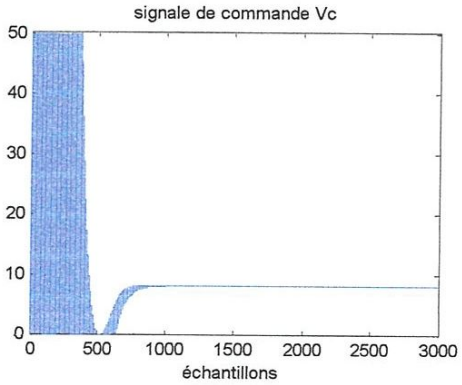


Fig. II.7. Signal de commande

II-C-9-3-Interprétation du résultat :

Le correcteur synthétisé par la commande intégrale avec compensation des pôles et des zéros permet à la fois d'annuler l'erreur statique et d'obtenir un processus corrigé stable et présentant une dynamique satisfaisante. On peut vérifier la réponse de l'asservissement en poursuite à travers la **figure. II.6**.

On voit bien que le temps de réponse a été amélioré mais on remarque un dépassement assez important dans le régime transitoire.

II-C-9-4-Conclusion:

En régime transitoire entre 0 et 50 volts la commande oscille du fait de sa limitation et se stabilise en régime permanant on peut diminuer l'amplitude du dépassement avec la conséquence d'une augmentation du temps de réponses si on diminue le gain de l'intégrateur

Chapitre

III

La Commande Par Retour D'état

III-A- Introduction

La commande par retour d'état est la commande des systèmes modélisés par leur représentation d'état. L'idée consiste toujours à piloter le système par un signal de consigne et à générer automatiquement le signal de commande en confrontant en permanence **[14]** la valeur de la consigne et le comportement réel du système.

L'écart entre consigne et comportement réel sert de base au signal de commande du système.

Dans la commande par retour d'état, nous n'allons pas mesurer le signal de sortie pour le boucler sur l'entrée, mais nous allons nous servir du vecteur d'état complet pour prendre connaissance du comportement du système.

III-B- Représentation d'état discret du four

On modifie le schéma fonctionnel du four comme suit afin de donner une représentation d'état de la fonction $T(p) = V_m(p)/V_a(p)$, ou $V_a(p)$ est l'image de la température dans l'enceinte du four.

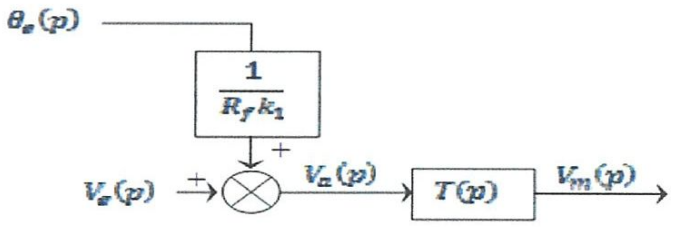


Fig. III.1. Représentation d'état discret du four

Avec $$T(p) = \frac{k1\ k2\ R_f}{1+(R_m C_m+R_f R_m+R_f C_a)p+R_f R_m C_m C_a\ p^2} \quad \textbf{(III.1)}$$

On peut écrire T(p) de la forme :

$$T(p) = \frac{\frac{k1\ k2\ R_f}{R_f R_m C_m C_a\ p^2}}{\frac{1}{R_f R_m C_m C_a}+\frac{(R_m C_m+R_f R_m+R_f C_a)}{R_f R_m C_m C_a}p+p^2} \quad \textbf{(III.2)}$$

On pose $$b_0 = \frac{k1\ k2\ R_f}{R_f R_m C_m C_a}$$

$$a_0 = \frac{1}{R_f R_m C_m C_a}$$

Et $$a_1 = \frac{R_m C_m+R_f R_m+R_f C_a}{R_f R_m C_m C_a}$$

La fonction $T(p)$ devient de la forme $T(p) = \frac{b_0}{p^2+a_1 p+a_0}$

Elle est d'ordre 2, sa représentation d'état nécessite donc 2 variables d'état :

Notons $Y(p) = Vm(p)$ sortie du processus

$U(p) = Vc(p)$: Entrée du processus

$$b_0 U(p) = (p^2 + a_1 p + a_0) Y(p) \quad \textbf{(III.4)}$$

Ceci conduit à l'équation différentielle suivante :

$$\frac{d^2 y(t)}{dt^2} + a_1 \frac{dy(t)}{dt} + a_0 y(t) = b_0 u(t) \quad \textbf{(III.5)}$$

Les variable d'état représentant les sorties des intégrateurs

$$x_1(t) = y(t) \quad \text{Et} \quad x_2(t) = \dot{x_1}(t)$$

$$\dot{x_2}(t) = -a_1 x_2(t) - a_0 x_1(t) + b_0 u(t)$$

Soit sous forme matricielle :

$$\begin{bmatrix} \dot{x_1}(t) \\ \dot{x_2}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ b_0 \end{bmatrix} u(t) \quad \textbf{(III.6)}$$

$$y(t) = v_m(t) = [1 \quad 0] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

Avec :

$v_m(t) = y(t)$

$v_c(t) = u(t)$

Finalement on obtient comme représentation d'état correspondant à la fonction de transfert T(p)

$$\begin{bmatrix} \dot{x_1}(t) \\ \dot{x_2}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ b_0 \end{bmatrix} v_c(t) \quad \textbf{(III.7)}$$

$$y(t) = v_m(t) = [1 \quad 0] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \quad \textbf{(III.8)}$$

III-C- Commande d'un four placements des pôles :

III-C-1- DISCRETISATION DU PROCESSUS (FOUR):

Vu que la constante du temps du processus est grande environ de 500 s celui-ci est discrétisé avec une période D'échantillonnage de 0.8 seconde .on envisage d'asservir la température du four par le biais de La mesure V_m ; pour cela on utilise la méthode du placement des pôles en ce fixant une Dynamique du second ordre avec :

1. une pulsation propre W_n=0.005rd/s
2. un facteur d'amortissement ξ =0.007

Soit le système à temps continu {A, B, C, D}

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases} \quad \textbf{(III.9)}$$

Calcul des matrices A_d, B_d, C_d, D_d du système discrétise **[15][16][17]**à partir des matrices A, B, C et D du système continu

Le système est discrétise au pas d'échantillonnage Te, les paramètres du processus aux instant d'échantillonnage kTe se formalisent ainsi

$$\begin{cases} x(k+1) = A_d x(k) + B_d u(k) \\ y(k) = C_d x(k) + D_d u(k) \end{cases} \quad \textbf{(III.10)}$$

Si l'entrée u de ce système est un signal échantillonné bloqué (avec B.O.Z)

On a

$$U(t) = u_d(k) \text{ , avec } t \in [kT, (k+1)T]$$

L'intégration de l'équation d'état entre les instant kT et $(k+1)T$

$$x_d(k+1) = e^{AT} x_d(k) + \int_{kT}^{(k+1)T} e^{A(t-kT)} Bu(t)dt \quad \textbf{(III.11)}$$

$$= e^{AT} x_d(k) + \int_{kT}^{(k+1)T} e^{A(t-kT)} dt B u_d(k)$$

Où

$x_d(k) = x(kT)$

On pose $y_d(k) = y(kT)$ Et

$$A_d = e^{AT} \quad \text{Et} \quad B_d = \int e^{AT} dt B \quad \textbf{(III.12)}$$

$$C_d = C \quad \text{Et} \quad D_d = D$$

Connaissant les matrices A, B, C, D on peut déterminer les matrices, A_d , B_d, C_d , D_d

$A_d = e^{AT} = L^{-1}[(pI - A)^{-1}]$ On a déterminé $(pI - A)$ **(III.13)**

Avec I : matrice d'identité tel que $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ par conséquence

$$\mathrm{pI} - \mathrm{A} = \begin{bmatrix} \mathrm{p} & 0 \\ 0 & \mathrm{p} \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix} = \begin{bmatrix} \mathrm{p} & -1 \\ -a_0 & \mathrm{p} + a_1 \end{bmatrix} \quad \textbf{(III.14)}$$

Donc

$$(pI - A)^{-1} = \frac{1}{p(p+a_1)+a_0} \begin{bmatrix} p + a_1 & 1 \\ -a_0 & p \end{bmatrix} \quad \textbf{(III.15)}$$

Donc on cherche $L^{-1}[(pI - A)^{-1}]$:

$$L^{-1}[(pI - A)^{-1}] = L^{-1}\left[\frac{1}{p(p+a_1)+a_0} \begin{bmatrix} p + a_1 & 1 \\ -a_0 & p \end{bmatrix}\right] \quad \textbf{(III.16)}$$

On développe $$L^{-1}[(pI - A)^{-1}] = \begin{bmatrix} L^{-1}\left(\frac{p+a_1}{p(p+a_1)+a_0}\right) & L^{-1}\left(\frac{1}{p(p+a_1)+a_0}\right) \\ L^{-1}\left(\frac{-a_0}{p(p+a_1)+a_0}\right) & L^{-1}\left(\frac{p}{p(p+a_1)+a_0}\right) \end{bmatrix} \quad \textbf{(III.17)}$$

Application :

$$b_0 = \frac{k1\,k2\,R_f}{R_f R_m\, C_m C_a} = \frac{100\times0.1\times0.1}{0.1\times3\times10\times5000} = 6.66 \times 10^{-5}$$

$$a_0 = \frac{1}{R_f R_m\, C_m C_a} = \frac{1}{0.1\times3\times10\times5000} = 6.66 \times 10^{-5} \quad \textbf{(III.18)}$$

$$a_1 = \frac{R_m\, C_m + R_f R_m + R_f C_a}{R_f R_m\, C_m C_a} = \frac{3\times10+0.1\times3+0.1\times5000}{0.1\times3\times10\times5000} = 0.03536$$

On remarque que a_0 est très petit la formule **(III.18)** devient :

$$L^{-1}[(pI - A)^{-1}] = \begin{bmatrix} L^{-1}\left(\frac{p + a_1}{p(p + a_1)}\right) & L^{-1}\left(\frac{1}{p(p + a_1)}\right) \\ L^{-1}\left(\frac{-a_0}{p(p + a_1)}\right) & L^{-1}\left(\frac{p}{p(p + a_1)}\right) \end{bmatrix}$$

$$L^{-1}[(pI - A)^{-1}] = \begin{bmatrix} L^{-1}\left(\frac{1}{p}\right) & L^{-1}\left(\frac{1}{p(p + a_1)}\right) \\ L^{-1}\left(\frac{-a_0}{p(p + a_1)}\right) & L^{-1}\left(\frac{1}{(p + a_1)}\right) \end{bmatrix}$$

$$L^{-1}[(pI - A)^{-1}] = \begin{bmatrix} L^{-1}\left(\frac{p + a_1}{p(p + a_1) + a_0}\right) & L^{-1}\left(\frac{1}{p(p + a_1) + a_0}\right) \\ L^{-1}\left(\frac{-a_0}{p(p + a_1) + a_0}\right) & L^{-1}\left(\frac{p}{p(p + a_1) + a_0}\right) \end{bmatrix}$$

$$L^{-1}[(pI - A)^{-1}] = \begin{bmatrix} 1 & \frac{1}{a_1}(1 - e^{-a_1 t}) \\ \frac{-a_0}{a_1}(1 - e^{-a_1 t}) & e^{-a_1 t} \end{bmatrix}$$

$$L^{-1}[(pI-A)^{-1}] = \begin{bmatrix} 1 & \frac{1}{0.03536}(1-e^{-0.03536\times 0.8}) \\ \frac{-6.66\times 10^{-5}}{0.03536}(1-e^{-0.03536\times 0.8}) & e^{-0.03536\times 0.8} \end{bmatrix}$$

$$L^{-1}[(pI-A)^{-1}] = \begin{bmatrix} 1.0000 & 0.7894 \\ -0.0001 & 0.9737 \end{bmatrix}$$

$$A_d = \begin{bmatrix} 1.000 & 28.2805(1-e^{-0.03536t}) \\ -18.8348\times 10^{-5}(1-e^{-0.03536t}) & e^{-0.03536t} \end{bmatrix} \quad \textbf{(III.20)}$$

$B_d = \int e^{AT} dt B$ Avec

$$B_d = \int \begin{bmatrix} 1.000 & 28.2805(1-e^{-0.03536t}) \\ -18.8348\times 10^{-5}(1-e^{-0.03536t}) & e^{-0.03536t} \end{bmatrix} \begin{bmatrix} 0 \\ 6.66\times 10^{-5} \end{bmatrix} dt \quad \textbf{(III.21)}$$

$$B_d = \begin{bmatrix} \int_0^{Te} 188.348\times 10^{-5}(1-e^{-0.03536t})dt \\ \int_0^{Te} 6.66\times 10^{-5} dt \end{bmatrix} = \begin{bmatrix} 188.34\times 10^{-5}\int_0^{Te}(1-e^{-0.03536t})dt \\ 6.66\times 10^{-5}\int_0^{Te} e^{-0.03536t}\,dt \end{bmatrix}$$

$$B_d = \begin{bmatrix} 188.34\times 10^{-5}\int_0^{Te}(1-e^{-0.03536t})dt \\ 6.66\times 10^{-5}\int_0^{Te} e^{-0.03536t}\,dt \end{bmatrix} = \begin{bmatrix} 0.214\times 10^{-4} \\ 0.5263\times 10^{-4} \end{bmatrix}$$

Et finalement: $A_d = \begin{bmatrix} 1.000 & 0.7894 \\ -0.0001 & 0.9737 \end{bmatrix}$; $B_d = \begin{bmatrix} 0.214\times 10^{-4} \\ 0.5263\times 10^{-4} \end{bmatrix}$

$C_d = C = [1 \quad 0]$ Et $D_d = D = 0$ **(III.23)**

Afin de procéder ou calcul du vecteur d'état il faux vérifier la commandabilité du processus.

Soit Com la matrice de commandabilité

$$Com = [Bd \;\; AdBd] \quad \textbf{(III.24)}$$

$$com = \begin{bmatrix} 0.214\times 10^{-4} & 0.3829\times 10^{-4} \\ 0.5263\times 10^{-4} & 0.5124\times 10^{-4} \end{bmatrix}$$

Calcul du déterminant de la matrice Com :

Det (com) = (0.214X10⁻⁴X0.5124X10⁻⁴)-(0.5263X10⁻⁴X0.3829X10⁻⁴)

Finalement le Det*(Com)* $\neq 0$ donc le processus est commandable

On peut déterminer le vecteur d'état K

On remarque que le rang de la matrice de $Com = 2$

Donc la dynamique du second ordre recherchée pour le processus asservi décrite par le polynôme $P(z)$ s'obtient à l'aide des équations suivantes :

$$P(z)=z^2+p_1z+p_2 \qquad \textbf{(III.25)}$$

On détermine p_1 et p_2 comme suite :

$$p_1 = -2e^{-\xi w_n}\cos\left(w_n Te\sqrt{1-\xi^2}\right)$$

Donc

$$p_1 = -2e^{-0.707\times0.005\times0.8}\cos\left(0.005\times0.8\sqrt{1-0.707^2}\right)=-1.9941$$

Et

$$p_2 = e^{-2\times\xi w_n Te} = e^{-2\times0.707\times0.005\times0.8}=0.9943$$

D'autre part on na : l'équation $P(z)$ peut s'écrire d'une autre façon :

$$Det\,[zI-(A_d-B_dK)]=P(z) \qquad \textbf{(III.26)}$$

Par identification de la relation **(III.26)** = **(III.25)**

$$P(z)=z^2+p_1z+p_2$$

Et de relation $Det\,[zI-(A_d-B_dK)]=P(z)$

Avec $K=[k_0\ k_1]$

On trouve $K=[-0.6210 \quad -393.0670]$

III-C-2- Schéma global de la simulation :

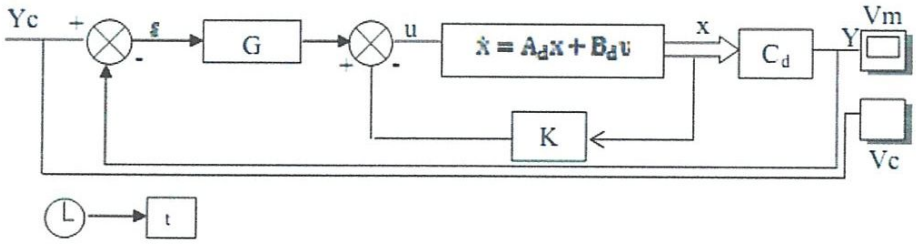


Fig. III.2. Schéma global de la simulation commande d'un Four

Placement des pôles

III-C-3- RESULTAT DE SIMULATION :

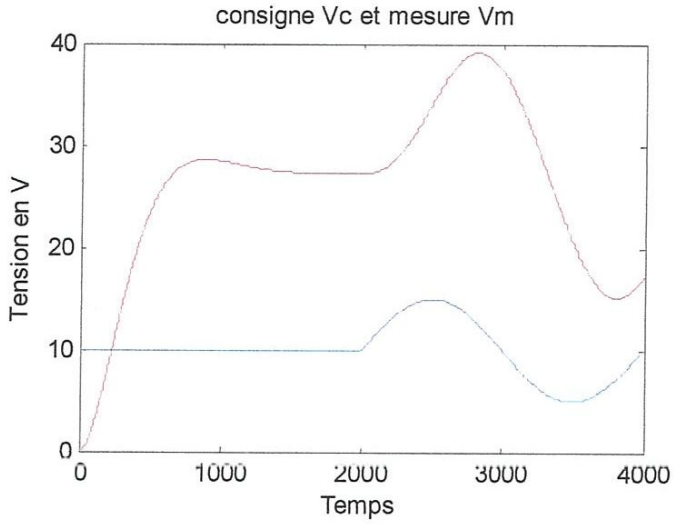


Fig. III.3.

