الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

Université Mohamed Khider – Biskra

Faculté des Sciences et de la technologie

Département : Génie électrique

Ref :……

جامعة محمد خيضر بسكرة
كلية العلوم و التكنولوجيا
قسم: الهندسة الكهربائية
المرجع:……

Mémoire présenté en vue de l'obtention
Du diplôme de

Magister en : Automatique

Option : Commande Identification des systèmes dynamique

IDENTIFICATION DES SYSTEMES NON LINEAIRES PAR RESEAUX DE NEURONES

Présenté par :

GACEM Salah Eddine

Soutenu publiquement le : 08/10/2015

Devant le jury composé de :

Dr. ALLAG Abdelkrim	**Professeur**	**Président**	**Université d' Eloude**
Dr. BOUMAHRAZ Mohamed	**Professeur**	**Rapporteur**	**Université de Biskra**
Dr. TERKI Nadjiba	**Maitre de Conférences 'A'**	**Examinateur**	**Université de Biskra**
Dr. ABDOU Latifa	**Maitre de Conférences 'A'**	**Examinateur**	**Université de Biskra**

Remerciements

Ce travail a été réalisé au sein du département de Génie Electrique de l'université de Biskra.

Mes sincères remerciements à Monsieur Dr.Mostefa TOUBA et Pr.Mohamed BOUMAHRAZE Maître de Conférences à l'Université de Biskra d'avoir pris le temps de juger ce travail et de m'avoir fait l'honneur d'être rapporteur de mon mémoire de magister.

Enfin, je tiens tout particulièrement à remercier Monsieur Dr.Abd Alrazak GEUTTAF pour leur aide.

Je dédie ce modeste travail

A mes parents
A ma femme
A ma petite fille
A mes frères et sœurs
A tous mes amis
A tous ceux qui m'ont aidé à
faire ce travail.

Remerciements

Ce travail a été réalisé au sein du département de Génie Electrique de l'université de Biskra.

Mes sincères remerciements à Monsieur Dr.Mostefa TOUBA et Pr.Mohamed BOUMAHRAZE Maître de Conférences à l'Université de Biskra d'avoir pris le temps de juger ce travail et de m'avoir fait l'honneur d'être rapporteur de mon mémoire de magister.

Enfin, je tiens tout particulièrement à remercier Monsieur Dr.Abd Alrazak GEUTTAF pour leur aide.

Je dédie ce modeste travail

A mes parents
A ma femme
A ma petite fille
A mes frères et sœurs
A tous mes amis

A tous ceux qui m'ont aidé à faire ce travail.

Sommaire

SOMMAIRE i
LISTE DES FIGURES v
INTRODUCTION GENERALE 1

Chapitre I:Identification des systèmes

I.1.INTRODUCTION 3
I.2.MODELISATION DES PROCESSUS 3
I.3.PRINCIPE DE LA MODELISATION MATHEMATIQUE 3
I.3.1.Modélisation théorique 4
I.3.2.Modélisation expérimentale 4
I.3.2.1.Extraction de données 4
I.3.2.2.Choix de la structure du modèle 4
I.3.2.3.Choix du critère d'estimation paramétrique 4
I.3.2.4.Estimation paramétrique 5
I.3.2.5.Validation du modèle 5
I.4.IDENTIFICATION DES PARAMETRES D'UN PROCESSUS DYNAMIQUE 6
I.4.1.Identification non paramétrique 6
I.4.2.Identification paramétrique 6
I.5.STRUCTURES DES MODELES NON LINEAIRES 7
I.5.1.Modèles de systèmes à base de blocs structurés 9
I.5.2.Modèles de systèmes à base d'une approche multimodèles 12
I.5.3.Modèle à base de réseaux de neurones 14
I.6. CHOIX DES CRITERES D'ESTIMATION PARAMETRIQUE ET DE SELECTION DE MODELE 14
I.6.1. Choix du critère d'estimation paramétrique 15
I.6.2. Choix du critère de sélection de modèle 15
I.7. ESTIMATION DES PARAMETRES DE MODELES 16
I.7.1.Méthodes d'estimation globale 17
I.7.1.1.Optimisation linéaire 17
I.7.1.2.Optimisation non linéaire 18
I.8.CONCLUSION 23

Chapitre II: Les réseaux de neurones artificiels

II.1.INTRODUCTION 24
II.2.HISTORIQUE 24
II.3. ELEMENTS DE BASE ET TERMINOLOGIE 25
II.3.1.Modélisation biologique 25
II.3.1.1.Le corps cellulaire 26
II.3.1.2.Les dendrites 27
II.3.1.3.L'axone 27
II.3.2.Fonctionnement des neurones 27
II.3.3.Le neurone formel 28
II.3.4.Fonction d'activation 29
II.4.DEFINITION ET PROPRIETES DES RESEAUX DE NEURONES 30
II.4.1. Le parallélisme 30
II.4.2. La capacité d'adaptation 31
II.4.3. La mémoire distribuée 31
II.4.4. La résistance aux pannes 31
II.4.5. La généralisation 31
II.5. STRUCTURES DE CONNEXIONS 31
II.5.1.Cas général 31
II.5.2.Les réseaux à couches 31
II.5.3.Les réseaux entierement connectés 32
II.6. L'APPRENTISSAGE 32
II.6.1.L'apprentissage supervisé 32
II.6.2.Apprentissage non supervisé 33
II.6.3 Apprentissage par renforcement 33
II.7.ARCHITECTURE DES RESEAUX DE NEURONES 34
II.7.1. Les réseaux statique 34
II.7.1.1.Réseaux multicouche 35
II.7.1.2.Réseaux RBF (fonctions radiales de base) 39
II.7.3. Les réseaux dynamique 42
II.7.3.1.Les cartes auto-organisatrices de Kohonen (CAOK) 42
II.7.3.2.Les réseaux de Hopfield 42
II.7.3.3.Le ART 43

II.8. AVANTAGES DES RESEAUX DE NEURONES 43
II.9. INCONVENIENTS DES RESEAUX DE NEURONES 44
II.10. DOMAINES D'APPLICATION..... 44
II.10.1.Reconnaissance des formes 44
II.10.2.Modélisation 44
II.10.3.Traitement de la parole 44
II.10.4.Traitements dépendant du temps 45
II.10.5.Détection d'anomalies en médecine 45
II.11.CONCLUSION 45

Chapitre III: Identification des systèmes non linéaires par réseaux de neurones

III.1.INTRODUCTION 46
III.2.ETAPES D'IDENTIFICATION 46
III.2.1. Structure d'erreur de prédiction (série-parallèle) 46
III.2.2. Structure d'erreur de sortie (parallèle) 47
III.3. IDENTIFICATION DE SYSTEMES NON LINEAIRES PAR RESEAUX DE NEURONES 48
III.3.1.Le modèle NFIR (Réponse non linéaire impulsionnelle finie) 49
III.3.2. Le modèle NARX (Non linéaire autorégressif avec entrée exogène) 49
III.3.3.Le modèle NOE (Erreur de sortie non linéaire) 50
III.3.4.Le modèle NARMAX 51
III.4.IDENTIFICATION PAR UN RESEAU MLP (PERCEPTRONS MULTICOUCHES) 52
III.4.1.Etudes et simulations 52
III.5.IDENTIFICATION PAR UN RESEAU RBF (Fonctions de base radiale) 59
III.5.1.Etudes et simulations 59
III.5.CONCLUSION 64

Chapitre IV: Identification de la machine asynchrone par réseaux de neurones

IV.1.INTRODUCTION 65
IV.2.MODELE D'ETAT DE LA MACHINE ASYNCHRONE 65

IV.3.IDENTIFICATION DE LA MACHINE ASYNCHRONE PAR RESEAUX DE NEURONES 66
IV.3.1.Identification par perceptron multicouche (Multilayer Perceptron MLP) 67
IV.3.1.1.Choix de l'architecture du réseau MLP 68
IV.3.1.2.Choix de l'algorithme d'adaptation 68
IV.3.1. 3.Simulations 69
IV.3.2.Identification par réseau à base de fonction radiale (RBF) 73
IV.3.2.1.Simulations 75
VI.4.CONCLUSION 80