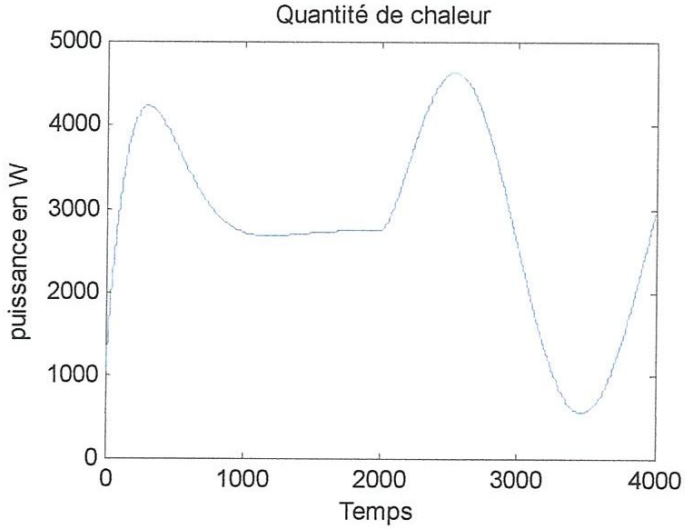


Fig. III.4.Résultats de simulation

III-C-4- INTERPRETATION DES RESULTATS :

On remarque que les signaux sont obtenus pour une consigne échelon de 100°C appliquée pendant 2000s, à laquelle s'ajoute une consigne sinusoïdale de 50°c.

Pour **la Fig. III.3** : on constate que l'erreur statique est importante

Pour la **Fig. III.4** : la commande se sature à 500w lorsqu'il y a des variations brusques de la consigne.

III-C-5- CONCLUSION :

Du faite que l'erreur statique est importante il faut remédier ce problème pour que le processus four comporte une intégration donc on s'intéresse à la commande par retour d'état avec intégration.

III-D- Commande d'un four par retour d'état avec intégration :

III-D-1- SCHEMA GLOBAL DE LA SIMULATION :

Le schéma fonctionnel du processus devient :

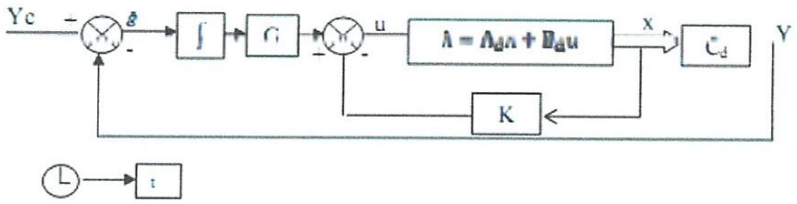


Schéma global de la simulation commande par retour d'état avec intégration Fig. III.5

Avec :

$$\begin{cases} u(k) = -kx(k) + g\varepsilon(k) \\ \varepsilon(k) = \frac{1}{1-z^{-1}}(y_c(k) - y(k)) \end{cases} \quad \textbf{(III.27)}$$

Soit $$\varepsilon(k+1) = \varepsilon(k) + y_c(k+1) - y(k+1) \quad \textbf{(III.28)}$$

On obtient après calcul :

$$\begin{cases} x(k+1) = (A_d - B_d k)x(k) + B_d g\varepsilon(k) \\ \varepsilon(k+1) = (C_d B_d - C_d A_d)x(k) + (1 - C_d B_d g)\varepsilon(k) + y_c(k+1) \end{cases} \quad \textbf{(III.29)}$$

Ce système peut être décrit par une met horde matriciel

$$\begin{bmatrix} x(k+1) \\ \varepsilon(k+1) \end{bmatrix} = \begin{bmatrix} A_d - B_d k & B_d g \\ C_d B_d k - C_d A_d & 1 - C_d B_d g \end{bmatrix} = \begin{bmatrix} x(k) \\ \varepsilon(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} y_c(k+1) \quad \textbf{(III.30)}$$

$$y(k) = (C_d \ \ 0) \begin{bmatrix} x(k) \\ \varepsilon(k) \end{bmatrix} \quad \textbf{(III.31)}$$

On considère la réponse du système à un échelon unitaire, alors :

$$\begin{cases} y_c(t) = 1 \\ y(\infty) = y_c(\infty) = 1 \end{cases} \quad \textbf{(III.32)}$$

Si on exprime le vecteur des erreurs suivantes

$$\begin{bmatrix} x_{err} \\ \varepsilon_{err} \end{bmatrix} = \begin{bmatrix} x(k) & x(\infty) \\ \varepsilon(k) & \varepsilon(\infty) \end{bmatrix} \quad \textbf{(III.33)}$$

On peut écrire

$$\begin{bmatrix} x_{err}(k+1) \\ \varepsilon_{err}(k+1) \end{bmatrix} = \begin{bmatrix} A_d - B_d k & B_d g \\ C_d B_d k - C_d A_d & 1 - C_d B_d g \end{bmatrix} \begin{bmatrix} x_{err}(k) \\ \varepsilon_{err}(k) \end{bmatrix} \quad \textbf{(III.34)}$$

On peut décomposer cette expression de façon à faire apparaître le vecteur des coefficients de retour d'état avec intégration :

$$\begin{bmatrix} x_{err}(k+1) \\ \varepsilon_{err}(k+1) \end{bmatrix} = \begin{bmatrix} A_d & 0 \\ -C_d A_d & 1 \end{bmatrix} \begin{bmatrix} x_{err}(k) \\ \varepsilon_{err}(k) \end{bmatrix} + \begin{bmatrix} -B_d k & B_d g \\ C_d B_d k & -C_d B_d g \end{bmatrix} \begin{bmatrix} x_{err}(k) \\ \varepsilon_{err}(k) \end{bmatrix} \quad \textbf{(III.35)}$$

Soit :

$$\begin{bmatrix} x_{err}(k+1) \\ \varepsilon_{err}(k+1) \end{bmatrix} = \begin{bmatrix} A_d & 0 \\ -C_d A_d & 1 \end{bmatrix} \begin{bmatrix} x_{err}(k) \\ \varepsilon_{err}(k) \end{bmatrix} + \begin{bmatrix} B_d \\ -C_d B_d \end{bmatrix} [-k \;\; g] \begin{bmatrix} x_{err}(k) \\ \varepsilon_{err}(k) \end{bmatrix} \quad \textbf{(III.36)}$$

On obtient alors les matrices d'état du système asservi :

$$A_1 = \begin{bmatrix} A_d & 0 \\ -C_d A_d & 1 \end{bmatrix} \qquad B_1 = \begin{bmatrix} B_d \\ -C_d B_d \end{bmatrix} \qquad C_1 = [1 \;\; 0 \;\; 0] \quad \textbf{(III.37)}$$

Et le vecteur de retour avec intégration :

$$K_I = [-K \;\; g]$$

Cette formulation sera utilisée pour obtenir le vecteurK_1. Vecteur de retour avec intégration

On prend pour le processus asservi une dynamique du second ordre :

1. une pulsation propre $W_n = 0.005 rd/s$
2. un facteur d'amortissement $\xi = 0.9$

III-D-2- RESULTAT DE SIMULATION :

Le logiciel utilisé est le **MATLAB 7**. Le schéma de principe du four a été réalisé à partir de **SIMULINK6**

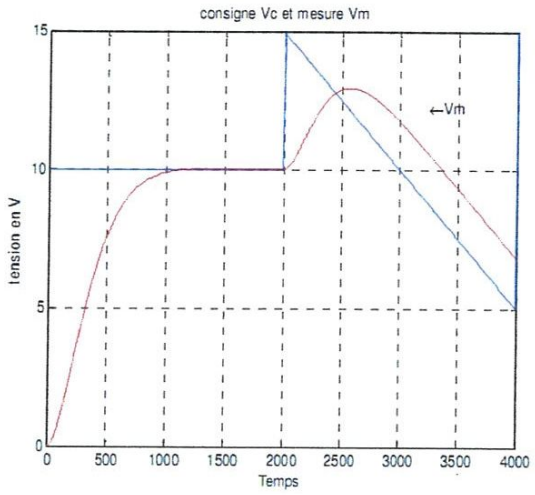


Fig.(III.6)

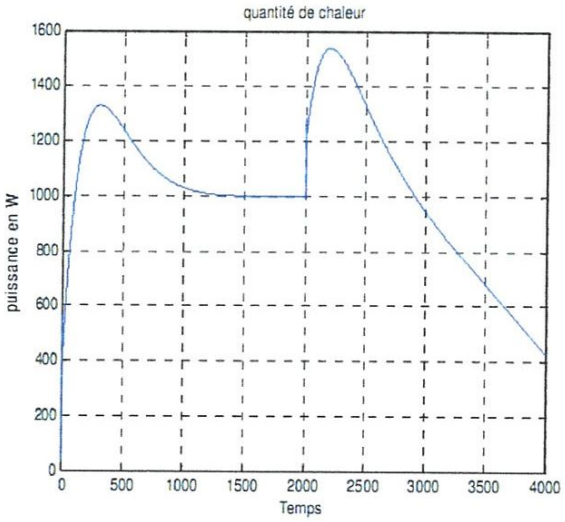


Fig.(III.7). Résultat de simulation

III-D-3- INTERPRETATION DES RESULTATS :

- ✓ On remarque qui il n'as Pas de dépassement.
- ✓ Pas d'erreur statique.
- ✓ On remarque que la consigne rampe déduit une erreur de traînage.

III-D-4- CONCLUSION :

On constate que lorsque on augmente le facteur d'amortissement c'est pour éviter un dépassement de la consigne Pour annuler l'erreur de traînage il faut ajouter une seconde d'intégration dans la chaîne direct.

III-E- COMMANDE D'UN FOUR ELECTRIQUE VENTILE PAR RETOUR D'ETAT EN UTILISANT LE RECONSTRUCTEUR DE KALMAN :

III-E-1- INTRODUCTION :

Dans le présent travail, le modèle du processus n'est pas connus avec précision et que la mesure de la température m ϑ est entache d'un bruit blanc gaussien et centré. La commande par retour d'état avec intégration à été réalisé par **[11].**

Notre but consiste à déterminer les estimateurs de variable lorsque l'environnement présente des perturbations aléatoires.

Les applications du filtre de kalman **[18] [19] [20] [21]** sont nombreuses dans les métiers de l'ingénieur; celui du génie des procédés n'échappe pas à cette règle. La commande par retour d'état avec intégration en utilisant le reconstructeur de kalman est la méthode recherchée dans ce présent travail.

Une loi de commande par retour d'état est synthétisée pour le réglage de la température d'un four électrique .cette commande est couplée à un observateur de processus stochastique dont le principe est de corriger la trajectoire du modèle en combinant les observations avec l'information, fournie par le modèle de façon à minimiser l'erreur entre la sortie mesuré et la sortie estimé.

III-E-2- PRINCIPE D'UTILISATION DU FILTRE DE KALMAN :

La modélisation d'état prenant en compte deux variables aléatoires [22], le bruit de processus et le bruit de mesure, on parle d'un modèle d'état stochastique.

Le modèle d'état discret est décrit à l'aide de deux équations :

- L'équation de processus :

$$x(k+1) = Ax(k) + Bu(k) + w(k) \qquad \textbf{(III.38)}$$

Ou *w(k)* est le bruit de modélisation lié à l'incertitude que l'on a sur le modèle de processus .c'est sans doute le paramètre le plus difficile à quantifier .on le considère blanc, gaussien et centré.

$$\begin{cases} E[w(k)] = 0 \\ E[w(k)w^T(k+n)] = Q \end{cases} \qquad \textbf{(III.39)}$$

Q Étant la matrice de variance de w(k). Si le système est stationnaire alors :

$$Q = \delta^2{}_w = cte$$

- l'équation de mesure :

$$Y(k) = Cx(k) + v(k) \qquad \textbf{(III.40)}$$

Ou *v(t)* est le bruit de mesure également considéré comme blanc, gaussien et centré.

$$\begin{cases} E[v(k)] = 0 \\ E[v(k)v^T(k+n)] = R \end{cases} \quad \textbf{(III.41)}$$

R Étant la matrice de variance de v(k). Si le système est stationnaire.

$$R = \sigma^2 v = cte$$

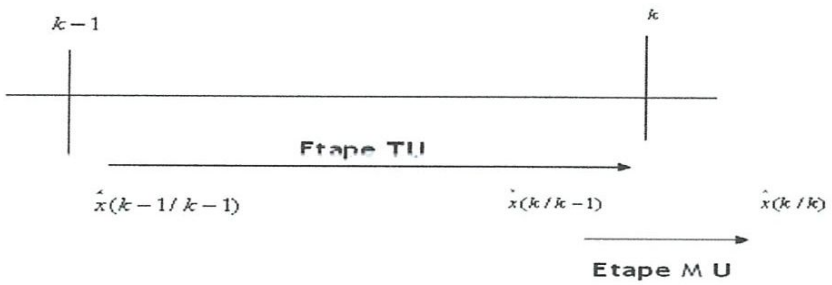


FIg. III.8. Estimation de l'état d'un processus

Une étape d'évolution de l'état du processus entre 2 instants d'échantionnage. Cette étape est appelée étape TU POUR << Time Update>>

L'état du processus évolue selon l'équation suivante :

$$\hat{x}(k/k-1) = A\hat{x}(k-1) + Bu(k-1) \quad \textbf{(III.43)}$$

Pendant cette phase située entre l'instant (k-1)Te et l'instant kTe on assiste à une à une augmentation de variance de l'erreur d'estimation p(k/k-1)

$$p(k/k-1) = E([x(k) - \hat{x}(k/k-1)][x(k) - \hat{x}(k/k-1)]^T) \quad \textbf{(III.44)}$$

$$p(k/k-1) = AP(k-1/k-1)A^T + Q \quad \textbf{(III.45)}$$

• Une étape de mise à jour de l'état du processus à l'instant t suite à l'acquisition de y(k) .Cette étape est appelée étape M U pour<< Measurement Update>>.

Cette étape est décrite par les équations suivantes :

$$\hat{x}(k/k) = \hat{x}(k/k-1) + L(k)[y(k) - C\hat{x}(k/k-1)] \quad \textbf{(III.46)}$$

Avec L(k) matrice de gain d'adaptation du filtre.

Pendant cette étape on constate une diminution de la variance de l'erreur du fait de l'erreur du fait de l'information apportée par la mesure de y(k)

$$p(k/k) = p(k/k-1) + P(k/k-1)C^T[R + CP(k/k-1)C^T]^{-1}CP(k/k-1) \quad \textbf{(III.47)}$$

Le gain est mis à jour à l'aide de la formule suivante :

$$L(k) = P(k/k-1)C^T[R + CP(k/k-1)C^T]^{-1} \quad \textbf{(III.48)}$$

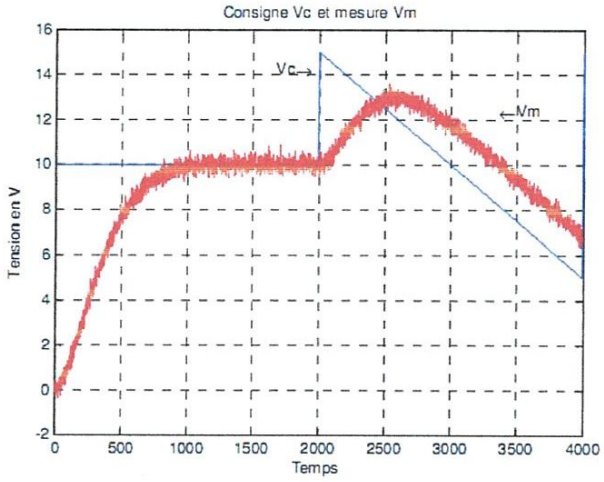
La mesure de la température ϑ_m est entachés d'un bruit blanc gaussien et centré dans ce cas on peut reconstruire l'état du processus à l'aide d'un reconstructeur stochastique de kalman **[23]** .la fonction kalman permet d'obtenir le vecteur gain de l'observateur a partir de la description du processus de la variance du bruit du processus Q et de la variance du bruit de mesure R.

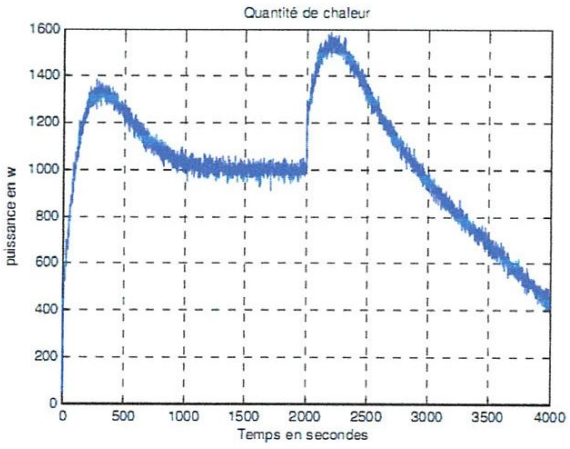
On prend pour notre application :

$$\begin{cases} E[vv^T] = R = 0.1 \\ E[ww^T] = Q = 0.1 \end{cases} \quad \textbf{(III.49)}$$

IV-E-3- RESULTAT DE LA SIMULATION :

A partir de la description du processus discrétiser précédemment obtenus de la **Fig. IV.5** pour laquelle on a ajouté le reconstructeur de kalman. On constate que le vecteur d'état ainsi reconstruit est utiliser pour produire le retour d'état, au travers du vecteur K est que La sortie estimée par le filtre de kalman est proche de celle du processus non bruité (absence des bruits de mesure et de modélisation).





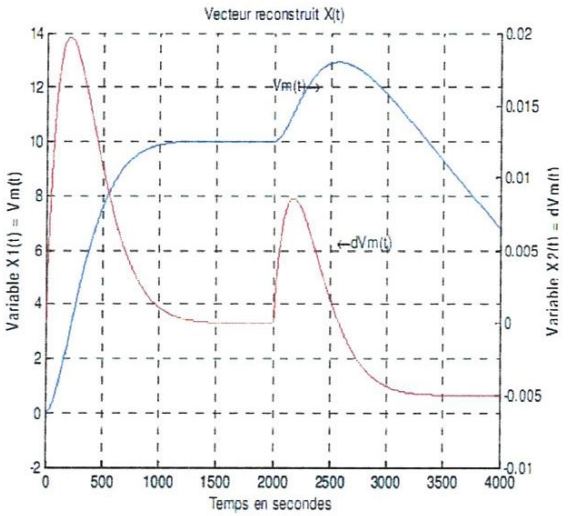


Fig. III.9.commande d'un four avec reconstructeur de kalman

III-E-3- CONCLUSION :

Le but du reconstructeur stochastique de kalman ajouter a la commande d'un four par retour d'état avec intégration consiste à déterminer des estimateurs de variables du système lorsque L'environnement présent des perturbations aléatoires. A Partir de ce modèle, un régulateur de la température par retour d'état avec intégration est élabore puis un couplage de cette approche avec un observateur de processus stochastique .les grandeur estimer sont ensuite utiliser pour élaborer la loi de commande l'efficacité du reconstructeur stochastique apparaît nettement à la comparaison des variable estimées, nettoyées des bruits de modélisation et de mesure, au signal mesure *Vm* fortement bruité. L'estimation de l'état est également satisfaite, ce procéder évite l'emploi des capteurs lorsque ceux-ci sont coûteux ou lorsque la mesure risque d'être difficile ou fortement bruitée.

Chapitre IV

Réseaux de Neurones

IV -A- INTRODUCTION

L'utilisation d'un régulateur analogique ainsi que la commande intégrale avec compensation des pôles et des zéros ayants une grande non linéarité ou une variation paramétrique pour la commande du four conduit à une dégradation des performances de contrôle. Une telle commande exige une modélisation la plus précise possible du four à commander. Ceci amène les chercheurs à introduire de nouvelle techniques de réglage afin d'obtenir les performances désires.

Les réseaux de neurones sont des systèmes de traitement de l'information dont la structure s'inspire de celle du système nerveux. Ils sont utilisés dans de nombreuses applications telles que l'approximation de fonctions, la reconnaissance de la forme, la modélisation de processus et la commande.

Les contrôleurs à base de réseaux de neurones artificiels sont très utilisés dans le contrôle de processus complexe, non linéaires ou mal identifies. Le grand intérêt porté aux réseaux de neurones provient de leurs capacités d'apprendre. Qui donne la possibilité d'approximer n'importe quelle fonction avec la précision souhaité cette capacité d'apprentissage va permettre aussi de prendre en compte les nouvelle contrainte qui agissent sur un système.

IV -B- RESEAUX DE NEURONES

Un réseau de neurones est un assemblage de constituants élémentaires interconnectés (appelés «neurones» en hommage à leur modèle biologique), qui réalisent chacun un traitement simple mais dont l'ensemble en interaction fait émerger des propriétés globales complexes. Chaque neurone fonctionne indépendamment des autres de telle sorte que l'ensemble forme un système massivement parallèle. L'information est stockée de manière distribuée dans le réseau sous forme de coefficients synaptiques ou de fonctions d'activation, il n'y a donc pas de zone de mémoire et de zone de calcul, l'une et l'autre sont intimement **liés [24][25][26][27][28].**

Un réseau ne se programme pas, il est entraîné grâce à un mécanisme d'apprentissage. Les tâches particulièrement adaptées au traitement par réseau de neurones sont: l'association, la classification, la discrimination, la prévision ou l'estimation, et la commande de processus complexes.

Les réseaux de neurones artificiels consistent en des modèles plus ou moins inspirés du fonctionnement cérébral de l'être humain en se basant principalement sur le concept de neurone.

IV -B-1- NEURONE BIOLOGIQUE :

Un neurone est une cellule particulière comme le montre la figure **(IV.B.1).**

Elle possède des extensions par lesquelles elle peut distribuer des signaux (axones) ou en recevoir (dendrites).

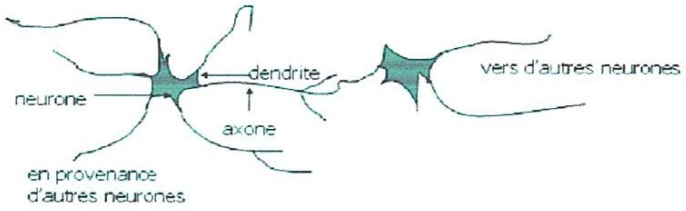


Fig IV.B.1 Schéma simplifié d'un neurone

IV -B-2- NEURONE FORMEL :

Des observations de neurone biologique, découle le modèle du neurone formel proposé par Mac Cul loch et Pitti en **1943** :

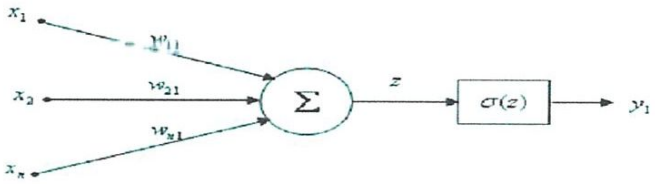


Fig. IV.B.2 Neurone formel

Les x_i représentent le vecteur d'entrée, elles proviennent soit des sorties d'autres neurones, soit de stimuli sensoriels (capteur visuel, sonore…)

Les w_{ij} sont les poids synaptiques du neurone j . Ils correspondent à l'efficacité synaptique dans les neurones biologiques ($w_{ij}>0$: synapse excitatrice;

$w_{ij}<0$: synapse inhibitrice). Ces poids pondèrent les entrées et peuvent être modifiés par apprentissage.

Le potentiel somatique z(t) est calculé par la somme de toutes les entrées pondérées:

$$z(t) = \sum_{i=1}^{n} w_{ij}\, x_i \qquad \textbf{(IV.B.1)}$$

On applique une fonction d'activation sur ce potentiel somatique pour obtenir la valeur de sortie du neurone j (la fonction d'activation calcule la valeur de l'état du neurone; c'est cette valeur qui sera transmise aux neurones avals:

$$y = \sigma(z(t)) \qquad \textbf{(IV.B.2)}$$

La fonction d'activation σ peut prendre les formes les plus courantes qui sont présentées à la figure **(IV.B.3):**

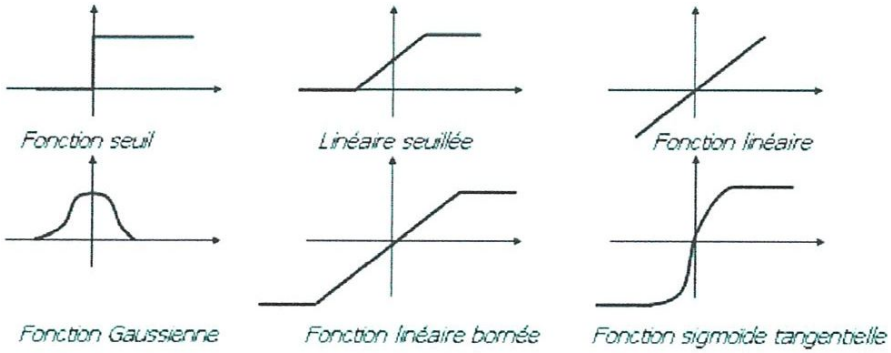


Fig. IV.B.3 Fonctions d'activations

La fonction d'activation des neurones de la couche d'entres est une fonction linéaire, pour les neurones des couches caches et les neurones de la couches de sortie la fonction d'activation est une fonction non linéaire (généralement une sigmoïde), elle permet d'introduire au réseau l'aspect non linéaire, elle doit être continue, différentiel et bornée

La figure **Fig. IV.B .4** montre la structure d'un neurone artificiel. Chaque neurone artificiel est un processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones amont. A chacune de ces entrées est associé un poids w abréviation de weight (poids en anglais) représentatif de la force de la connexion. Chaque processeur élémentaire est doté d'une chaque connexion est associé un poids.

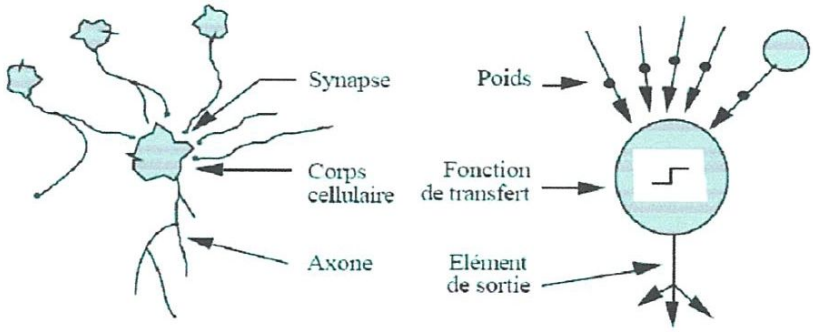


Fig. IV.B.4 mise en correspondance neurone biologique /neurones artificiel

IV -C- ARCHITECTURE DES RESEAUX DE NEURONES

IV -C-1- LES RESEAUX NON BOUCLES :

Ce sont des réseaux unidirectionnels sans retour arrière (feedforward). Le signal de sortie est directement obtenu après l'application du signal d'entrée. Si tous les neurones ne sont pas des organes de sortie on parle de neurones cachées, **(Fig. IV.B.5).**ce sont des systèmes statiques, utiliser principalement pour la modélisation statique de processus la fonction réaliser par un réseau de neurones non bouclé est une fonction algébrique.

La fonction non linéaire réaliser par un réseau, dépend des valeurs des poids

Le temps n'intervient pas comme variable fonctionnelle, le réseau n'a pas de mémoire et ses sorties ne dépendent pas de son passé.

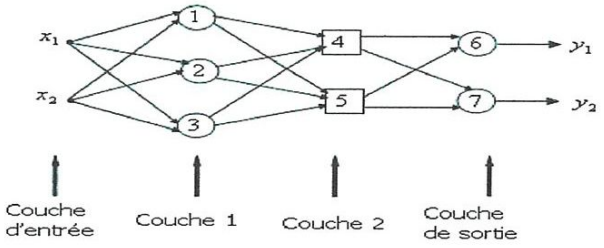


Fig. IV.B.5. Réseau non bouclé

IV -C-2- LES RESEAUX BOUCLES :

Il s'agit de réseaux de neurones avec retour en arrière (feedback network or récurrent network) **(Fig. IV.B.6)**.dont le graphe des connexions contient des cycles; ce sont des systèmes dynamiques, utilisés comme filtre non linéaire, ainsi pour la modélisation et la commandes de processus ; la fonction réaliser par un réseau non bouclé est un ensemble d'équation différentielle.

La fonction non linéaire réaliser par un réseau, dépend des valeurs des poids w_{ij} pour qu'un réseau effectue une tache donnée, il faut donc ajuster la valeur de ses poids .l'opération d'ajustement des poids est appelée apprentissage.

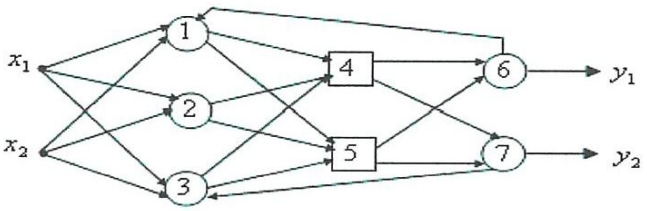


Fig. IV.B.6 Réseau bouclé

IV -D- APPRENTISSAGE

La mémoire permet de s'adapter à un environnement changeant et d'anticiper ces changements. Elle a deux fonctions distinctes: l'acquisition d'un élément d'information par un processus d'apprentissage, et le rappel de l'information stockée.

Les entrées du réseau de neurone perçoivent des informations sur l'environnement. La mémoire et son organisation sont intimement liées à la structure et aux valeurs des paramètres du réseau que l'apprentissage consiste à modifier. Les sorties du réseau permettent d'agir sur l'environnement ou sur le réseau lui-même par le biais de connexions dites récurrentes.

On peut distinguer trois types d'apprentissage:

- L'apprentissage supervisé.
- L'apprentissage non supervisé (auto organisation).
- L'apprentissage par renforcement.

Pour ces trois types d'apprentissage, il y a également un choix traditionnel entre :

- L'apprentissage << off- line>> : toutes les données sont dans une base d'exemples d'apprentissage qui sont traités simultanément.
- L'apprentissage<<on-line>> : les exemples sont présentés les un après les autres au fur et à mesure de leurs disponibilités.

Ainsi que plusieurs types de structures de réseaux:

- À connexions récurrentes.
- À couches.

Et plusieurs modes de fonctionnement:

- Statique (qui ne tient pas compte de ses états antérieurs).
- Dynamique (qui dépend de ses états antérieurs).
- Stochastique (dont l'activation est soumise à une loi de probabilité).

Il existe aussi différents types d'entrées sorties:

- Discret - relation d'ordre (nombres entiers par exemple).
- Continu - relation d'ordre (nombres réels).

L'apprentissage d'un réseau de neurones artificiels est induit par une procédure Itérative d'ajustement de ses paramètres internes (poids synaptiques et nombres de neurones). Cette procédure d'ajustement est décrite par un algorithme d'apprentissage Celui-ci détermine alors le comportement du réseau. Ainsi le

Comportement d'un même réseau diffère selon l'algorithme d'apprentissage utilisé.

Pour modifier ses paramètres. Et aussi l'apprentissage consiste à appliquer un ensemble d'exemples (entrées-sorties désirées) qui constituent la base d'apprentissage .et à partir de la différence entre les sortie du réseau et les sortie désirées, les poids d'interconnections seront ajusté suivant une règle d'apprentissage prédéfinie, de manière que les sortie du réseau soient aussi proches que possible des sortie désire pour commencer l'apprentissage, les poids d'interconnections doivent être initialisés arbitrairement. L'apprentissage sera considère comme accompli lorsque le réseau atteint un niveau de performances défini par l'utilisateur. Ce niveau signifie que le réseau a achevé une précision statistique désirée. Après l'apprentissage, le réseau peut être utilisé pour prédire le nouvel échantillon de sortie, basé seulement sur la connaissance de l'échantillon d'entrée.