CONCLUSION GENERALE 81
ANNEXE I 83
REFERENCES BIBLIOGRAPHIQUES 94

Liste des figures

Figure I.1. Identification d'un système.....5
Figure I.2.Schéma d'identification paramétrique 6
Figure I.3. Modèle de type Hammerstein. 10
Figure I.4. Modèle de type Wiener. 11
Figure I.5. Modèle de type Wiener-Hammerstein. 12
Figure I.6.Représentation graphique de la méthode de Newton. 20
Figure II.1.Schéma d'une synapse. 26
Figure II.2. Anatomie d'un neurone. 27
Tableau II.1 Transition entre le neurone biologique et le neurone formel. 28
Figure II.3. Le neurone formel. 29
Figure II.4. Les fonctions d'activation (décision) les plus utilisées. 30
Figure II.5. Apprentissage supervisé. 33
Figure II.6. Apprentissage non supervisé. 34
Figure II.7. Structure d'un réseau MLP. 36
Figure II.8. Structure d'un réseau RBF. 41
Figure II.9. Réseau à connexions récurrentes. 43
Figure.III.1.Erreur de prédiction. 48
Figure.III.2.Erreur de sortie. 49
Figure.III.3. Structures le modèle NFIR par réseaux de neurones 50
Figure.III.4. Structures le modèle NARX par réseaux de neurones 51
Figure.III.5. Structures le modèle NOE par réseaux de neurones 51
Figure.III.6. Structures le modèle NARMX par réseaux de neurones 52
Figure III.7.Sortie du procédé et sortie du modèle. 54
Figure III.8.Erreur d'identification par MLP. 54
Figure III.9.Sortie du procédé et sortie du modèle. 55
Figure III.10.Erreur d'identification par MLP. 55
Figure III.11.Sorties du procédé et sorties du modèle. 56
Figure III.12.Erreur d'identification par MLP. 56
Figure III.13.Sorties du procédé et sortie du modèle. 57
Figure III.14.Erreur d'identification par MLP. 57
Figure III.15.Sorties du procédé et sortie du modèle (première sortie). 58

Figure III.16.Erreur d'identification par MLP. 59
Figure III.17.Sorties du procédé et sortie du modèle (deuxième sortie). 59
Figure III.18.Erreur d'identification par MLP. 59
Figure III.19.Sorties du procédé et sortie du modèle. 60
Figure III.20.Erreur d'identification par RBF 61
Figure III.21.Sorties du procédé et sortie du modèle. 61
Figure III.22.Erreur d'identification par RBF 61
Figure III.23.Sorties du procédé et sortie du modèle. 62
Figure III.24.Erreur d'identification par RBF 62
Figure III.25.Sorties du procédé et sortie du modèle. 63
Figure III.26.Erreur d'identification par RBF 63
Figure III.27.Sorties du procédé et sortie du modèle (première sortie). 64
Figure III.28.Erreur d'identification par RBF 64
Figure III.29.Sorties du procédé et sortie du modèle (deuxième sortie). 64
Figure III.30.Erreur d'identification par RBF 65
Figure.VI.1. Identification de la machine asynchrone par réseaux de neurones. 67
Figure.VI.2. Modèle neuronal de la machine asynchrone. 68
Figure.VI.3. Courant statorique suivant l'axe *d*. 70
Figure.VI.4. Erreur d'identification par MLP de Courant statorique suivant l'axe *d* 71
Figure.VI.5. Courant statorique suivant l'axe *q*. 71
Figure.VI.6. Erreur d'identification par MLP de Courant statorique suivant l'axe *q* 71
Figure.VI.7. La vitesse rotorique 72
Figure.VI.8. Erreur d'identification par MLP du vitesse rotorique 72
Figure.VI.9. Courant statorique suivant l'axe *d*. 72
Figure.VI.10. Erreur d'identification par MLP de Courant statorique suivant l'axe *d* 73
Figure.VI.11. Courant statorique suivant l'axe *q*. 73
Figure.VI.12. Erreur d'identification par MLP de Courant statorique suivant l'axe *q*. 73
Figure.VI.13. La vitesse rotorique 74
Figure.VI.14. Erreur d'identification par MLP du vitesse rotorique 74
Figure.VI.15. Courant statorique suivant l'axe *d*. 76
Figure.VI.16. Erreur d'identification par RBF de Courant statorique suivant l'axe *d* 77
Figure.VI.17. Courant statorique suivant l'axe *q*. 77
Figure.VI.18. Erreur d'identification par RBF de Courant statorique suivant l'axe *q*. 77

Figure.VI.19. La vitesse rotorique 78
Figure.VI.20. Erreur d'identification par RBF du vitesse rotorique 78
Figure.VI.21. Courant statorique suivant l'axe *d.* 78
Figure.VI.22. Erreur d'identification par RBF de Courant statorique suivant l'axe *q* 79
Figure.VI.23. Courant statorique suivant l'axe d 79
Figure.VI.24. Erreur d'identification par RBF de Courant statorique suivant l'axe q 79
Figure.VI.25. La vitesse rotorique 80
Figure.VI.26. Erreur d'identification par RBF du vitesse rotorique 80

INTRODUCTION GENERALE :

Dans l'industrie, l'augmentation du taux de production avec minimisation des coûts est un objectif stratégique et pour arriver à tous cela il est nécessaire de connaître le système et identifie ces paramètres. Dans les dernières années, l'identification des systèmes a connu un développement considérable tant sur le plan des applications des techniques existantes que sur le plan de développement de nouvelles techniques.

L'identification des systèmes non linéaire s'appuie sur le fait d'avoir un modèle qui représente le système non linéaire tout en se basant sur l'ajustement des paramètres de telle sorte que le modèle du système identifié soit similaire au système inconnu (réel). L'identification se fait généralement en deux étapes, dont la première consiste à trouver un modèle de représentation mathématique, ceci sera assuré à partir d'une suite de mesure d'entrée –sortie du système à identifie. Dans la deuxième étape se fera l'estimation des paramètres, cette dernière se base sur un critère d'erreur entre la sortie directe et la sortie du modèle d'identification. Lorsqu'on désire obtenir un modèle paramétrique pour un processus (système), c'est-à-dire une relation mathématique comprenant un nombre fini de termes (par exemple fonction de transfert, équation différentielle ou aux différences), on peut souhaiter en premier lieu exprimer les lois physiques connues régissant son fonctionnement et en déduire la relation mathématique cherchée; on parle alors de modèle de connaissance.

La théorie des technique d'identification sont adapté aux problèmes linéaires mais en pratique ces méthodes ne s'avèrent pas toujours applicables a cause de non linéarité. Donc des nouvelles méthodes s'avèrent nécessaire pour l'identification de ces systèmes tels que les réseaux de neurones, logique floue …etc.

L'identification des systèmes non linéaires par réseaux de neurones a fait l'objet de nombreux travaux de recherche depuis une trentaine d'années à cause de la capacité d'apprentissage, d'approximation et de généralisation que possèdent ces réseaux [50][51]. En effet, cette nouvelle approche fournit une solution efficace à travers laquelle de larges classes des systèmes non linéaires peuvent être modélisés sans une description mathématique précise.

L'objective du présent travail consiste en l'exploitation des réseaux de neurone artificiels (RNA) à l'identification des systèmes non linéaires. En effet, les RNA eux-mêmes sont des structures à modèles boite-noire non linéaires, ils ont une capacité

d'approximation générale des systèmes non linéaires. Le but visé par ce travail c'est de voir l'effet du choix de l'architecture d'un RNA sur la procédure d'identification voire la validation du modèle estimé. Plusieurs modèles non linéaires existent dans la littérature tels que : NFIR, NARX, NARMAX, etc...