Ainsi il existe type trois grandes classes d'apprentissage existent :

IV -D-1- L'APPRENTISSAGE NON- SUPERVISE :

Ce type d'apprentissage est choisi lorsqu'il n'y pas de connaissances à priori des sorties désirés pour des entrées données. En fait, c'est de l'apprentissage par exploration où l'algorithme d'apprentissage ajuste les poids des liens entre neurones de façon à maximiser la qualité de classification des entrées.

Pour les réseaux à apprentissage non supervisé (Hop Field, Kohonen, etc.), on présente une entrée au réseau et on le laisse évoluer librement jusqu'à ce qu'il se stabilise.

IV -D-2- L'APPRENTISSAGE PAR RENFORCEMENT :

Dans ce cas, bien que les sorties idéales ne soient pas connues directement, il y a un moyen quelconque de connaître si les sorties du RNA s'approchent ou s'éloignent du but visé. Ainsi, les poids sont ajustés de façons plus ou moins aléatoire et la modification est conservée si l'impact est positif ou rejetée.

IV -D-3- PRINCIPE DE L'APPRENTISSAGE SUPERVISE :

Cet algorithme d'apprentissage ne peut être utilisé que lorsque les combinaisons d'entrées-sorties désirées sont connues. L'apprentissage est alors facilité et par là, beaucoup plus rapide que pour les deux autres algorithmes puisque l'ajustement des poids est fait directement à partir de l'erreur, soit la différence entre la sortie obtenue par le RNA et la sortie désirée.

Une fois le type d'apprentissage choisi selon les connaissances à priori du problème à résoudre, un autre aspect important consiste à déterminer si le réseau de neurones, dans sa phase d'exploitation, sera statique ou dynamique. Dans le premier cas, un préapprentissage sera effectué alors que dans le second, un apprentissage continu sera requis pour permettre au RNA de réagir aux changements de son environnement. Algorithme d'apprentissage ne peut être utilisé que lorsque les combinaisons.

L'apprentissage supervisé consiste à modifier, durant une phase d'entraînement, les valeurs des différents paramètres (poids synaptiques, seuil d'activation...) du neurone (ou du réseau). Ce sont ces valeurs qui définissent le comportement du neurone (ou du réseau), i.e. qui définissent quelle est la valeur de sortie du neurone (ou du réseau) en fonction des valeurs d'entrée lors de la phase d'exploitation. **[29][30]**

Lors de la phase d'entraînement on extrait un sous-ensemble de vecteurs d'entrée de la base d'apprentissage appelé vecteurs d'apprentissage, et un sous-ensemble complémentaire de vecteurs de test. On présente alors au neurone (ou réseau) un vecteur d'apprentissage, on calcule sa sortie et on modifie ses poids synaptiques en suivant une règle d'apprentissage. Lorsqu'on a présenté tous les vecteurs d'apprentissage, on présente les vecteurs de test qui n'ont pas servi à l'apprentissage, et on calcule le taux d'erreurs que fait le réseau, c'est-à-dire sa capacité à généraliser. Tant qu'on est en dessous d'un certain seuil, on recommence la phase d'entraînement, avec un nouveau sous-ensemble d'apprentissage et un nouveau sous-ensemble de test. Si les résultats sont satisfaisants, on passe en phase d'exploitation.

La phase d'exploitation est la mise en application directe du réseau. On ne modifie plus ses poids.

Mais comment doit-on modifier ces paramètres pour que le neurone se comporte comme on veut? C'est-à-dire pour qu'il associe à un vecteur d'entrée, la valeur de sortie désirée.

IV -D-4- Préapprentissage :

Sous cette condition, la phase d'apprentissage et d'exploitation du réseau de neurones est bien distincte. Une fois l'apprentissage terminé, la structure et les poids entre les neurones sont fixés. Lorsque la tâche à effectuer par le RNA ne change pas significativement avec le temps, un RNA statique peut être très efficace et beaucoup plus simple d'implémentation. Par ailleurs, il n'est alors pas aussi important de se soucier de la simplicité de la fonction d'apprentissage et de la rapidité de convergence du RNA.

IV -D-5- APPRENTISSAGE CONTINU :

Un apprentissage continu est requis lorsque les combinaisons d'entrées-sorties idéales risquent de changer avec le temps ou selon diverses conditions externes. Le gain de flexibilité inhérent à la capacité d'adaptation du RNA s'obtient au détriment d'une implémentation plus complexe du système puisque l'algorithme d'apprentissage doit alors être intégré à la structure de calcul du RNA.

Ainsi, pour un système en temps réel, la rapidité d'adaptation ou de convergence de la fonction d'apprentissage devient alors un facteur très important. Par ailleurs, la complexité d'implémentation de la fonction d'apprentissage devient également un élément à considérer dans le choix de l'algorithme d'apprentissage, parfois au détriment de la performance.

L'ajustement du poids des liens entre les neurones peut s'effectuer selon diverses équations mathématiques, dont la plus populaire est sans aucun doute la loi de Hebb. Les autres équations sont souvent des dérivées de cette dernière. Par ailleurs, le choix de l'équation d'adaptation des poids dépend en grande partie de la topologie du réseau de neurones utilisé. Notons qu'il est aussi possible de faire évoluer l'architecture du réseau, soit le nombre de neurones et les interconnexions, mais avec d'autres types d'algorithmes d'apprentissage.

IV -D-6- LA REGLE DE HEBB :

Cette règle très simple émet l'hypothèse **[31]** que lorsqu'un neurone « A » est excité par un neurone « B » de façon répétitive ou persistante, l'efficacité (ou le poids) de l'axone reliant ces deux neurones devrait alors être augmentée.

Lorsque deux neurones proches sont actifs simultanément, le poids synaptique les reliant est augmenté de manière à faciliter une telle co-activité ultérieure. Dans les autres cas, le poids reste inchangé.

Ce qui se traduit par:

$$w_{ij}(t+1) = w_{ij}(t) + \propto (t).x_i(t).y_j(t) \qquad \textbf{(IV.B.3)}$$

Où $\alpha(t)$ est le coefficient d'apprentissage, w_{ij} le poids synaptique reliant l'entrée x_i au neurone j, et y_j la sortie du neurone j.

Le problème de cette règle est qu'on ne peut mémoriser qu'un nombre limité d'associations vecteur d'entrée valeur de sortie.

IV -D-7- LOI DE HOP FIELD :

Cette loi [32] se base sur la même hypothèse que la loi de Hebb. Mais ajoute une variable supplémentaire pour contrôler le taux de variation du poids entre les neurones avec une constante d'apprentissage qui assure à la fois la vitesse de convergence et la stabilité du RNA.

IV -D-8- LOI DE DELTA :

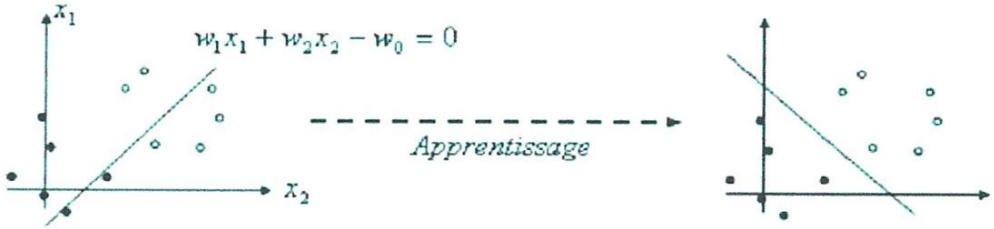
Cette loi **[33]** est aussi une version modifiée de la loi de Hebb. Les poids des liens entre les neurones sont continuellement modifiés de façon à réduire la différence (le delta) entre la sortie désirée et la valeur calculée de la sortie du neurone. Les poids sont modifiés de façon à minimiser l'erreur quadratique à la sortie du RNA. L'erreur est alors propagée des neurones de sortie vers les neurones des couches inférieures, dans la règle de delta appelé aussi règle de widrow-hoff, l'apprentissage est réalisé par itération. Les poids à la phase initiale sont générés aléatoirement. Ensuite la valeur de w_{ij} au temps (t+1) s'obtient par

$$w_{ij}(t+1) = w_{ij}(t) + \propto (t).\delta_j(t).x_i$$

Où $\delta_j(t)$ est l'erreur faite par le neurone.

IV -D-9- LE PERCEPTRON :

Dans le neurone du Perceptron **[34]** on utilise la fonction d'activation à seuil, qui permet de classer les vecteurs d'entrée dans un hyperplan. La règle d'adaptation permet de modifier la position de l'hyperplan séparateur dont l'équation dans l'espace d'entrée est définie par les poids du neurone, afin de réduire l'erreur (d(t)-y(t)). Ainsi, à chaque présentation d'un couple (x,y) mal classé (c'est-à-dire dont la sortie proposée y(t) par le neurone ne correspond pas à la sortie désirée d't), on modifie le vecteur poids de manière à ramener le point mal classé du bon côté de l'hyperplan **(Fig. IV.B.6).** Si le point est bien classé, on ne fait rien.



Droite séparatrice dont l'équation dans L'espace d'entrée est définie par les poids Synaptiques du perceptron

solution qui sépare correctement l'espace d'entrée en deux classes

Fig. IV.B.7 Principe d'apprentissage

Règle du Perceptron :

$$w_i(t+1) = w_i(t) + \propto (t).(d(t) - y(t)).x_i \qquad \textbf{(IV.B.4)}$$

Où

$$y(t) = sign(z(t)) \Leftrightarrow \begin{cases} p(t) > 0 \Rightarrow y(t) = 1 \\ p(t) < 0 \Rightarrow y(t) = -1 \end{cases}$$

Et

d(t) ∈ [-1,1]

Le Perceptron ne converge que si les classes sont linéairement séparables, on prend pour exemple la fonction logique OU- Exclusif qui ne peut pas être simulée par un Perceptron.

Le problème de Ou- Exclusif :

x1	x2	x1⊕x2	
0	0	0	○---► 0
0	1	1	●---► 1
1	0	1	
1	1	0	

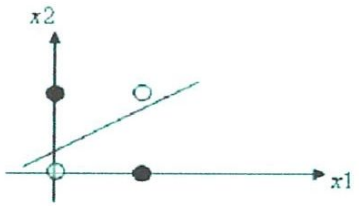


Fig. IV.B.8 Ou Exclusif

L'apprentissage ne converge pas, i.e l'erreur ne tend jamais vers zéro, les poids sont sans cesse modifiés car le Perceptron n'arrive jamais à classer correctement tous les vecteurs d'entrées **(Fig. IV.B.7).**

IV -D-10- L'ADALINE: ADAPTIVE LINE AR Elément :

Soit un neurone dont la fonction de sortie est l'Identité:

$$y(t) = \sigma.\big(z(t)\big) = z(t) \qquad \textbf{(IV.B.5)}$$

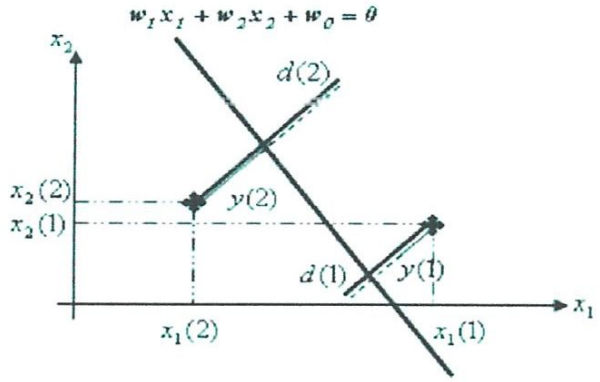
La sortie de neurone dépend linéairement des entrées d'où son nom .Il s'agit de résoudre un système d'équations linéaire (car **σ** est l'identité pour l'ADA LINE).

Exemple pour n vecteurs d'apprentissage à deux dimensions présentées lors de la phase d'entraînement:

$$\left.\begin{array}{l} \sigma.(x_1(1)w_1 + x_2(1)w_2 - w_0 = y(1)) \\ \sigma.(x_1(2)w_1 + x_2(2)w_2 - w_0 = y(2)) \\ . \\ . \\ \sigma.(x_1(n)w_1 + x_2(n)w_2 - w_0 = y(n)) \end{array}\right\} \qquad \textbf{(IV.B.6)}$$

Faire l'apprentissage du neurone consiste donc à trouver les poids w_i qui sont solutions de ce système .On constate que ce système n'a en générale pas de solution, i.e. si le nombre d'exemples présentés (ici n) est supérieur à la dimension de l'espace d'entrée plus un (pour les seuils) (ici 2+1 = 3) donc au nombre d'inconnues (w_0, w_1 et w_2).

Comme le système n'a souvent pas de solution, on va essayer de trouver une droite telle que la somme des écarts entre la distance d(k) que l'on désire associer au vecteur d'entrée x(k) et la distance réelle y(k) de ce vecteur à la droite, soit minimale **(Fig. IV.B.8).**



—— *Distance désirée*
—— *Distance réelle*
---- *Erreur $(d(k)-y(k))$ dont on veut que la somme pour tous les couples $(X(k),d(k))$ soit minimale*

Fig.IV.B.9 Présentation des écarts entre d(k) et y(k)

On va chercher par une méthode itérative (méthode du gradient stochastique) à minimiser un critère d'erreur de notre choix. Le critère des moindres carrés **[35]** est très souvent utilisé:

$$\varepsilon(\vec{w},k) = \frac{1}{2}(d(k) - y(k))^2 \qquad \textbf{(IV.B.7)}$$

Où d(k) est la sortie désirée et y(k) la sortie réellement fournie par le neurone pour chaque vecteur d'apprentissage, et w est le vecteur poids.

On note que lorsqu'une solution existe, elle annule cette fonction coût. On va donc calculer le gradient de la fonction d'erreur ε.

La règle d'apprentissage s'obtient alors par:

$$w(t+1) = w(t) - \alpha(t).\overrightarrow{grad}(\varepsilon(\vec{w})) \qquad \textbf{(IV B.8)}$$

Où la coordonnée i du vecteur grad est donnée par:

$$\frac{d}{dw_i}\varepsilon(\vec{w}) = -\big(d(t) - y(t)\big).\sigma'\big(z(t)\big).x_i(t) \qquad \textbf{(IV.B.9)}$$

Soit la règle du delta pour l'apprentissage de l'ADA LINE

$$w_i(t+1) = w_i(t) + \propto (t).(d(t) - y(t)).x_i(t) \qquad \textbf{(IV.B.10)}$$

Et la règle du delta généralisée pour l'apprentissage d'un ADALINE non linéaire:

$$w_i(t+1) = w_i(t) + \propto (t).\big(d(t) - y(t)\big).\sigma'.\big(z(t)\big).x_i(t) \qquad \textbf{(IV.B.11)}$$

On note que dans ce dernier cas, la fonction de sortie **σ** doit être dérivable (par exemple une sigmoïde (tanh), mais pas un échelon).

On peut constater que ces règles ressemblent à la règle de Hebb mais la sortie du neurone est remplacée par l'erreur de sortie (écart entre la sortie réelle et la sortie désirée).

L'ADALINE réalise un séparateur linéaire **(Fig.IV.B.9):**

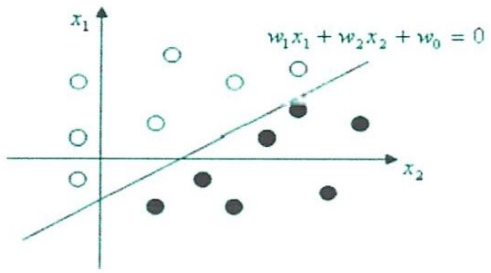


Fig IV.B.10 Réalisation d'un séparateur linéaire

L'ADALINE converge toujours vers la solution des moindres carrés, que les classes d'entrée soient ou non linéairement séparables, mais la solution obtenue n'est pas forcément celle qui sépare correctement les classes.

L'ADALINE non linéaire sépare toujours les classes linéairement séparables car les éléments éloignés de la zone de séparation ont peu d'influence (zone de saturation de la fonction d'activation).

L'information stockée par un neurone au niveau de ses poids est apprise des données. Les poids du réseau sont modifiés pour que la sortie qu'il propose soit le plus proche possible de la sortie désirée que les données imposent.

L'apprentissage est en fait une phase d'optimisation itérative d'une fonction de coût dont le minimum est la solution recherchée, i.e. la configuration des poids telle que la sortie obtenue soit le plus proche possible de la sortie désirée au sens des moindres carrés.

Pour minimiser une fonction continue, il existe d'autres techniques telles que la descente de gradient. Cette technique permet cependant d'illustrer de manière simple les principes de l'apprentissage valables pour un neurone seul, mais aussi pour un ensemble de neurones.

IV-E- LES RESEAUX MULTICOUCHES ET L'APPROXIMATION DE FONCTIONS

IV -E-1- LE PERCEPTRON MULTICOUCHE (MLP) :

IV -E-1-1-STRUCTURE DU RESEAU :

Une seule couche de neurones ne pouvant réaliser que des séparations linéaires, l'idée vient alors de rajouter des couches dites cachées pour réaliser un réseau de neurone multicouche. Dans une couche, les neurones ne sont pas connectés entre eux.

Le MLP (Multi Layered Perceptron) est un exemple d'un tel réseau. On peut alors voir la sortie d'un réseau comme une somme pondérée des sorties des neurones de la couche précédente. Si au lieu de neurones linéaires, on utilise des neurones non linéaires (fonction de sortie de type sigmoïde), la sortie d'un réseau est une somme pondérée de sigmoïdes, et donc si l'on place les sigmoïdes des neurones des couches internes judicieusement (en réglant leurs poids), on peut théoriquement approcher n'importe quelle fonction, pourvu que l'on ait une couche cachée et suffisamment de neurones dans cette couche.

Le problème se pose alors de régler correctement les poids d'un tel réseau, donc de réaliser son apprentissage.

IV-E-1-2- L'ALGORITHME DE LA RETRO PROPAGATION DU GRADIENT D'ERREUR (BACKPROPAGTION)

Pour réaliser l'apprentissage d'un réseau multicouche, on utilise la règle d'apprentissage du delta généralisé pour chaque neurone j **[36]:**

$$w_{ij}(t+1) = w_{ij}(t) + \propto (t).\delta_j(t).x_i \qquad \textbf{(IV.B.12)}$$

Où $\delta_j(t)$ est l'erreur faite par le neurone j.

Exemple avec un réseau à deux entrées, trois neurones dans une couche cachée, et deux neurones dans la couche de sortie **(Fig. IV.B.10)** :

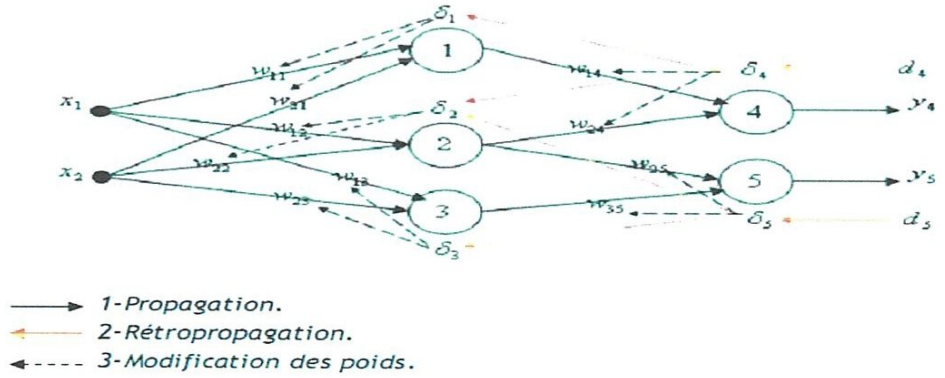


Fig. IV.B.11 Algorithme de Rétro propagation

Pour pouvoir modifier les poids synaptiques reliant la couche d'entrée à la couche cachée (w_{11} ; w_{12} ; w_{12} et w_{21} ; w_{22} ; w_{23}) il faut connaître les sorties désirées d1, d2 et d3 qui

permettent d'appliquer la règle du delta généralisé, i.e. connaître les erreurs: $\leftarrow_1, \leftarrow_2, \leftarrow_3$ que font les neurones 1, 2 et 3.

L'idée consiste alors à propager les erreurs δ_4 et $\delta_5$vers les neurones 1, 2 et 3, au travers des poids w14, w24, w25 et w35, d'où le nom de rétro propagation du gradient d'erreur de l'algorithme proposé indépendamment par Rumelhart, Le Cun et Hilton en **1984.**

Algorithme de rétro propagation du gradient d'erreur

1 - Calculer par propagation les valeurs de sortie de tous les neurones de la couche d'entrée vers la couche de sortie:

$$y_j(t) = \sigma\left(\sum_{i=0}^{n} w_{ij}(t)\, y_i(t)\right)$$

Où x_0=-1 et wj0 est le seuil du neurone j, et y_i correspond à l'entrée x_i du réseau connectée au neurone j ou bien à la sortie du neurone i de la couche précédente.

2 - Calculer par rétro propagation l'erreur des neurones de la couche de sortie vers la couche d'entrée :

$$\delta_j(t) = \left(d_j(t) - y_j(t)\right).\ \nabla'\left(\sum_{i=0}^{n} w_{ij}(t)\, y_i(t)\right)$$

Pour tous les neurones j de la couche de sortie reliés à n neurones i de la couche cachée précédente, et

$$\delta_j(t) = \sum_{k=1}^{m} w_{jk}(t).\,\delta_k(t).\,\sigma'\left(\sum_{i=0}^{n} w_{ij}(t)\, y_i(t)\right)$$

Pour tous les neurones j d'une couche cachée reliés à m neurones k de la couche suivante, et n neurones i de la couche précédente où y_i Correspond à l'entrée x_i du réseau connectée au neurone j ou bien à La sortie du neurone i de la couche précédente.

3 - Modifier tous les poids du réseau en appliquant la règle du delta Généralisé:

$$w_{ij}(t+1) = w_{ij}(t) + \propto (t).\,\delta_j(t).\,y_i(t)$$

Où yi correspond à l'entrée xi du réseau connectée au neurone j ou bien à la sortie du neurone i de la couche précédente.

La théorie se rattachant à l'apprentissage des RNA peut paraître complexe à première vue. Bien que certains algorithmes soient réellement complexes, l'apprentissage supervisé d'un réseau à propagation avant avec un des algorithmes les plus populaire, soit « Back propagation », est plutôt simple. Voici les différentes étapes à suivre lors de l'apprentissage d'un réseau de neurones à propagation avant avec l'algorithme « Back propagation ».

1ère étape : Initialiser les poids des liens entre les neurones. Souvent une valeur entre 0 et 1, déterminée aléatoirement, est assignée à chacun des poids.

2ème étape : Application d'un vecteur entrées-sorties à apprendre.

3ème étape : Calcul des sorties du RNA à partir des entrées qui lui sont appliquées et calcul de l'erreur entre ces sorties et les sorties idéales à apprendre.

4ème étape : Correction des poids des liens entre les neurones de la couche de sortie et de la première couche cachée selon l'erreur présente en sortie.

5ème étape : Propagation de l'erreur sur la couche précédente et correction des poids des liens entre les neurones de la couche cachée et ceux en entrées.

6ème étape : Boucler à la 2e étape avec un nouveau vecteur d'entrées-sorties tant que les performances du RNA (erreur sur les sorties) ne sont pas satisfaisantes.

Exemple:

1 - on calcule y_1, y_2, y_3 (on n'oublie pas le seuil w0 qui n'est jamais représenté):

$$\begin{cases} y_1 = \sigma.(x_1 w_{11} + x_2 w_{21} - w_{01}) \\ y_2 = \sigma.(x_1 w_{12} + x_2 w_{22} - w_{02}) \\ y_3 = \sigma.(x_1 w_{13} + x_2 w_{23} - w_{03}) \end{cases} \quad \textbf{(IV.B.13)}$$

Puis y_4 et y_5 :

$$\begin{cases} y_4 = \sigma.(y_1 w_{14} + y_2 w_{24} - w_{04}) \\ y_5 = \sigma.(y_2 w_{25} + y_3 w_{35} - w_{05}) \end{cases} \quad \textbf{(IV.B.14)}$$

2 - on calcule les erreurs de la couche de sortie:

$$\begin{cases} \delta_4 = (d_4 - y_4).\sigma'(y_1 w_{14} + y_2 w_{24} - w_{04}) \\ \delta_5 = (d_5 - y_5).\sigma'(y_2 w_{25} + y_3 w_{35} - w_{05}) \end{cases} \quad \textbf{(IV.B.15)}$$

Et les erreurs de la couche cachée:

$$\begin{cases} \delta_1 = w_{14}.\delta_4.\sigma'(x_1 w_{11} + x_2 w_{21} - w_{01}) \\ \delta_2 = (w_{24}.\delta_4 + w_{25}.\delta_5).\sigma'(x_1 w_{12} + x_2 w_{22} - w_{02}) \\ \delta_3 = w_{35}.\delta_{45}.\sigma'(x_1 w_{13} + x_2 w_{23} - w_{03}) \end{cases} \quad \textbf{(IV.B.16)}$$

3 - on calcule la nouvelle valeur de chaque poids entre la couche d'entrée et la couche cachée:

$$\begin{cases} w_{11}(t+1) = w_{11}(t) + \alpha(t).\delta_1(t).x_1 \\ w_{21}(t+1) = w_{21}(t) + \alpha(t).\delta_1(t).x_2 \\ w_{12}(t+1) = w_{12}(t) + \alpha(t).\delta_2(t).x_1 \\ w_{22}(t+1) = w_{22}(t) + \alpha(t).\delta_2(t).x_2 \\ w_{13}(t+1) = w_{13}(t) + \alpha(t).\delta_3(t).x_1 \\ w_{23}(i+1) = w_{23}(i) + \alpha(i).\delta_3(i).x_2 \end{cases} \quad \textbf{(IV.B.17)}$$

et entre la couche cachée et la couche de sortie:

$$\begin{cases} w_{14}(t+1) = w_{14}(t) + \alpha(t).\delta_4(t).x_1 \\ w_{24}(t+1) = w_{24}(t) + \alpha(t).\delta_4(t).x_2 \\ w_{25}(t+1) = w_{25}(t) + \alpha(t).\delta_5(t).x_2 \\ w_{35}(t+1) = w_{35}(t) + \alpha(t).\delta_5(t).x_3 \end{cases} \quad \textbf{(IV.B.18)}$$

On vient de réaliser un pas d'apprentissage. Il faut recommencer ces opérations pour tous les vecteurs d'apprentissage, puis tester la qualité de l'apprentissage avec les vecteurs de test qui n'ont pas servi à l'apprentissage: ce qui permet de tester les capacités de la généralisation du réseau.

L'algorithme de rétro propagation du gradient consiste à effectuer une descente de gradient sur la fonction de coût déjà utilisée pour le neurone seul:

$$\varepsilon(\vec{w}, k) = \frac{1}{2}(d(k) - y(k))^2 \quad \textbf{(IV.B.19)}$$

Où y est la sortie du réseau (donc une somme pondérée de sigmoïdes) et non la sortie d'un neurone seul.

En dérivant cette expression par rapport à chaque poids w, on retrouve la règle d'apprentissage donnée dans l'algorithme.

IV -F- PROPRIETES ET PROBLEMES

IV -F-1- PROPRIETE D'APPROXIMATION UNIVERSELLE :

Si chaque neurone a une fonction de sortie de type sigmoïde (ex.: tanh), et plus généralement si cette fonction est non linéaire (sauf polynomiale) et dérivable, alors Cybenko a démontré en **1989** qu'un réseau multicouche est capable d'approximer n'importe quelle fonction, à condition que le nombre de neurones dans la couche cachée soit suffisant.

Le problème est qu'on ne connaît pas de méthode systématique permettant de donner a priori le nombre de neurones nécessaires dans la couche cachée pour une application donnée. Cependant il existe des méthodes qui permettent de faire évoluer la structure du réseau en cours d'apprentissage:

Une méthode par élagage consiste à initialiser le réseau avec un très grand nombre de neurones dans la couche cachée et à supprimer ceux d'entre eux dont les poids synaptiques sont très faibles et n'influencent pas trop le comportement du réseau.

Une méthode par construction consiste à rajouter des neurones dans la Couche cachée au fur et à mesure des «besoins», i.e. lorsque l'erreur globale du réseau ne diminue plus.

IV -F-2- PROBLEME DES VALEURS INITIALES DES POIDS DU RESEAU :

Un autre problème est le temps de convergence de l'algorithme de rétro propagation. En effet, plus la somme pondérée des entrées d'un neurone est forte, plus le neurone se trouve dans la zone de saturation de sa fonction d'activation σ (tanh), donc plus la dérivée σ' est faible (i.e. la pente de la fonction σ en zone de saturation), et moins les poids du neurone sont modifiés.

Il faut donc démarrer l'apprentissage en initialisant les poids du réseau à des valeurs suffisamment faibles qui placent la fonction d'activation dans sa zone linéaire: on choisit donc en général des valeurs initiales inférieures à 0.1.

IV -F-3- PROBLEME DU SUR APPRENTISSAGE :

Il faut aussi donner suffisamment d'exemples bien répartis (i.e. représentatifs) pour que le réseau généralise correctement, mais pas trop pour qu'il ne fasse pas de sur apprentissage (i.e. de l'apprentissage par cœur) au détriment des capacités de généralisation.

Un moyen simple de vérifier qu'il n'y a pas sur apprentissage consiste à comparer l'erreur quadratique globale du réseau qui décroît toujours, et l'erreur faite par le réseau sur la base de test qui diminue puis augmente lorsqu'il y a sur apprentissage. La base de test ne doit jamais servir à l'apprentissage.

IV -F-4- PROBLEME DU REGLAGE DU PAS D'APPRENTISSAGE :

Le réglage du pas d'apprentissage $\alpha(t)$ joue aussi un rôle important dans la vitesse de convergence.

Ainsi il est préférable qu'il soit grand au début de l'apprentissage, et diminue au fur et à mesure que le réseau se rapproche de la solution. (La valeur du pas d'apprentissage est de l'ordre de 0.1 à 0.001).

IV -G- APPLICATION DES RESEAUX DE NEURONES

IV -G-1- IDENTIFICATION DE SYSTEME :

L'identification de système **[37][38][39]** ou identification paramétrique est une technique de l'automatique consistant à obtenir un modèle mathématique d'un système à, partir de mesures

IV -G-2- PRINCIPE ET OBJECTIVE :

L'identification consiste à appliquer des signaux de perturbation à l'entrée d'un système (par exemple pour un système électronique, ceux ci peuvent être de type binaire aléatoire ou pseudo aléatoire, Galois, sinus à fréquence multiple…) et en analyse la sortie dans le but d'obtenir un modèle purement mathématique.

L'identification peut se faire soit dans le temps (espace temporel) ou en fréquence (espace de place). Eviter les modèles purement théoriques à partir des équations physiques (en général des équations différentielles) qui sont longs à obtenir et souvent trop complexes pour le temps de développement donné.

IV -G-3- LES DIFFERENTS TYPES DE MODELE

Le principe d'une identification paramétrique est d'extraire un modèle mathématique à partir d'observations. Le modèle doit permettre de calculer la sortie du procédé y à n'importe quel instant t si les conditions initiales du système sont connues. Pour cela on peut se servir des valeurs des entrées aux instants présents et précédents *(u(t), u(t-1)….)* et des valeurs précédentes de la sortie *(y(t-1), y(t-2))* dans le cas d'un modèle régressif.

En général, le modèle est représenté sous forme de fonction de transfert utilisant la Transformée en Z. L'identification nécessite une structure de modèle connu a priori pour venir identifier dans cette structure les différents paramètres.

Voici les 3 structures de modèles les plus utilisés .

- **Le modèle ARX** : Le modèle ARX (Auto Régressive model with eXternal inputs) est un modèle auto régressif qui inclut des entrées u(t) et un bruit blanc ξ(t) de moyenne nulle.
- **Le modèle ARMAX** : Le modèle ARMAX (Auto Régressive Moving Ave rage with eXternal inputs) reprend les attributs du modèle ARX mais inclus une fonction de transfert avec une moyenne ajustable sur le bruit blanc. En général le bruit blanc permet de modéliser des perturbations non mesurables dans le modèle. Or, ces perturbations non mesurables (fluctuations thermiques, vibrations du sol…) sont rarement de moyenne nulle et peuvent aussi répondre à un modèle.
- **Le modèle ARIMAX (ou CARIMA)** : Dans le modèle ARIMAX (Auto Régressive Integrated Moving Average with eXternal inputs) le modèle du bruit est directement intégré

IV -G-4- PROCEDURE D'IDENTIFICATION :

Pour obtenir un modèle consistant il est important d'exister le processus avec toutes les fréquences de sa plage de fonctionnement .le signal appliquer doit donc être riche en fréquences (posséder un large spectre).on général on applique un signal périodique pseudo aléatoire.

Il est important de respecter une procédure rigoureuse pour identifier **[40]** un procède :

- Détermination d'un protocole de test : propriété statistique des signaux d'entre pour balayer toutes les fréquences intéressantes et le nombre de point de mesure doit être significatif pour le test>100.
- Détermination de la structure du modèle : type du modèle, ordre et retard
- Identification : choix d'un algorithme pour trouver le modèle en minimisant les erreurs entre la mesure et le modèle, en général algorithme basé sur la méthode des moindres carrés.