Ce mémoire est composé de quatre chapitres répartis comme suit :

Le premier chapitre présente des généralités sur l'identification, quelques points importants concernant son utilité et son importance ont été donnés. Les différents types de modèles de systèmes sont définis ainsi que les étapes de l'identification, et les algorithmes d'adaptation qui sont des éléments essentiels de l'identification.

Le deuxième chapitre donne un aperçu général sur les réseaux de neurones; les définitions et les concepts de base.

Le troisième chapitre détaille le problème d'identification des systèmes non linéaires en utilisant les réseaux de neurones en utilisant deux architectures différentes (réseau multicouches MLP et réseau fonctions radiales de base RBF).

Dans le quatrième chapitre nous présenterons l'identification par réseaux de neurones sur la machine asynchrone, et pour cela en utilisant deux architectures différentes des réseaux de neurones (MLP et RBF).

Enfin, la conclusion générale présente le bilan de ce travail ainsi que les perspectives envisagées.

I.1.INTRODUCTION :

Le problème d'identification des systèmes suscite un grand intérêt et ce depuis plusieurs décennies. L'activité de recherche menée autour de ce problème a fait l'objet de très nombreux colloques et séminaires et a donné lieu à d'innombrables publications. Si l'identification des systèmes linéaires est maintenant arrivée à maturité, celle des systèmes non linéaires reste à ce jour un thème de recherche loin d'être tari.

Identifier un système, c'est caractériser un système nommé modèle à partir de la connaissance expérimentale des entrées et des sorties du système réel. On appelle le système non linéaire, un système ne pouvant pas être représenté par une équation différentielle ordinaire à coefficients constants, c'est-à-dire pour lequel le théorème de superposition ne s'applique pas.

Le problème de la détermination des lois physiques constitue un problème de modélisation qui dans certains cas, lorsque les paramètres sont connus avec suffisamment de précision, peut suffire. Le problème d'identification est d'établir des modèles mathématiques (approximatifs) basés sur des mesures prises sur le système à identifier [1].

I.2.MODELISATION DES PROCESSUS :

Les systèmes réels sont difficiles à étudier, donc on est amené à les représenter mathématiquement pour pouvoir les commander. Le problème posé est: comment obtient on ce modèle mathématique? C'est ce qu'on appelle identification ou modélisation des processus. Un processus est un objet soumis à des actions externes, à l'intérieur duquel des grandeurs interagissent, et sur lequel on peut faire des mesures. Un modèle est une représentation de la réalité visible ou observable. Modéliser c'est «remplacer du visible compliqué avec de l'invisible simple», Le but de toute modélisation de processus est de construire un modèle, c'est-à dire une représentation mathématique de son fonctionnement. Ce modèle sera utilisé pour effectuer des prédictions de la sortie du processus, ou pour la conception d'un correcteur, ou encore pour simuler le processus au sein d'un système de commande [1].

I.3.PRINCIPE DE LA MODELISATION MATHEMATIQUE :

La modélisation mathématique est une représentation qui traduit le fonctionnement d'un système à travers des relations mathématiques liant les différentes variables du système.

I.3.1.Modélisation théorique :

La représentation du système est faite à partir des lois (physiques, chimiques, biologiques, etc.) régissant le fonctionnement du système. Il est donc nécessaire d'avoir une connaissance complète du système. Cette modélisation peut présenter des difficultés lorsqu'elle est appliquée à des systèmes complexes. Les modèles de ce type sont appelés modèles de connaissance ou modèles de type « boîte blanche ».

I.3.2.Modélisation expérimentale :

La représentation est faite sur la base de données recueillie sur le système à modéliser. Cette représentation ne requiert aucune connaissance du système. Les modèles de ce type sont appelés modèles de représentation ou de type « boîte noire ». Ils sont représentés en général sous la forme d'une relation de type « entrée-sortie ».

Dans certains cas, des connaissances a priori sur le système permettent de fixer la structure du modèle. La combinaison de ces connaissances a priori et des données expérimentales recueillies permet d'aboutir à une représentation du système communément appelée modèle de type « boîte grise ».L'établissement de ce type de modèle est une procédure itérative comportant cinq phases [2] (figure.I.1) :

I.3.2.1.Extraction de données :

Durant cette phase, des mesures sont effectuées sur les variables sensés caractérisés le système. Ces variables peuvent être des variables externes qui agissent sur le système (entrées de commande ou perturbations mesurables), des variables internes qui traduisent l'état du système (variables d'état), ou la réponse du système (variable de sortie). Il existe souvent des perturbations non mesurables qui agissent sur le système (en entrée ou en sortie) rendant plus difficile sa modélisation.

I.3.2.2.Choix de la structure du modèle :

Il s'agit de définir d'une façon formelle la relation expliquant le fonctionnement du système. Cette relation correspond à une famille de fonctions mathématiques dont une seule correspond au modèle recherché.

I.3.2.3.Choix du critère d'estimation paramétrique :

C'est le choix de la fonction objectif (fonction coût) dont l'optimisation (minimisation) permet de déterminer la structure du modèle de façon unique. Ce critère est fonction de

l'écart entre la sortie du système et celle du modèle. Le critère quadratique est généralement choisi.

I.3.2.4.Estimation paramétrique :

Après avoir choisi la structure du modèle, il faut estimer les paramètres de ce dernier. Ces paramètres sont les poids de connexions entres les neurones qui sont adaptés de telle sorte à minimiser un critère de performance

I.3.2.5.Validation du modèle :

C'est une procédure qui permet d'évaluer l'exactitude (ou la fidélité) du modèle. Pendant cette phase, le modèle est testé avec des données non utilisées pendant la phase d'identification.

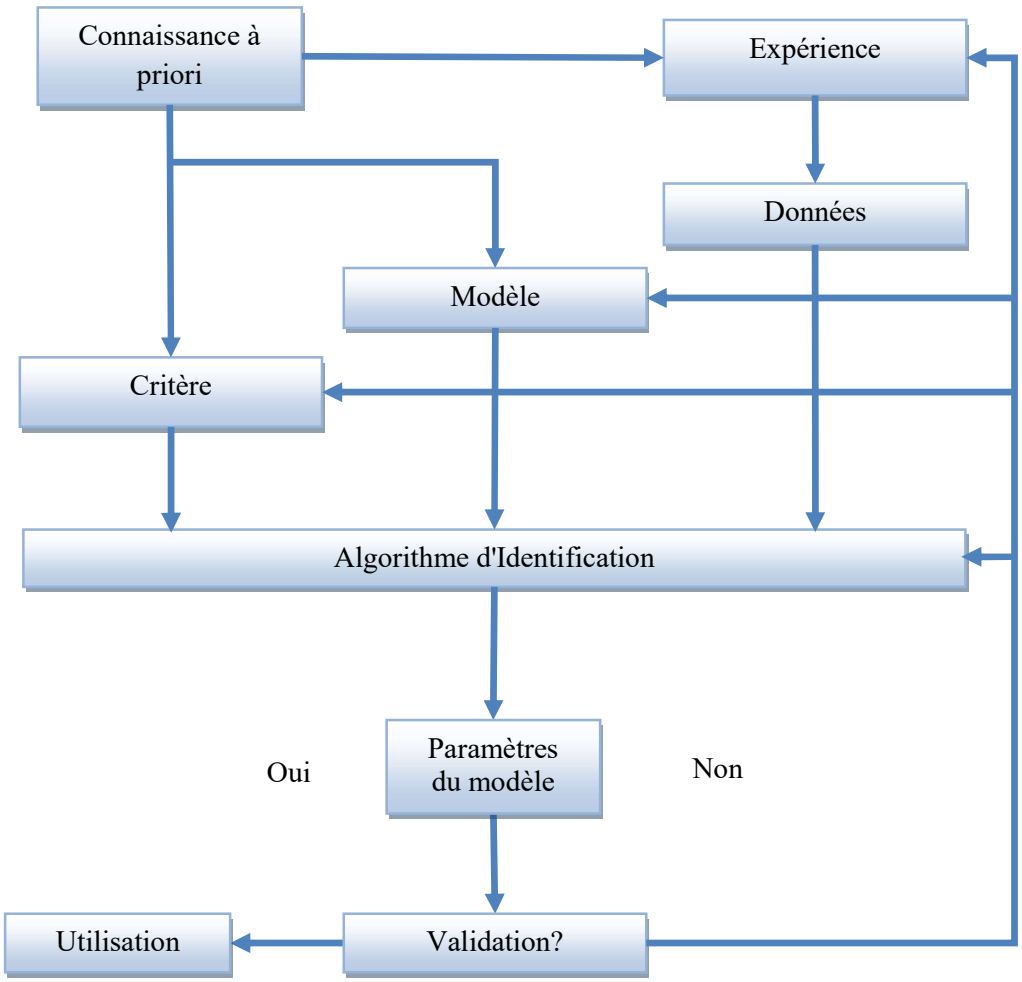


Figure I.1. Identification d'un système.

I.4.IDENTIFICATION DES PARAMETRES D'UN PROCESSUS DYNAMIQUE :

L'identification c'est l'opération de détermination des caractéristiques dynamiques d'un procédé (système). On s'intéresse à l'identification des modèles dynamiques paramétriques échantillonnées qui sont les plus appropriés pour la conception et l'ajustement des systèmes numériques [3][4].

I.4.1.Identification non paramétrique :

Les méthodes non paramétriques se comportent de façon neutre vis-à-vis des données, se refusant à inclure trop d'à priori sur la véritable nature du signal. Elles consistent à déterminer ce qui aurait été fait dans le cas idéal d'un signal déterministe connu et à bâtir des estimateurs point par point de l'autocorrélation et du spectre.

I.4.2.Identification paramétrique :

Les méthodes paramétriques consistent à ajuster un modèle aux données observées. Les paramètres des modèles, en nombre faible, caractériseront le signal ; on pourra ainsi injecter des connaissances à priori sur le processus dynamique qui a engendré le signal.

La procédure standard pour réaliser cet ajustement est l'identification paramétrique dont le schéma est rappelé en figure (I.2).

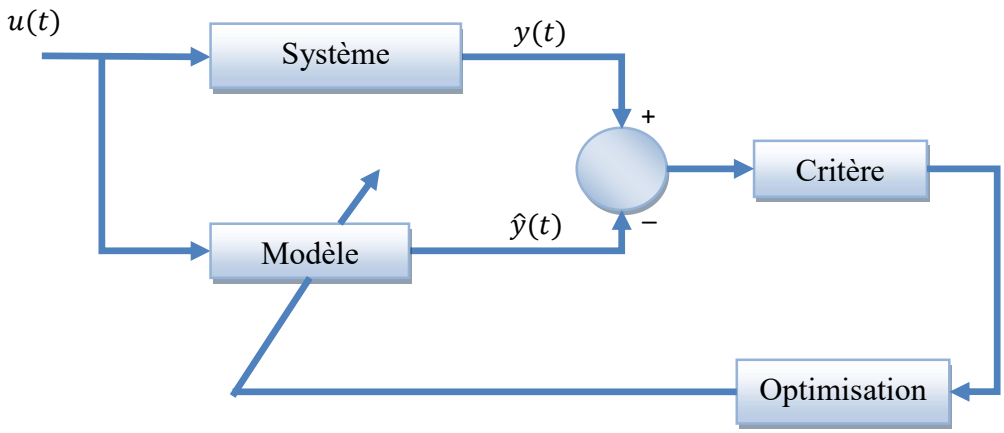


Figure I.2. Schéma d'identification paramétrique.

Cette procédure comporte un très grand nombre de variantes, correspondant au choix à priori, au niveau du modèle, du critère de mesure d'erreur de l'entrée choisie et de l'algorithme d'optimisation.

Les avantages qui peuvent en être attendus sont la souplesse de l'analyse, l'introduction naturelle d'informations à priori, la parcimonie de la représentation et des choix variés d'espèces de représentations paramétriques et offre d'autres possibilités telles que :

- Modélisation des bruits.
- Identification des modèles de perturbation.
- Détection et mesure des fréquences d'oscillations
- Analyse spectrale des signaux.

I.5.STRUCTURES DES MODELES NON LINEAIRES :

La construction d'un modèle « boite noire » d'un système repose sur l'hypothèse selon laquelle il existe une relation déterministe liant les entrées du système à sa sortie. D'une manière générale, le modèle prédictif de comportement d'un système dynamique non linéaire peut s'écrire sous la forme:

$$y_s\,(t\ +h) = F_s\big(\,u(t), \tilde{y}(t)\big) + e(t+h) \qquad \text{(I.1)}$$

Où :

$y_s(t\ +h)$ est la sortie du système à l'instant $t\ +h$, h étant le pas de prédiction ; $F_s(\cdot)$ est une fonction non linéaire déterministe inconnue, appelée prédicateur théorique ; $u(t)$est un vecteur dont les composantes sont des éléments des entrées externes du système à l'instant *t* :

$$u(t) = [u_1(t), u_1(t-1), \cdots, u_1(t-n_{u1}+1), \cdots, u_k(t), u_k(t-1), \cdots, u_k(t-n_{uk}+ 1), u_{ni}(t), u_{ni}(t-1), \cdots, u_{ni}(t-n_{uni}+1)]^T \qquad \text{(I.2)}$$

Avec n_{uk} l'ordre de l'entrée u_k, $k = 1, \ldots, n_i$, n_i étant le nombre d'entrées ; $\tilde{y}_s(t)$est un vecteur dont les composantes sont liées à l'état du système à l'instant *t* ,on peut avoir par exemple :

$$\tilde{y}(t) = [y_s(t), y_s(t\ -\ 1), \ldots, y_s(t\ -\ n_{ys}\ +\ 1)\ ]^T \qquad \text{(I.3)}$$

Avec n_{ys} l'ordre de la sortie y_s ;

e(t+h) est une variable aléatoire de moyenne nulle et de variance σ^2 représentant le bruit. La représentation formelle donnée par la relation (I.1) intègre les connaissance sa priori du

système ainsi que des hypothèses concernant son comportement : connaissance des variables descriptives, caractère statique ou dynamique, linéaire ou non linéaire, présence ou absence de perturbations.

Le modèle est statique si la sortie à l'instant *t* ne dépend que des entrées à l'instant *t*, il est dynamique si la sortie dépend aussi des entrées et des sorties précédentes. Le modèle est linéaire si $y_s(t+h)$ est une combinaison linéaire de $u(t)$ et de $\tilde{y}(t)$.Dans le cas contraire, il est non linéaire. Le modèle est récurrent si $\tilde{y}(t)$comporte des variables d'état estimés à l'instant *t* . A l'inverse, si la sortie du modèle ne dépend que des entrées et des sorties mesurées.

L'estimation paramétrique des modèles non linéaires récurrents est un problème majeur en identification.

En considérant la sortie du système comme une variable aléatoire $Y_s(t)$ dont une réalisation est $y_s(t)$, on peut écrire [5] :

$$Y_s(t+h) = E[Y_s(t+h)|t\,] + e(t+h) \tag{I.4}$$

Où est l'espérance mathématique conditionnelle de $Y_s(t+h)$, lorsqu'on dispose de toutes les informations disponibles jusqu'à l'instant *t*, et e(t +h) correspond à la partie non prédictible de $Y_s(t+h)$ à l'instant *t* (erreur de modélisation).La sortie y du système, donnée par le modèle $F_s(\cdot)$, est :

$$y(t+h) = E[Y_s(t+h)|t\,] = F_s(u(t), \tilde{y}_s(t)) \tag{I.5}$$

La fonction $F_s(\cdot)$ est approchée par la fonction $F(\cdot)$ de structure connue paramétrée par un vecteur θ. La sortie $\hat{y}$estimée par ce prédicteur réel est :

$$\hat{y}(t+h) = F(u(t), \tilde{y}_s(t), \theta) = F(\varphi(t), \theta) \tag{I.6}$$

Où $\varphi(t) = [u(t)^T, \tilde{y}_s(t)^T]^T$est le vecteur de régression ou d'information constitué de l'ensemble des variables explicatives du système dans la zone de validité du modèle. Il est obtenu par la concaténation des éléments des vecteurs $u(t)$ et $\tilde{y}(t)$. Chaque élément de $\varphi(t)$ est appelé régresseur.