- Validation du modèle : réalisation de plusieurs tests de vérification .il est nécessaire pour cette étape d'utiliser des mesure déférentes de celle utiliser lors de l'identification.

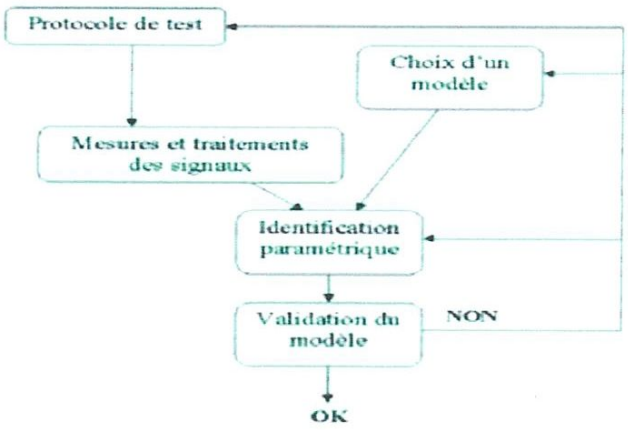


Fig. IV.B.12. Procédure d'identification

1Pour la démarche pratique d'une identification **[41]** il faut suivre les démarches suivantes:

- L'expérience
- La caractérisation
- Choix de la met horde
- La validation de la méthode
- Réalisation pratique

IV -G-5- IDENTIFICATION NEURONALE :

Identification : c'est l'obtention d'un **[42][43][44]** modèle mathématique pour représenter le comportement dynamique d'un processus.

Types de modèle :

Parfois modèle de connaissances (équations différentielles) mais :

- Souvent trop idéaliser par rapport au système réel.
- Difficulté à connaître la valeur des paramètres physiques.
- Lourdeur des calculs pour résolution des équations .différentielle.

Le plus souvent, modèle mathématique simple (en général linéaire) avec des paramètres sans signification physique : c'est l'identification « boite noir ».

Différente structures de réseaux, ainsi que différents algorithmes d'apprentissage, ont été développées pour des réseaux de taille plus modeste. Ces derniers ont prouve leur efficacité dans différente domaine tel que : la reconnaissance de formes, le traitement du signal, filtrage, la reconnaissance vocale, l'aide a la décision ainsi que la simulation .Ils ont été aussi utilisé dans le domaine de l'automatiques, soit pour la commande de système .contrôle, soit pour l'identification de procèdes.

IV -G-6- PRINCIPE DE L'IDENTIFICATION PAR RESEAUX DE NEURONE :

Tous les travaux sont basés sur des méthodes d'apprentissage supervisé avec des systèmes automatiques échantillonnés. La méthodologie de l'identification classique a été conservée mais adapter aux réseaux de neurones .elle passe par quatre étapes :

- Choix de la structure du réseau.
- Choix de l'entre.
- Algorithme d'apprentissage.
- Validation.

Il y a deux structures de base de modèle d'identification réseau de neurone ; la structure parallèle et la structure séries parallèle.

IV -G-7- IDENTIFICATION PARALLELE .

La structure parallèle utiliser la chaîne de retour directe des sorties du réseau **(FigV.B.13).**Elle estime les sorties et emploi ces évaluations pour prévoir les futures sortie .cependant. Cette structures ne garantie pas la stabilité en raison de la chaîne de retour.

IV -G-8- IDENTIFICATION SERIE PARALLELE :

La structure série parallèle n'emploi pas la chaîne de retour. Au delà de cela, elle emploi la sortie de l'installation réelle pour prévoir les futures sorties. Le back propagation statique est employé et généralement la stabilité et la convergence sont garanties avec cette méthode **[45]**.

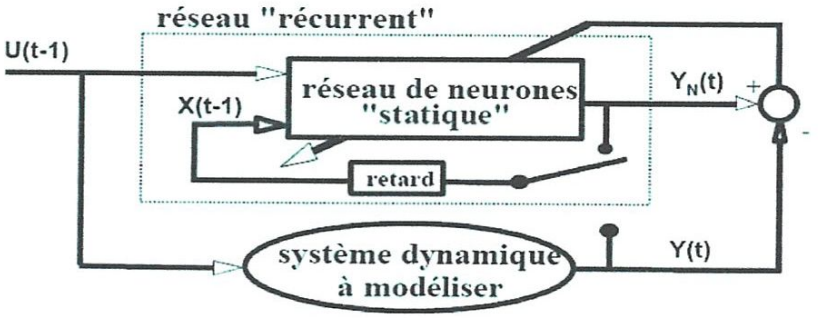


Fig IV.B.13. Identification parallèle est série parallèle

IV -G-9- QUELQUES DOMAINES D'APPLICATION :

L'identification par réseaux de neurones requiert à priori la sélection des entres du réseau, ainsi que la topologie interne du réseau. Une des formes les plus simples d'une telle topologie, est le Perceptron Multicouches (MLP), c'est un réseau à une seule couche cachée, sa capacité d'approximation universelle a fait de lui un bon choix pour faire la modélisation des systèmes.

Il existe plusieurs techniques bien établies pour la structure de modèle linéaire tel que le modèle de la représentation d'état linéaire FIR (réponse pulsionnelle finie), ARX (auto régressive avec entrées exogènes), ARMAX (auto régressive a moyenne ajustée est entres exogènes) mais leur aptitudes à représenter des procédés non-linéaires est limitée donc,

plusieurs structures de modèle de type boite noires non linéaire ont été développe inclut modèle à espace d'état non linéaire FIR, NARX, NOE, NARMAX. Ils peuvent être utilisés en injonction avec le réseau de neurone **[43]**

Le choix d'un tel modèle dépend de l'application, des informations disponibles et de la complexité du modèle ; le modèle NARX est le plus utilisé vu sa simplicité et sa structure non récursive.

Exemple : identification d'un processus

Les réseaux comportant des neurones de type adaline sont capable de résoudre des systèmes linéaire .la commande *[w, b]= solvelin (x, t)* Permet de déterminer un réseau de type Adeline en calculant les poids w et les biais b par une méthode algébrique

IV -G-10. INTERET DU NEURONAL EN IDENTIFICATION .

- Propriété d'approximateur universels parcimonieux implique modèle non - linéaires quelconques
- Rapidité de calcul en exécution
- Capacité naturelle à réaliser un modèle

IV -H- COMMANDE NEURONALE

En commande de procédé, le neuronal, n'a pas besoin de modèle analytique du processus à commander .cette caractéristique se révèle intéressante dans le cas de modèles non linéaires difficilement modélisables mathématiquement .Il s'agit de faire de la commande de processus par un réseau de neurones **Fig. IV.B.14**

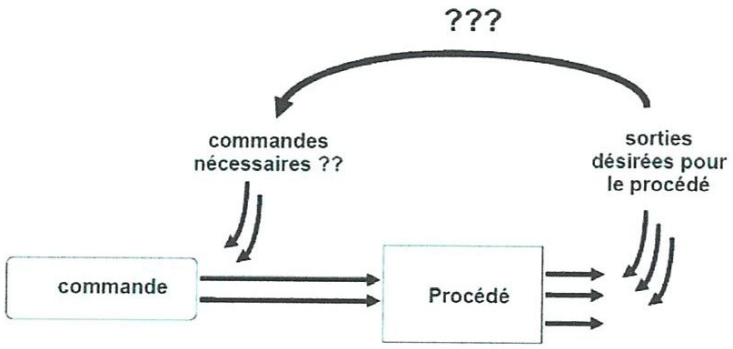


Fig. IV.B.14. Commande de processus par un réseau de neurones

IV -H-1- PRINCIPE :

Un dilemme se pose lorsqu'il s'agit de faire de la commande neuronale d'où proviennent les informations concernant le contrôle du processus, sachant que c'est le réseau lui-même qui doit les produire .pour résoudre ce problème, un certains nombre de solutions existent.

IV -H-2- SOLUTION :

Copier un régulateur déjà existant l'intérêt de cette méthode ne semble pas évidant .En effet pourquoi copier un régulateur qui existe déjà ? Il existe deux cas ou cette copie d'un régulateur par un réseau de neurones est intéressante .le première cas se présente lorsque ce régulateur est un être humain .le deuxième cas est celui ou le réseau peut réguler le processus à partir d'un modèle moins difficile à évaluer que celui qui est demandé par le régulateur habituel.

IV -H-3- COMMANDE SUPERVISEE :

Il est possible d'enseigner à un réseau de neurone les actions correctes en employant un contrôleur existant ou réfroaction humaine .ce type de commande supervisée **[46].**mais pourquoi nous voudrions copier un contrôleur existant qui fait déjà le travail ?la plus part des contrôleurs de tradition sont basé autour d'un point de fonctionnement .Ceci signifie que le contrôleur peut fonctionner correctement si le plant /processus fonctionne autour d'un certain point .les contrôleurs échoueront s'il y a n'importe quelle sorte d'incertitude ou de changement de l'usine inconnue. Les avantage de la neuro-commande est si une incertitude à l'usine se produit le réseau de neurone pourra adapter les paramètre et maintien commander l'usine quant d'autre contrôleurs robustes a échouer .Dans la commande superviser, un professeur fournit des actions correctes pour le réseau de neurones pour apprendre .Dans la figures **Fig. IV.B15**, la formation des cibles est fournie par un contrôleur existant, le réseau de neurone ajuste ses poids jusques ce que le rendement de réseau de neurone soit semblable au contrôleur**[46].**

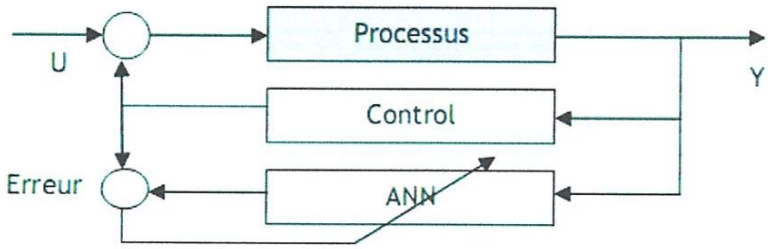


Fig. IV.B15. La commande supervisée.

Quant le réseau de neurone est ferme, il est placé dans la boucle de la chaîne de chaîne de retour .Puisque le réseau de neurone est exercé en utilisant les cibles existantes de contrôleur, il devrait pouvoir commander le processus .A ce stade, il y a un réseau de neurone qui commande le processus semblable au contrôleur existant .l'avantage de la neuro- commande est la capacité d'être adaptatif en ligne **Fig. IV.B16**. Le signal d'erreur (signal de sortie désiré signal de sortie réel) est calculé et employer pour ajuster les poids en ligne.

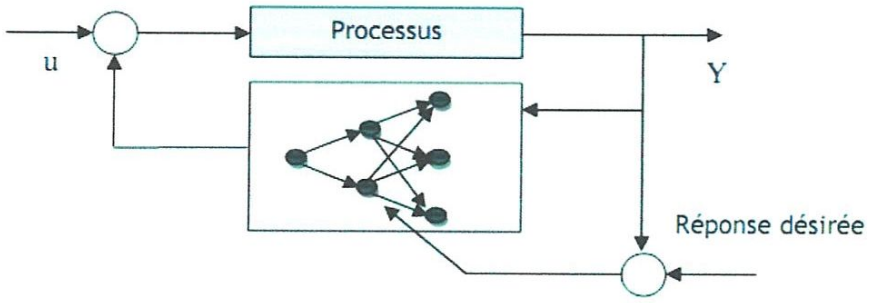


Fig. IV.B16. La commande neurone adaptatif.

IV -H-4- COMMANDE PAR MODELE DE REFERENCE :

L'utilisation d'un modèle de référence permet de bénéficier plus rationnellement, Quant c'est possible, de la connaissance a priori du système pour synthétiser la commande .Dans cette met horde la fonction du coût instantanée n'est pas choisie pour rapprocher le plus possible, a chaque pas du temps, l'objective souhaité mais de la connaissance du système contrôle et des capacités de actionneurs **Fig. IV.17 [46].**

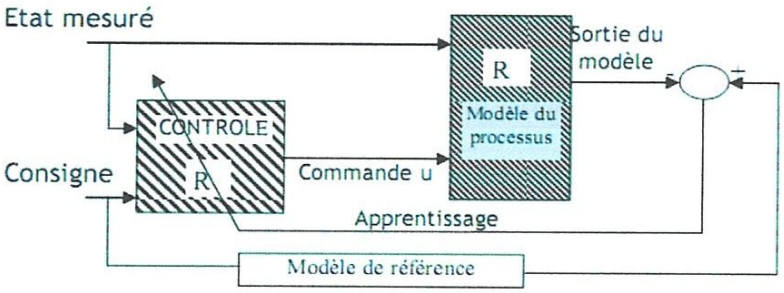


Fig. IV.17 Apprentissage d'un contrôleur avec modèle de référence

IV -H-5- COMMANDE PAR MODELE INTERNE :

La figure ci-dessous **Fig. IV.B18** présente le modèle de IMC (interna model control).comme son nom l'indique, la commande avec mode interne met en jeu, autre un contrôleur, un modèle du processus, dit modèle interne ; l'erreur de modélisation est utiliser pour modifier la consigne, si bien que le système est robuste aux erreurs de modélisation, ce qui n'est pas le cas pour la commande par modèle inverse **[47].**

En vertu de cette propriété, le comportement de tout système statique peut être approches par un réseau de neurone non bouclé approprie, et celui de tout système dynamique par un réseau boucle. C'est cette propriété, conjuguée à l'existence d'algorithme d'apprentissages performants, qui fait l'intérêt des réseaux de neurones formels pour l'automatique.

L'identification par réseaux de neurones requiert à priori la sélection des entres du réseau, ainsi que la topologie interne du réseau. Une des formes les plus simples d'une telle topologie, est le Perceptron Multicouches (MLP), c'est un réseau à une seule couche cachée, sa capacité d'approximation universelle a fait de lui un bon choix pour faire la modélisation des systèmes.

Il existe deux méthodes essentielles de conception d'un contrôleur neuronal :

- Méthodes directes : ou le réseau de neurone est lui-même le contrôleur .la méthode la plus fondamentale est nommée commande direct inverse .elle utilise le modèle inverse identifier du processus comme contrôleur .c'est la commande que nous allons utilise **[48].**
- Méthodes indirectes : le contrôleur utilise le réseau de neurones pour prédire la sortie du système.

Il existe deux approches d'apprentissage possible, celle du modèle direct d'un système, et celle de l'apprentissage du modèle inverse.

IV -I-1- IDENTIFICATION DU MODELE DIRECT [49] [50] :

Il s'agit de réaliser l'apprentissage direct d'un processus .les séquence d'entres et de sortie désirées définissant le modèle du système, sont la séquence des entres de commande u (k) appliquer et les sortie du processus y (k) mesurées pendant le fonctionnement la **Fig. IV.B.21** illustre la procédure d'apprentissage du modèle neuronale direct d'un système.

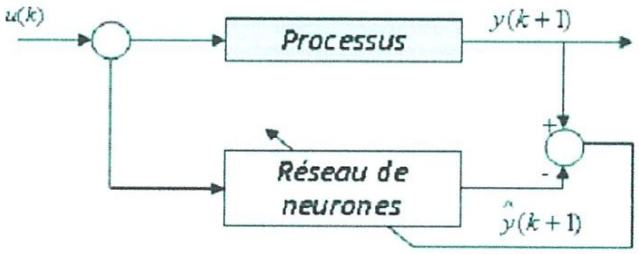


Fig. IV.B.21 Identification du modèle direct par RNA

IV-I-2-IDENTIFICATION DU MODELE INVERSE

Dans l'apprentissage générale, l'objective est de minimiser le critère de performance suivantes :

$$j(\theta) = \sum_{i=i}^{N} (y(t) - \hat{y}(t))^2$$

Comme illustre par la figure suivante **Fig. IV.B22** :

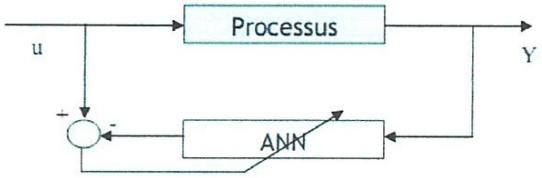


Fig. IV.B22 Identification du modèle inverse par R NA

Il existe deux méthodes pour l'établissement du modèle inverse :

Apprentissage sépare : appelée aussi apprentissage généralisé

Apprentissage en temps réel :(ou apprentissage spécialisé)

Cette met horde d'identification est utiliser pour le control du four électrique ventilé. Cependant, et puisque l'identification que nous allons est celle d'un système dynamique (four ventilé), on doit donc choisir un réseau de neurones dynamique **[50].**

IV -J- CONCLUSION

Les réseaux de neurones permettent de résoudre, avec une grande précision, les problèmes de modélisation ou d'identification linéaires et non -linéaires. On dispose d'un système dont on connaît les entrées et les sorties quantitatives. A partir d'un certain nombre d'exemples fournis aux réseaux, ceux-ci apprennent à se comporter comme le système. Une fois cet apprentissage réalisé on obtient des modèles ayant un nombre réduit de paramètres. L'apprentissage de ces réseaux peut ensuite être aisément renforcé sur de nouveaux exemples.