L'erreur de prédiction du modèle est obtenue par :

$$\varepsilon(t) = y_s(t) - \hat{y}(t) \tag{I.7}$$

La détermination des régresseurs est un problème majeur rencontré en modélisation expérimentale. L'idéal est de ne sélectionner que les variables caractéristiques des non linéarités du système. Ces variables étant inconnues, plusieurs démarches sont proposées pour la sélection des entrées [6][7]. L'une d'elle, comme expliquée dans [7], consiste à choisir un ensemble de variables d'entrée le plus grand possible permettant d'obtenir le« modèle complet ». Les performances de ce modèle sont comparées à celles des modèles dont les variables d'entrée constituent des sous-ensembles des variables du « modèle complet». Le modèle qui présente les meilleures performances est choisi. Avec cette démarche le nombre de modèle croît de façon exponentielle avec le nombre de variables, ce qui complique la mise en œuvre. Des stratégies sous-optimales (décrites dans [7]) mais plus simples à mettre en œuvre sont souvent utilisées en pratique. Il s'agit de la stratégie d'élimination (« step wise back ward regression») et de la stratégie de construction (« step wise forward regression»).

La description du système donnée par l'équation (I.6) conduit à deux questions fondamentales : l'identification structurelle de la fonction *F(·)* et l'estimation paramétrique. L'identification structurelle consiste à choisir une structure de *F(·)* adéquate pour la description du système. L'estimation paramétrique de *F(·)* consiste à estimer la valeur $\hat{\theta}$ de θ qui minimise une norme de l'écart entre la sortie réelle du système y_s et celle du modèle $\hat{y}$ pour l'ensemble des observations :

$$\hat{\theta} = \arg\ min_{\theta}\ (\|y_s(t+h) - F(\varphi(t), \theta)\|) \qquad \text{(I.8)}$$

La valeur $\hat{\theta}$ détermine la structure choisie pour *F(·)* de façon unique.
La variété des modèles de représentation des systèmes dynamiques non linéaires repose sur le choix de la structure de la fonction *F(·)*.

I.5.1.Modèles de systèmes à base de blocs structurés :

Un moyen relativement simple pour la représentation du comportement non linéaire d'un système repose sur l'utilisation de modèles à base de blocs structurés (*Block oriented models*) [8]. Ces derniers sont élaborés grâce à la combinaison de deux blocs de base : un élément ***non linéaire statique*** et un élément ***linéaire dynamique***.

La combinaison de ces deux blocs peut conduire à l'élaboration d'un grand nombre de structures de modèles. Parmi les différents modèles à base de blocs structurés, les modèles le Hammerstein et de Wiener sont probablement les plus répandus. Ils se présentent sous la

forme d'un modèle dynamique linéaire en cascade avec un élément statique de type non linéaire.

L'identification des systèmes non linéaires à l'aide de blocs structurés remonte au milieu des années 60 [8] et a été largement abordée dans la littérature [9]. Bien que considérablement exploités pour la modélisation, la commande et le diagnostic, ces modèles font toujours l'objet de nombreux travaux de recherche.

a. Le modèle de Hammerstein :

Le modèle de Hammerstein se compose d'un élément statique ***non linéaire*** suivi d'un élément ***dynamique linéaire***. Sa structure est présentée sur la figure (I.3).

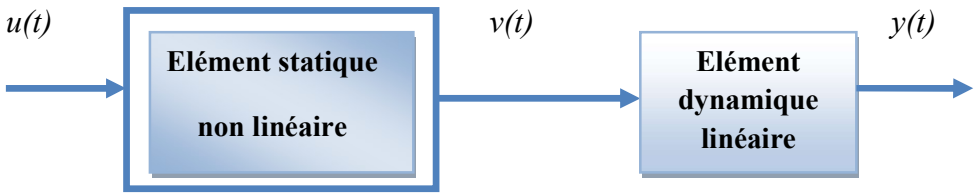


Figure I.3. Modèle de type Hammerstein.

La représentation d'état du modèle de type Hammerstein est donnée par :

$$v(t) = f(u(t)) \tag{I.9}$$

$$x(t+1) = Ax(t) + Bv(t) \tag{I.10}$$

$$y(t) = Cx(t) + Dv(t) \tag{I.11}$$

Où $x \in R^n$ est l'état, $u \in R^r$ le signal d'entrée du modèle, $v \in R^m$ le signal d'entrée du bloc dynamique linéaire, $f(.)$: $R^r R^m$ la fonction vectorielle caractérisant l'élément statique non linéaire et $y \in R^p$ le signal de sortie du modèle. Il convient de remarquer que les seules grandeurs accessibles physiquement sont le signal d'entrée $u(t)$et le signal de sortie $y(t)$. Le signal $v(t)$ est un signal fictif de modélisation et n'est pas accessible.

Les modèles de Hammerstein se révèlent bien adaptés à la caractérisation, au moyen d'un modèle linéaire, du comportement dynamique d'un système dont l'actionneur présente un caractère non linéaire (saturation, zone morte, etc..).

b. Le modèle de Wiener :

D'un point de vue structurel, le modèle de type Wiener est le pendant du modèle de Hammerstein. Il s'obtient en inversant l'ordre des éléments de base. Sa structure est présentée sur la figure (I.4) :

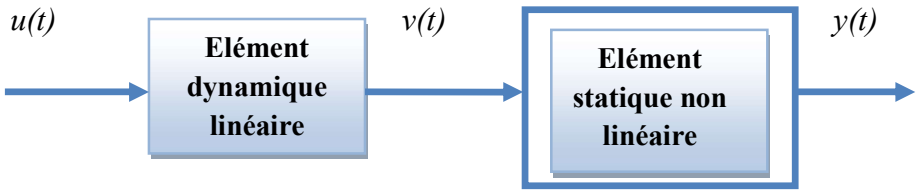


Figure I.4. Modèle de type Wiener.

La représentation d'état du modèle de type Wiener est donnée par :

$$x(t+1) = Ax(t) + Bu(t) \quad \text{(I.12)}$$

$$v(t) = Cx(t) + Du(t) \quad \text{(I.13)}$$

$$y(t) = f(v(t)) \quad \text{(I.14)}$$

Où $x \in R^n$ est l'état, $u \in R^m$ le signal d'entrée du modèle, $v \in R^r$ le signal d'entrée du bloc statique non linéaire, $f(.) : R^r R^p$ la fonction vectorielle caractérisant l'élément statique non linéaire et $y \in R^p$ le signal de sortie du modèle. Comme dans le cas du modèle de Hammerstein, les seules grandeurs physiquement accessibles sont le signal d'entrée $u(t)$ et le signal de sortie $y(t)$.

Les modèles de Wiener se révèlent bien adaptés à la caractérisation, au moyen d'un modèle linéaire, du comportement dynamique d'un système dont le capteur présente un caractère non linéaire.

c. Modèle de type Hammerstein-Wiener :

La combinaison des deux structures (Hammerstein et Wiener) permet d'élaborer des modèles plus complexes. Ainsi, par exemple, le modèle de type Hammerstein-Wiener dont la représentation d'état est donnée par :

$$y(t) = f(u(t)) \quad \text{(I.15)}$$

$$x(t+1) = Ax(t) + Bv(t) \quad \text{(I.16)}$$

$$s(t) = Cx(t) + Du(t) \quad \text{(I.17)}$$

$$y(t) = \tilde{f}(s(t)) \quad \text{(I.18)}$$

est obtenu en considérant un modèle de Hammerstein suivi d'un modèle de Wiener. Un modèle de type Wiener-Hammerstein peut également être obtenu par ce même principe mais en inversant l'ordre des modèles.

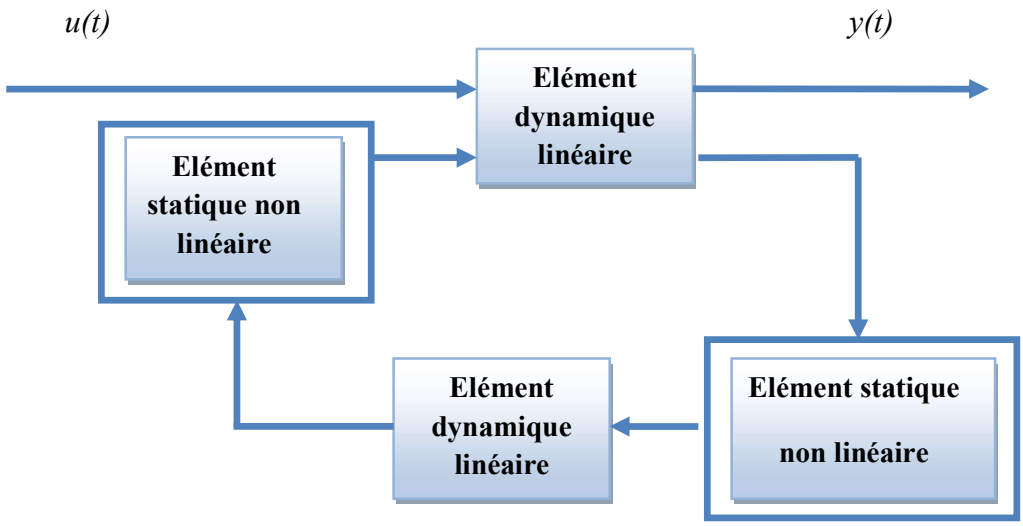


Figure I.5.Modèle de type Wiener-Hammerstein.

Il est également possible d'imaginer d'autres combinaisons plus complexes en introduisant par des boucles de rétroaction [10][11] telles que celles illustrées sur la figure (I.5). Remarquons toutefois qu'en pratique ces formes très élaborées peuvent se révéler difficilement exploitables.

I.5.2.Modèles de systèmes à base d'une approche multimodèles :

La représentation multimodèles d'un système non linéaire peut être obtenue à partir de différentes structures. Une représentation d'état de sous-modèles permet de les mettre facilement en évidence. Cette représentation d'état du multimodéles revêt l'avantage d'être compacte, simple et plus générale qu'une présentation sous la forme d'une équation de régression entrée/sortie.

Deux grandes familles de multimodèles sont répertoriées selon que les sous-modèles sont *homogènes* dans le sens où ils partagent la même structure et le même espace d'état ou *hétérogènes*, c-à-d que leur structure et leur espace d'état diffèrent. Dans la référence [12]

on met en évidence, dans un contexte de modélisation floue directement transposable au cadre multimodèle, deux structures essentielles de multimodèles. Leur différence provient de la façon dont les sous-modèles sont combinés. La première structure, connue sous l'appellation de *multimodèles de Takagi-Sugeno*, est constituée de sous-modèles partageant un vecteur d'état unique (sous-modèles homogènes). Dans la seconde, connue sous l'appellation de *multimodèles découplé*, les sous-modèles possèdent chacun un vecteur d'état indépendant (sous-modèles hétérogènes).

a. Multimodèles de Takagi-Sugeno :

La structure du multimodèles de Takagi-Sugeno (T.S) ou à états couplés, initialement proposée dans un contexte de modélisation floue par *Takagi-Sugeno* dans les années80, a été depuis, largement popularisée dans un contexte multimodèles par les travaux de *Johansen et Foss*[16]. Elle est certainement la structure la plus couramment utilisée dans le cadre de l'approche multimodèles. Le multimodèles de T.S est connu sous différents appellations : réseaux de modèles locaux à mélange de paramètres *(local model network by blending the parameters)*,multimodèles à modèles locaux couplé ou à état couplé ou encore multimodèles à état unique, etc...

La représentation dans l'espace d'état de ce multimodèles est donnée par :

$$x_i(t+1) = A_i x(t) + B_i u(t) \tag{I.19}$$

$$x(t+1) = \sum_{i=1}^{L} \mu_i(\zeta(t)) x_i(t+1) \tag{I.20}$$

$$y(t) = \sum_{i=1}^{L} \mu_i(\zeta(t)) C_i\, x(t) \tag{I.21}$$

Où $x \in R^n$ est le vecteur d'état commun aux sous-modèles, $u \in R^m$ le vecteur de commande, $y \in R^p$ le vecteur de sortie et $\zeta(t)$ la variable d'indexation des fonctions de pondération $\mu_i(.)$. Les fonctions de pondération respectent les propriétés de somme convexe.

b. Multimodèles découplé :

L'agrégation des sous-modèles peut s'effectuer par le biais d'un deuxième mécanisme d'interpolation. Dans [12] on propose un multimodèles, issu de l'agrégation de sous-modèles, qui se présente sous la forme d'une structure à états découplés. Le multimodèles

découplé apparait dans la littérature sous différentes appellations : réseaux de modèles locaux à états locaux *(local-state local model network),* multimodèles locaux *(Muliple Local Model)*, réseaux de modèles locaux par mélange des sorties (*local model network dy blending the outputs),* multimodèles sans état commun *(Multiple model with non common state),* NFDFDS *(Neuro-Fuzzy and De-coupling Fault Diagnosis Scheme).*

La représentation dans l'espace d'état de ce multimodèles est donnée par :

$$x_i(t+1) = A_i x_i(t) + B_i u(t) \quad \text{(I.22)}$$

$$y_i(t) = C_i x_i(t) \quad \text{(I.23)}$$

$$y(t) = \sum_{i=1}^{L} \mu_i(\zeta(t)) y_i(t) \quad \text{(I.24)}$$

Où $x_i \in R^n$ et $y_i \in R^p$ sont respectivement le vecteur d'état et le vecteur de sortie du $i^{ème}$ sous-modèle et où $u \in R^m$et $y \in R^p$ sont respectivement le vecteur de commande et le vecteurde sortie.

I.5.3.Modèle à base de réseaux de neurones :

Initialement étudiés en vue de modéliser le comportement du cerveau humain, les modèles à base de réseaux de neurones sont aujourd'hui des outils de calcul mathématique sophistiqué utilisés dans des domaines très divers. Un modèle neuronal est constitué de plusieurs unités de calcul élémentaires (les neurones artificiels), fonctionnant en parallèle. Chaque neurone reçoit des informations (qui peuvent être les entrées du modèle ou les sorties d'autres neurones), les traite, et envoie le résultat du traitement vers d'autres neurones. Le ou les neurones de sorties permet de reproduire le comportement du système à modéliser. Ces types de réseaux sont capables de représenter des systèmes très

complexes. Il existe différents types de réseaux de neurones artificiels dont les Perceptrons Multi-Couches, et les Réseaux de Fonctions à Base Radiale.

I.6. CHOIX DES CRITERES D'ESTIMATION PARAMETRIQUE ET DE SELECTION DE MODELE :

La fonction non linéaire *F(·)* définie par l'équation (I.6) F(·) peut être un modèle flou, un modèle neuronal, etc. Quelle que soit la structure adoptée pour la fonction *F(·)*,on aboutit toujours à une famille de fonctions paramétrées par θ. Une procédure d'identification paramétrique doit permettre d'estimer θ de façon à ce qu'un critère de

performance *J* (ou fonction de coût) du modèle soit optimal. Cependant, dans la mesure où plusieurs possibilités existent pour représenter un seul et même système, il est nécessaire de disposer d'un outil de comparaison, c'est-à-dire, d'un critère de sélection de modèles. Cet outil, basé sur des critères de performances à atteindre, permet de sélectionner le meilleur modèle.