Au-delà de la modélisation et de l'identification, les réseaux de neurones ont un grand nombre d'applications potentielles.

Chapitre

V

Commande Du Four Par Modèle Inverse Neuronal

V-A- INTRODUCTION

Les régulateurs classiques ne sont pas adapté aux processus thermique ils sont conçu pour des processus linéaire sans retard .or un four est un système non linéaire. Afin d'améliorer le control et le maintien de la température du four électrique ventilé. On doit utiliser une nouvelle stratégie de commande à base de réseaux de neurones. Ces derniers sont des techniques puissantes de traitement non linéaire de données et qui ont fait leurs preuves dans de nombreux domaines d'application à caractère industriel.

L'utilisation d'une commande **[51][52]** directe par modèle inverse basé sur les réseaux de neurone est la méthode appliquée dans ce présent travail.

V-B- SYNTHESE D'UNE COMMANDE << NEURONALE>> PAR INVERSION DU MODELE DU PROCESSUS

La met horde la plus simple pour construire **[42]** un système de commande neuronal à partir d'un modèle du système identifier sous forme d'un réseau de neurone en boucle ouverte est l'inversion directe du modèle .le système de commande est alors simplement l'inverse du modèle du processus .si ce modèle est non linéaire, son inverse l'est généralement : il peut être constitue par un réseau de neurone, dont l'apprentissage et l'utilisation sont schématiser sur **la Fig. V.1.**

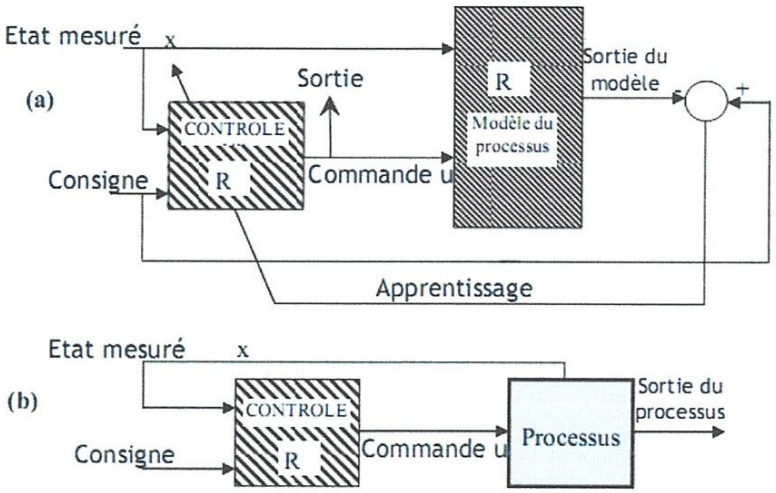


FIG.V-1. Principe de l'apprentissage (a) et de l'utilisation (b) d'une commande neuronale en boucle fermée par inversion du modèle

Dans cette figure, on a adjoint au réseau de neurone qui constitue le modèle du processus un réseau de neurone qui calcule la loi de commande .ce réseau est aussi un réseau non bouclé qui pour entre l'état et d'une façon optionnelle, la consigne désiré (état au temps suivant) dans le cas ou l'on souhaite que cette consigne soit variable. Sinon, le contrôleur admet pour entre unique l'état du système au temps k, la sortie du contrôleur neuronal est la commande au temps qui lors de l'apprentissage, est appliquer à l'entré de commande du modèle, et qui lors de l'utilisation est appliquer à l'entre du processus.

L'ensemble (contrôleur+modèle) constitue un réseau de neurones non bouclé qui admet pour sortie l'état au temps suivant .l'apprentissage s'effectue en minimisant la différence entre l'état désiré ou consigne et la sortie du réseau .seuls les paramètres du contrôleur (poids et biais) sont variable et modifiés par le processus d'apprentissage .les paramètre du modèle restent inchangés par le processus d'apprentissage qui ce traduit dans la figure par des hachures de style différent.

La fonction de coût est généralement un écart quadratique entre la sortie désire et la sortie mesure .si des contrainte sont imposer à la commande, elle peut l'être directement dans le réseau contrôleur. Par exemple, si la commande admissible est bornée, on peut exprime ces contraintes dans les fonction d'activation de la couche du sortie du contrôleur (sigmoïde).on peut aussi exprimé ces contrainte en introduisant une sortie auxiliaire au niveau du contrôleur et en rétro propageant une pénalité qui dépend de la commande produite.

V-C- PRINCIPE DE LA COMMANDE DU FOUR PAR MODELE INVERSE NEURONAL

La mise en équation des différents phénomènes physiques intervenant dans le procédé **[51][52]** conduit à des équations non linéaires qui ne traduiront qu'imparfaitement le comportement du système. On a choisi par conséquent, de rechercher un modèle de comportement du procédé qui repose sur l'exploitation de mesures expérimentales caractéristiques de son fonctionnement dynamique: c'est l'objet de l'identification. Ce modèle est exprimé sous forme d'une fonction de transfert discrète.

Cette dernière utilise l'inverse du système Four **(figure Fig. V.2)** comme contrôleur, d'où l'appellation "commande par modèle inverse neuronale". Ce modèle inverse neuronal d'un processus donné est un réseau de neurones qui permet de calculer l'entrée (ou le signal de commande) à partir des valeurs antérieures de ces entres sortie.

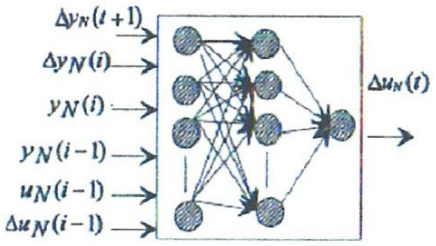


Fig. V.2 contrôleur neuronal

L'étape de modélisation nécessite l'acquisition d'un fichier d'entrées-sorties du processus pour que les différentes modes du processus soient existées, on appliquera un signal riche en fréquence : un signal rampe autour duquel on superpose une séquence binaire pseudo aléatoire afin d'avoir un modèle qui puisse être utilisé en tout point de fonctionnement.

On crée ainsi un fichier d'entrées-sorties que l'on utilisera pour l'apprentissage hors ligne du réseau de neurone.

A partir d'un fichier contenant u(t) et y(t) on calcule les variations

$$\Delta u(t) = u(t) - u(t-1) \;\; et \;\; \Delta y(t) = y(t) - y(t-1)$$

Avec l'utilisation de la sigmoïde unipolaire, on doit normaliser les différents signaux u(t), y(t), Δu(t) et Δy(t) entre les valeurs limites 0.1 et0.9.

V-C-1- SCHEMA DE LA SIMULATION ENTRES SORTIE POUR L'APPRENTISSAGE DU RESEAU:

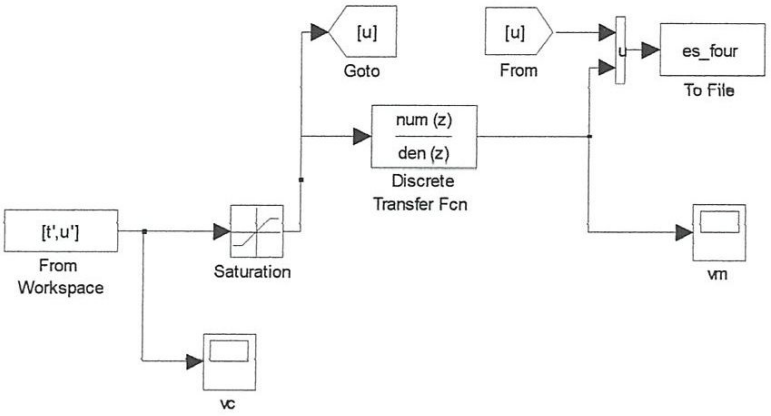
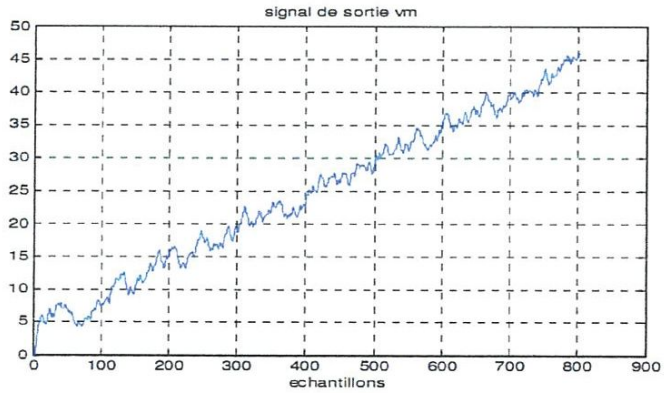


Fig. V.3 entrées sortie pour l'apprentissage du réseau

V-C-2- AFFICHAGE DES SIGNAUX SORTIE ET DE COMMANDE :



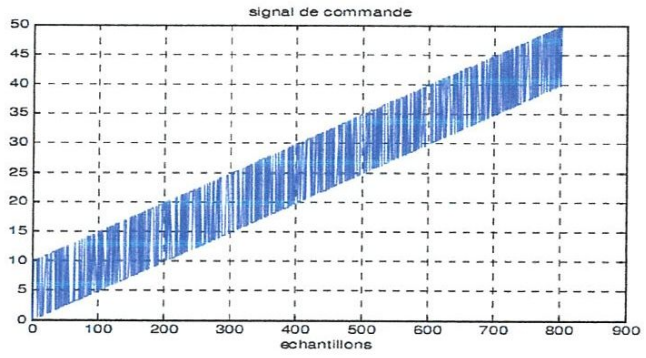


Fig. V.4. Affichage des signaux (de sortie et de commande)

Nous choisissons l'architecture suivante dans laquelle les blocs N et DN représentant respectivement les opérations de normalisation et de dé normalisation des signaux .

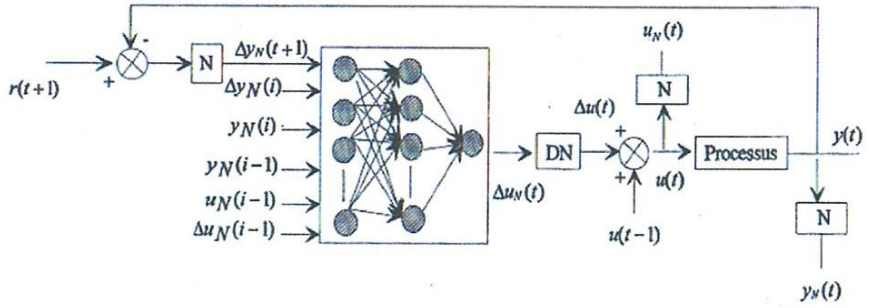


Fig. V.5. Commande par modèle inverse neuronal du four [1]

L'apprentissage est une procédure en ligne liée au contrôle du modèle référence; il constitue donc un contrôle adaptatif. L'idée est de réduire au minimum le critère j(θ): **Fig.V.6.**

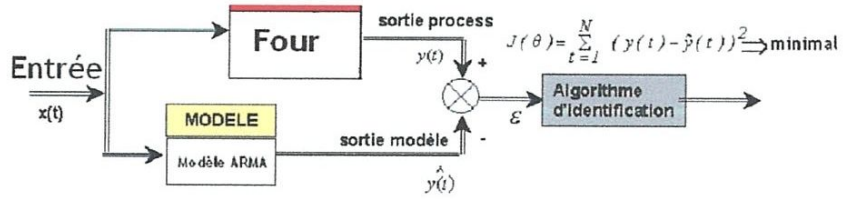


Fig. V.6. Principe de réduction du critère j(θ)

$$j(\theta) = \sum_{t=1}^{N} (y(t) - \hat{y}(t))^2$$

On a considéré dans ce présent article le cas de l'identification d'un modèle **ARMA** d'un processus du type

$$H(z) = z^{-1} \frac{b_1 + b_2 z^{-1}}{1 - a_1 z^{-1}} \qquad \textbf{(V.2)}$$

L'équation de récurrence liant la sortie estimée ŷ (t) à la commande u (t),

$$\hat{y}(t) = a_1 y(t-1) + b_1 u(t-1) + b_2 u(t-2) \quad \textbf{(V.3)}$$

Peut être mise sous la forme

$$\hat{y}(t) = [a_1 \quad b_1 \quad b_2] \begin{bmatrix} y(t-1) \\ u(t-1) \\ u(t-2) \end{bmatrix} \quad \textbf{(V.4)}$$

Selon la loi de Widrow-Hoff **[33]**, à l'instant d'échantillonnage t = k Te, la sortie estimée ŷ (t) peut être considérée comme la cible d'un neurone linéaire.

Elle s'écrit en fonction des poids W et du biais b sous forme

$$\hat{y}(t) = W(k)P(k) + b(k) \quad \textbf{(V.5)}$$

Où,

$$W(k) = [a_1 \quad b_1 \quad b_2]$$

$$P(k) = [y(t-1) \; u(t-1) \; u(t-2)]^T \quad \textbf{(V.6)}$$

$$b(k) = 0$$

L'apprentissage du réseau consiste à modifier, à chaque pas, les poids et les biais afin de minimiser la somme des carrés des erreurs en sortie en utilisant la loi de Widrow-Hoff. L'erreur à minimiser s'écrit alors sous la forme.

$$er(k) = y(k) - \hat{y}(k) \quad \textbf{(V.7)}$$

La loi de commande s'écrit alors sous la forme

$$u(t) = \frac{1}{H(p)} r(t) \quad \textbf{(V.8)}$$

L'équation de récurrence liant la commande u (t) à la consigne r (t),

$$u(t) = \frac{1}{b_1} r(t) + \frac{a_1}{b_1} r(t-1) - \frac{b_2}{b_1} u(t-1) \quad \textbf{(V.9)}$$

Peut être mise sous la forme

$$u(t) = \left[\frac{1}{b_1} \quad \frac{a_1}{b_1} \quad -\frac{b_2}{b_1}\right] [r(t) \; r(t-1) \; u(t-1)]^T \quad \textbf{(V.10)}$$

$$u(t) = W(k)P(k) + b(k) \quad \textbf{(V.11)}$$

Où,

$$W(k) = \left[\frac{1}{b_1} \quad \frac{a_1}{b_1} \quad -\frac{b_2}{b_1}\right] \quad \textbf{(V.12)}$$

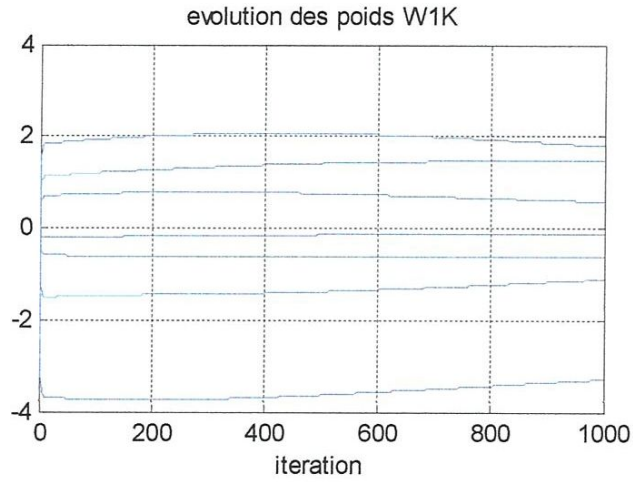
$$P(k) = [r(t)\ r(t-1)\ u(t-1)]^T$$

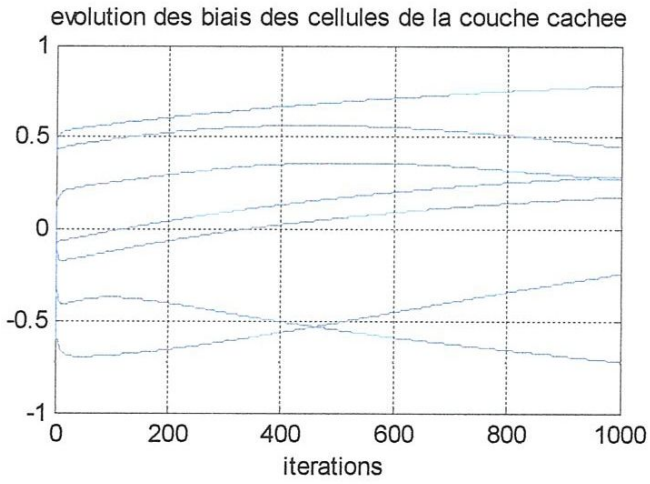
$$b(k) = 0$$

V-C-3- RESULTANTS DE LA SIMULATION :

Le logiciel utilisé est le **MATLAB 7**. Le schéma de principe du four a été réalisé à partir de **SIMULINK6.** Deux S-fonction ont été utilisées pour l'incorporation des programmes d'identification et de commande codés en langage C.