Par ailleurs, le choix de la structure de modèle s'accompagne de la détermination de sa topologie interne (par exemple, nombre de règles pour un modèle flou, nombre de neurones cachés pour un modèle neuronal). L'utilisation d'un critère de sélection de modèles permet aussi de déterminer la meilleure topologie pour une structure de modèle donnée.

I.6.1. Choix du critère d'estimation paramétrique :

L'estimation des paramètres d'un modèle se fait en optimisant un critère de performance dans le but d'approcher la sortie du système par celle du modèle. Ce critère de performance est une fonction de l'écart (ou résidu) ε entre la sortie réelle du système et celle du modèle :

$$\varepsilon(t) = y_s(t) - \hat{y}(t) \tag{I.25}$$

Le critère quadratique est le plus utilisé. Il s'exprime par :

$$J = \frac{1}{2}\sum_{t=1}^{k} \varepsilon(t)^2 \tag{I.26}$$

Où *k* désigne le nombre d'observations considérées.
Dans certains cas, si la variance du bruit varie fortement pour certaines observations, il est nécessaire d'utiliser le critère quadratique pondéré défini par :

$$J = \frac{1}{2}\sum_{t=1}^{k} \omega(t)\varepsilon(t)^2 \tag{I.27}$$

Où ω(t) est la fonction de pondération affectée au résidu ε(t), de façon à réduire l'influence des points situés dans les zones de forte variance.

I.6.2. Choix du critère de sélection de modèle :

La structure du modèle détermine sa complexité en termes de nombre de paramètres nécessaires pour décrire le système et de procédure d'estimation de ces paramètres.

L'objectif est d'obtenir un modèle qui soit, d'une part, aussi simple que possible comportant un minimum de paramètres, et d'autre part, aussi précis que possible en disposant d'une bonne capacité de généralisation.

L'estimation de la capacité de généralisation d'un modèle nécessite la disposition d'une base de données de test n'ayant pas été utilisée dans la phase d'apprentissage. Cela réduit la base d'apprentissage et peut donc conduire à des pertes d'informations. Des méthodes de sélection de modèles basées sur la parcimonie (moins de paramètres et plus de précision) ont été la borées [14][15][16].

La méthode de validation croisée consiste à diviser la base d'apprentissage en *k* sous ensembles de tailles identiques et à procéder à *k* étapes d'identification-validation. A chaque étape, *k−1* sous ensembles sont utilisés pour l'identification et le dernier sous ensemble pour la validation. L'erreur quadratique moyenne de généralisation (Mean Squared Generalization Error - MSGE) est ensuite évaluée par :

$$MSGE = \frac{1}{k}\sum_{j=1}^{k}\sum_{t=1}^{N_j}(y_s(t) - y_j(t))^2 \tag{I.28}$$

Où $y_j(\cdot)$ est le modèle identifié à l'itération *j* et N_j la taille du sous ensemble de validation à l'itération *j*.

La structure de modèle permettant d'avoir la plus petite MSGE est sélectionnée. Parmi les différents modèles de cette structure, le modèle $y_j(\cdot)$ ayant la plus petite erreur sur les données de test est choisi.

I.7. ESTIMATION DES PARAMETRES DE MODELES :

L'identification paramétrique résulte de la minimisation du critère *J* défini par la relation (I.26). Cependant deux cas sont à considérer :

- Premier cas : l'ensemble des données (base d'apprentissage D_N constituée de *N* données d'observation sur le système) nécessaires à l'identification est disponible et l'estimation est faite sur cet ensemble simultanément : on parle alors de méthodes globales de minimisation. L'évaluation du critère *J* se fait sur toute la base d'apprentissage D_N. Le problème d'optimisation décrit par l'équation (I.8) peut se mettre sous la forme suivante :

$$\hat{\theta} = \arg\ min_{\theta}\ (J(\theta, D_N)) \tag{I.29}$$

Où

$$J(\theta, D_N) = \frac{1}{2}\sum_{t=1}^{N} \varepsilon(t,\theta)^2 \quad \text{(I.30)}$$

$$\varepsilon(t,\theta) = y_s(t) - \hat{y}(t,\theta) \quad \text{(I.31)}$$

L'inconvénient de cette approche est la nécessité de disposer de l'ensemble des éléments du vecteur de régression $\varphi(t)$ (voir relation (I.6)) avant de pouvoir identifier le modèle. Cette démarche ne s'applique pas aux modèles récurrents qui, en dehors des données issues des mesures, font intervenir des données issues du modèle en cours d'identification. L'approche ne convient pas non plus à l'identification en temps réel de systèmes pour lesquels les données nécessaires ne sont pas entièrement disponibles.

- Deuxième cas : certaines données nécessaires à l'identification des modèles récurrents ne sont pas entièrement disponibles (par exemple sorties estimés ou erreurs de prédiction). Il convient alors de faire une estimation récursive qui permet une estimation des paramètres au fur et à mesure que les données sont disponibles. La minimisation du critère *J* se fait de proche en proche sur des sous ensembles réduits de D_N où toutes les données nécessaires sont disponibles. Cette approche est aussi adaptée à l'identification en temps réel de systèmes.

Les paragraphes suivants décrivent différentes méthodes d'estimation paramétrique des modèles non linéaires.

I.7.1.Méthodes d'estimation globale :

I.7.1.1.Optimisation linéaire :

Un modèle peut être non linéaire par rapport au vecteur de régression mais linéaire par rapport aux paramètres. Dans ce cas l'équation (I.6) peut s'écrire sous la forme :

$$\hat{y}(t+h) = F(\varphi(t),\theta) = \phi\big(\varphi(t)\big)\theta = \Phi(t)^T\theta \quad \text{(I.32)}$$

Où$\varphi(t)$ est une fonction non linéaire du vecteur de régression $\Phi(t)$ telle que :

$$\Phi(t) = \phi\big(\varphi(t)\big) \quad \text{(I.33)}$$

La valeur optimale θ_{opt} minimisant le critère (I.26) est obtenue par l'estimateur des moindres carrés :

$$\hat{\theta}_{opt} = (\Phi_I^T \Phi_I)^{-1} \Phi_I^T y_s \quad \text{(I.34)}$$

Où $\Phi_I$est la matrice d'information :

$$\Phi_I = \begin{pmatrix} \Phi(1)^T \\ \vdots \\ \Phi(N)^T \end{pmatrix} \quad \text{(I.35)}$$

$y_s$est le vecteur de sortie :

$$y_s = \begin{pmatrix} y_s(1) \\ \vdots \\ y_s(N) \end{pmatrix} \quad \text{(I.36)}$$

Dans le cas où la matrice $\Phi_I^T \Phi_I$est mal conditionnée (le rapport entre la plus grande valeur propre et la plus petite est élevé), son inversion pose problème et il n'est pas possible d'estimer θ par (I.34). On utilise dans ce cas une technique de régularisation [17][18] qui consiste à modifier le critère à minimiser en y ajoutant un terme de pénalité permettant d'améliorer le conditionnement de la matrice $\Phi_I^T \Phi_I$.

I.7.1.2.Optimisation non linéaire :

Pour les modèles non linéaires par rapport aux paramètres, l'estimation paramétrique se fait par une méthode itérative d'optimisation non linéaire. Cette technique est basée sur la recherche d'une direction de l'espace des paramètres suivant laquelle le critère $J(\theta, D_N)$ diminue, et la modification pas à pas de θ dans cette direction. L'équation de modification du vecteur des paramètres θ est de la forme générale :

$$\hat{\theta}_{k+1} = \hat{\theta}_k - \mu_k d_k \quad \text{(I.37)}$$

Où :

$\hat{\theta}_k$ est l'estimée de θ à l'itération k.

μ_k est le pas de recherche.

d_k est la direction de recherche dans l'espace des paramètres.