L'évolution des différent poids et biais est représentée par les courbes suivantes





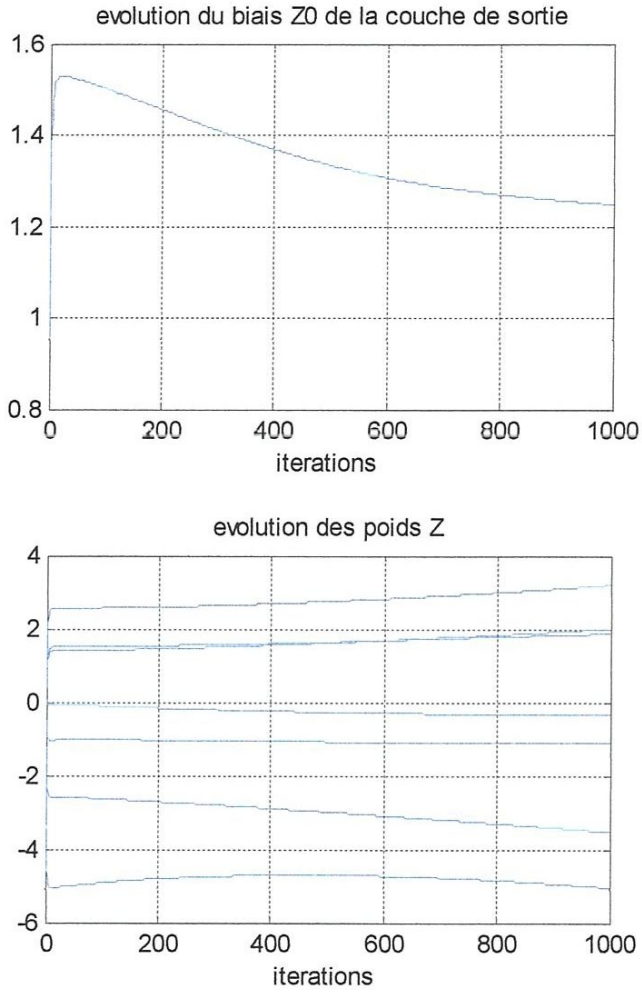


Fig. V.7. Evolution des poids et des biais

V-C-4- INTERPRETATION DES RESULTATS

L'évolution des poids et du biais à chaque itération de l'apprentissage est donnée par la **figure(V.7)** Au bout de la 45ème itération, on obtient une bonne convergence des poids et du biais

V.D.COMMANDE INVERSE NEURONAL

Pour réaliser un système de commande avec modèle inverse neuronal, la première cellule de la couche d'entrée recevra la valeur normaliser de l'écart entre la valeur future r (t+1) de la consigne et la valeur courante de la sortie y (t), du processus

Le signal issu de la cellule de sortie du réseau, après sa dé normalisation, multiplier par un gain k, est ajouter a la commande appliquer au processus à l'instant d'échantillonnage précédent .la valeur du gain k influe sur la stabilité du système bouclé

Comme son l'indique, le modèle inverse est utiliser directement comme contrôleur il commande le processus en boucle fermée .le schéma de la commande neuronale est représenter par la **Fig. V.8.**

Suivant où la sortie du réseau .cors pondant à la variation de commande normalisé, est multipliée par un gain k

V-D-1- SCHEMA GLOBAL DE LA SIMULATION :

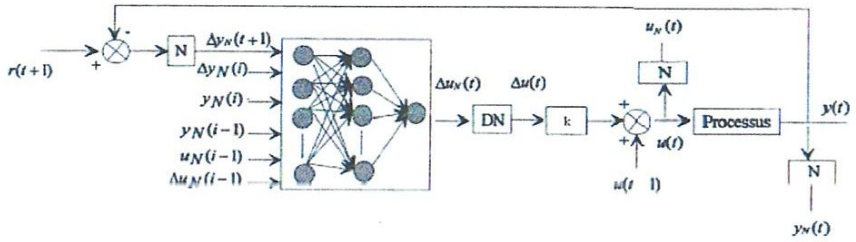


Fig. V.8. Schéma global de la simulation

V-D-2- RESULTAT DE LA SIMULATION :

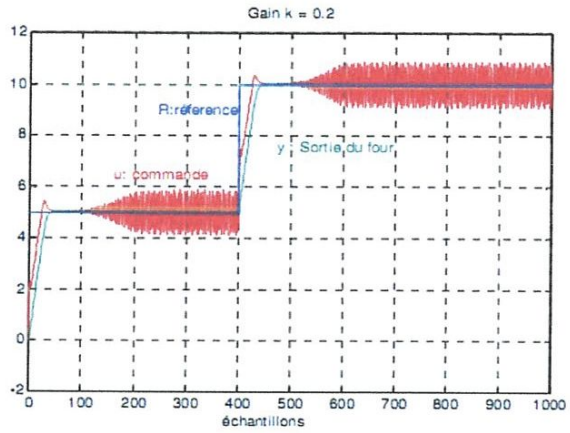


Fig. V.9. Sortie du système avec un gain k = 0.2.

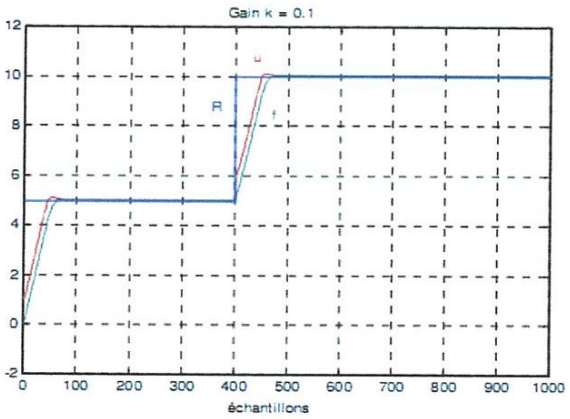


Fig. V.10. Sortie du système avec un gain k = 0.1.

V-D-3- INTERPRETATION DES RESULTATS :

Du fait que le réseau fournit la variation de commande, l'erreur statique est nulle.

Le temps de réponse est nettement meilleur avec une diminution remarquable du dépassement dans le régime transitoire. Il y a juste des oscillations dans le régime permanent comme le montre la **Fig. V. 9.**

En jouant sur le réglage grâce à un gain k introduit à la sortie du réseau nous sommes parvenus à éliminer ces oscillations **Fig. V.10** et à atteindre un régime permanent presque parfait puisque l'erreur statique est tellement petite qu'on peut la considérer comme nulle.

V-E- Conclusion :

Le régulateur analogique ainsi que la commande intégrale avec compensation des pôles et des zéros pouvait être des commandes idéales pour le four si les facteurs non -linéaires du système n'ont pas affecté ses performances.

L'implantation d'algorithmes de commande, comparée à une réalisation purement analogique offre de nombreux atouts.

Un avantage décisif réside dans une souplesse d'emploi exceptionnelle. Il est très facile de modifier les logiciels codant les algorithmes de réglage et d'identification de façon à ce que les non linéarités du système n'affectent pas ses performances.

De plus, l'approche basée sur les réseaux de neurones possède un grand nombre de méthodes pour surmonter des problèmes où il est impossible d'obtenir avec précision les modèles des processus et des perturbations comme dans le cas des fours électriques.

Grâce à l'apprentissage en ligne, les algorithmes d'identification et de réglage dans le cas de la commande par modèle inverse neuronal peuvent détecter toute perturbation extérieure au système, de plus, le régulateur qui n'est autre que l'inverse du modèle est très facile à déterminer.

Conclusion Générale

CONCLUSION GENERALE

Dans ce présent travail, notre objectif est l'utilisation de nouvelles stratégies de commande afin d'améliorer le control et le maintien de la température du four électrique ventilé. Ces stratégies devraient faire face d'une manière efficace aux perturbations extérieures et aux variations des paramètres du système contrôlé.

Notre attention s'est fixée sur le contrôle et le maintien de la température du four électrique ventilé en utilisant un contrôleur à base de réseaux de neurones. Ces derniers sont des techniques puissantes de traitement non linéaire de données et qui ont fait leurs preuves dans de nombreux domaines d'application à caractère industriel.

On a eu en premier lieu fait le réglage par le contrôleur de type PI (commande intégral avec compensation des pôles et des zéros) puis le réglage par retour d'état linéaire classique. Ces deux régulateurs n'ont pas permis de maîtriser les régimes transitoires suite aux perturbations. Ainsi, ces deux types de réglage nécessitent la connaissance avec précision le modèle du four électrique ventilé, aussi bien, ils présentent une grande non -linéarité. Une variation paramétrique conduit à une dégradation des performances du système de contrôle.

Pour remédier aux inconvénients de ces contrôleurs classiques (PI et retour d'état linéaire), une approche basée sur les réseaux de neurones a était utilisée.

Cette approche possède un grand nombre de méthodes pour surmonter des problèmes où il est impossible d'obtenir avec précision les modèles des processus et des perturbations comme dans le cas des fours électriques.

L'utilisation d'une commande directe par modèle inverse basé sur les réseaux de neurone a été appliquée. Pour l'identification du modèle inverse du système à régler, on a considéré le modèle ARMA. Le modèle inverse neuronal est utilisé directement comme contrôleur afin de commander le processus en boucle fermée.

L'utilisation des réseaux de neurones a donné de bonnes performances dynamiques du système four avec:

- Bonne convergence des poids et du biais.
- Diminution remarquable du dépassement dans le régime transitoire.
- Erreur statique est nulle en régime permanent.
- Temps de réponse meilleur de la température du four.

Comme perspectif d'avenir de notre travail, nous proposons l'apprentissage en ligne des réseaux de neurones afin de détecter toutes perturbations extérieures au système. Ceci est obtenu grâce aux algorithmes d'identification utilisés dans la commande par modèle inverse neuronal.

Bibliographie

BIBLIOGRAPHIE

[1] [D5910]- "four électrique à résistances. Présentation générale ", par Jean- François BOURGEOIS, Alain GIRAULT, Richard JAUME, Marianne LE BOULCH EDF Division Recherche &Développement et Claude OBERLIN Ingénieur Senior SEE© Techniques de l'ingénieur Dossier : D5910 et Date de parution : 02/2005.

[2] [D5911]- "four électrique à résistances Technologies de mise en oeuvre ", par Jean- François BOURGEOIS, Alain GIRAULT, Richard JAUME, Marianne LE BOULCH EDF Division Recherche &Développement et Claude OBERLIN Ingénieur Senior SEE© Techniques de l'ingénieur Dossier : D5911 et Date de parution : 02/2005.

[3] [D5912]- "four électrique à résistances Application industriello", par Jean François BOURGEOIS, Alain GIRAULT, Richard JAUME, Marianne LE BOULCH EDF Division Recherche &Développement et Claude OBERLIN Ingénieur Senior SEE© Techniques de l'ingénieur Dossier : D5912 et Date de parution : 08/2007.

[4] [D5913]- "four électrique à résistances pour en savoir plus " , par Jean- François BOURGEOIS, Alain GIRAULT, Richard JAUME, Marianne LE BOULCH EDF Division Recherche &Développement et Claude OBERLIN Ingénieur Senior SEE© Techniques de l'ingénieur Dossier : D5913 et Datedeparution: 02/2005.

[5] SACCADURA (J. F), "initiation aux transfert thermique ".Ed. Technique et Documentation Paris1978

[11] Mokhtari M et Marie M, "Application de Matleb 5 et Simulink 2", Ed.Springer, 1998.

[12] P.Codron et S. le Bâlois, " automatique, système linéaire et continus", Date de parution : 12/05/1998 Editeur : Dunod -ISBN : 2-10-003826-5.

[13] F. Manne ville et S. Esquieu, " Système bouclé linéaire", Editeur : Dunod : EAN13 : 9782100490226 Date de parution 1992.

[14] GRANJOR (Yves), "Systèmes linéaire, non linéaire, temps continu, temps discret, représentation d'état et cours exercices corrigés", .Date de parution : 01/06/2003 Dunod nombre de page : 382 ISBN 2-10-007118-1 www.dunod.com/pages/ouvrages/ficheouvrage.asp?id=9782100071180 - 20k.

[15] BOURLES (Henri), "Système linéaire de la modélisation à la commande", Editeur : Hermès Science Publication (15mars 2006), Pages : 279. ISBN-10 :2746213001-Lavoisier, 2006-www.amazon.fr.

[16] B. Bergeron, " Une représentation d'état pour la commande à temps discret ", séminaire du groupe de travail en commande Robuste, IRCyN, Nantes, 25mai 1999.

[17] M.CHADLI. "Commandes de système linéaires ", école supérieur, d'Ingénieurs en électrotechnique et électronique (ESIEE), Amiens.14GSP-14GSE ,2006 /2007.

[18] D.ALaZard. "Introduction aux filtre de Kalman ", Note de cours, Exercices corriger, Session Malab.Janvier 2005-Version-0.0, Page27-34.

[19] Brown R.G., Hwang P.Y.C., "Introduction to Random signals and Applied Kalman Filtering", New York, Willey §Sounds, Inc, 1992.

[20] Kamoun S. " Développement de méthodes d'estimation paramétrique et d'état de systèmes stochastiques ", CD-ROM de la huitéme conférence internationale des Sciences et Techniques de l'automatiqueSTA'2007,5-7.

[21] Arnold Heemink, Delft University, "Technology data assimilation using Kalman Interring", Summer school INRIA, CEA, EDF (2006).

[22] Sorenson H., Kalman Filtering, "Theory and Application, New York", IEEE Press, 1985.

[23] A. ELKebir, A Chaker, S Zebirate,"Article commande d'un four Electrique Ventilé Par Retour d'état En utilisant le Reconstructeur de Kalman", première Conférence Internationale Sur l'Electrotechnique, l'Electronique l'Automatique et la Maintenance CIEAM'08.

[24] E.M. Petriu, "Neural Networks for Modelling Application", University of Ottawa, 1998.

[25] M. Mokhtari, A.Mesbah, "Apprendre et maîtriser Matleb version 4 et 5 et Simulink ",1997.

[26] H. Demuth, M. Beale, "The Neural Network toolbox for Use with Mat lab", the math, Inc, Copyright 1977.

[27] J. François Jodouin, " Les Réseaux de Neurones : Principes et Application", Hermès Sciences Publicat, 21sept1994.

[28] Y. Morere, "Réseaux de Neurones", www.math-info-paris5.fr /-bouzy/Doc/AA /Réseaux de Neurones 1.pdf.

[29] R.JEAN, "Neuron-Fuzzy Mode Ling: Architecture, Analysis and Application", PhD Thesis, University of California, Berkley 1998.

[30] H.R. Bherenji, p. Khedkar, "Learning and Tuning Fuzzy Logic controllers Through Reinforcements",IEEE Transactions on Neural Network, vol (3), pp724-740, 1996.

[31] D.Hebb, "The Organisation of Behaviour", New York: Wiley, 1949.

[33] Windrow B.and lehr M.A, "30years of adaptive neural networks Perceptron, Madaline, and back propagation", Proceeding of IEEE, 78, 1415-1442, 1991.

[34] F. Rosenblatt, "The Perceptron: a probabilistic model for information storage and organisation in the brain", Psychological Review65:386-408-1958.

[35] MASSOUM Ahmed, " Commande à Structure Variable Commande Neuro-Floue ", p115, Thèse soutenue à sidi bel abbés, Algérie, 2007.

[36] PH.LERY, "la Rétro propagation",http://servasi.insa-rouen.fr;2000.

[37] LANDAUID, " Identification et commande des systèmes" ,deuxième édition revue et augmentée, 1993 (ISBN 2-86601-365-4)

[38] NAHAYO.Fulgence, " Projet-Examen:Identification des systèmes", Université libanaise DEA Modélisation et Simulation Informatique, page : 1, 2,3..11, Mai 2009.

[39] Ljung, "System Identification", Theory for the User (ISBN 0-13-881640-9).