Il existe différentes méthodes d'optimisation selon le choix de la direction de recherche d_k. Nous présentons ici certaines de ces méthodes qui font intervenir le gradient ou le hessien du critère *J*. Pour ces méthodes, il est donc nécessaire que le critère *J* possède des dérivées premières ou des dérivées secondes selon le cas.

a. Algorithme du gradient stochastique :

L'algorithme du gradient stochastique introduit par Widrow en 1970 est une approximation de l'algorithme du gradient déterministe. L'algorithmedu gradient stochastique (Least Mean Squares (LMS)) LMS est certainement l'algorithme adaptatif le plus populaire qui existe en raison de sa simplicité. Il pilote les paramètres du modèle ajustable de prédiction à partir des informations recueillies sur le système à chaque pas d'échantillonnage [13][19]. L'objectif de l'algorithme du gradient stochastique est de minimiser un critère quadratique en termes d'erreur de prédiction.

Rappel sur la méthode du gradient :

Soit une fonctionnel quadratique $J(x)$, et on veut trouver l'optimal de $J(x)$ c'est-à-dire qu'on cherche $\hat{x}$ de telle façon que $J(x)$ est minimal. La méthode du gradient est donnée par :

$$\hat{x}(t) = \hat{x}(t-1) + \mu \left[-\frac{\partial J(x)}{\partial (x)} \right]_{x=\hat{x}(t-1)} \tag{I.38}$$

Avec μ le pas d'adaptation.

b. Algorithme de Newton stochastique :

L'algorithme de Newton stochastique est désigné pour l'identification des paramètres du processus dynamique. Il permet de minimiser l'erreur d'une manière plus efficace. Cet algorithme est utilisé dans plusieurs applications du traitement de signal.

b.1. Principe de la méthode :

La méthode de Newton est parmi les plus utilisées pour la résolution des équations. Considérons une fonction (x) , et on cherche x tel que $f(x) = 0$.
En commençant par une valeur initiale x_0 qui est assez proche de x en extrapolant la tangente en x_0 jusqu'à son intersection avec l'axe OX, on continue jusqu'à obtenir une valeur de x tel que : $f(x) = 0$.

La figure (I.6) nous donne une description graphique de la méthode de Newton.
D'après la figure suivante, on écrit:

$$tg(\alpha) = f'(x_0) = \frac{f(x_0) - f(x_1)}{x_0 - x_1} = \frac{f(x_0)}{x_0 - x_1} \tag{I.39}$$

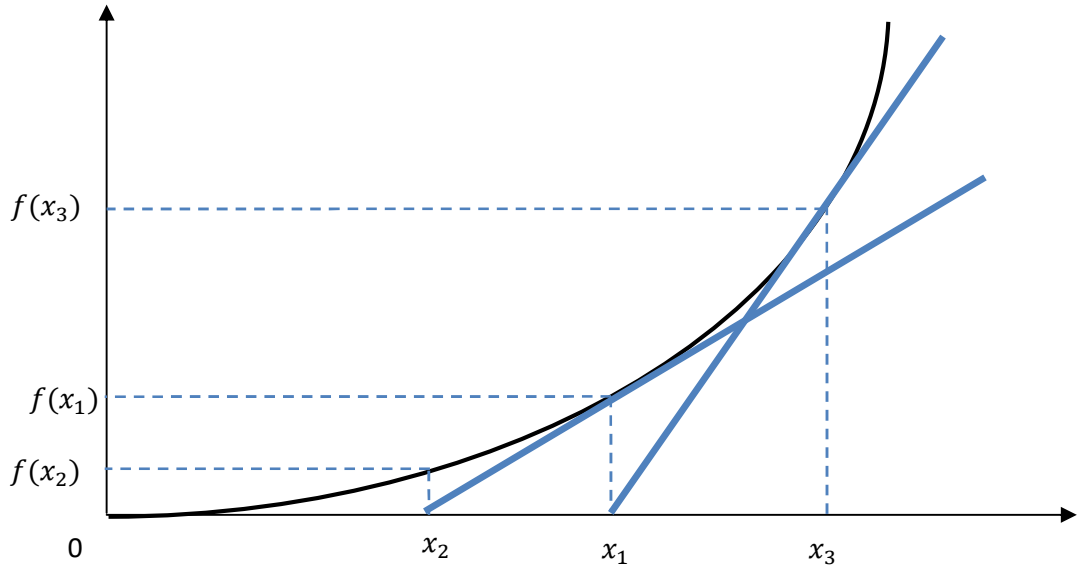


Figure I.6. Représentation graphique de la méthode de Newton.

A partir de cette dernière équation, on peut écrire les termes suivants:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}\ , x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}\ , \dots\ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{I.40}$$

D'où la formule générale s'écrira:

$$x(t) = x(t-1) - \frac{f[x(t-1)]}{f'[x(t-1)]} \tag{I.41}$$

b.2.Equations de mise en œuvre de la méthode de Newton :

Soit $J(x)$ une fonction quadratique, et on veut la minimiser, en utilisant la méthode de Newton alors on écrit :

$$MinJ(x) \Longrightarrow \left.\frac{\partial J(x)}{\partial x}\right|_{x=\hat{x}} = 0 \tag{I.42}$$

On pose

$$f(x) \Longrightarrow \left.\frac{\partial J(x)}{\partial x}\right|_{x=\hat{x}} = J'(x) \tag{I.43}$$

$$f'(x) \Longrightarrow \left.\frac{\partial^2 J(x)}{\partial x^2}\right|_{x=\hat{x}} = J''(x) \tag{I.44}$$

En remplaçant dans l'équation (I.42) on a :

$$x(t) = x(t-1) - \frac{J'[x(t-1)]}{J''[x(t-1)]} \tag{I.45}$$

Cette dernière équation porte le nom : Algorithme de Newton (cas scalaire).sous la forme vectorielle l'équation (I.45) s'écrit :

$$\hat{x}(t) = \hat{x}(t-1) - [J''[\hat{x}(t-1)]]^{-1}[J'[\hat{x}(t-1)]]^{T} \tag{I.46}$$

Où J^T représente un vecteur ligne.

c .Algorithme des moindres carrés :

La méthode des moindres carrés a été introduite par Karl Gauss en 1809. Elle est devenue la base de toutes les méthodes d'identification et d'estimation des paramètres, cette méthode est basée sur la minimisation d'une fonction quadratique J définie comme [4]:

$$J_N = \frac{1}{N}\sum_{t=1}^{N}[\varepsilon(t)]^2 \tag{I.47}$$

Où

$\varepsilon(t)$:représente l'erreur de prédiction.

N: nombre d'échantillons .

La méthode des moindres carrés est basée sur la détermination des meilleurs paramètres, c'est à dire ceux qui minimiseront un certain critère d'optimalité [20].

Le critère représente la somme des carrés des erreurs de prédiction, et qui est mentionné par l'équation (I.47).

La minimisation du critère $J_N(\theta)$ consiste à trouver un optimum, c'est à dire de calculer sa dérivée :

$$\left[\frac{\partial J_N(\theta)}{\partial \theta}\right]_{\theta=\hat{\theta}(N)} \tag{I.48}$$

$$\left[\frac{\partial J_N(\theta)}{\partial \theta}\right]_{\theta=\hat{\theta}(N)} \tag{I.49}$$

A partir de ces deux dernières équations (I.48) et (I.49), on déduit la solution optimale au sens des moindres carrés de la forme suivante :

$$\hat{\theta}(N) = [\sum_{t=1}^{N} \varphi(t)\varphi^T(t)]^{-1}.\sum_{t=1}^{N} \varphi(t)y(t) \quad \text{(I.50)}$$

Nous constatons que la matrice $(\varphi(t)\varphi^T(t))$ est de grandes dimensions, si le nombre d'échantillons N est important, le calcul de son inverse n'est pas conseillé, pour cela on utilise l'estimation par moindres carrés récursifs.

c.1.Algorithme des moindres carrés récursif :

Pour la mise en œuvre de l'algorithme récursif, on pose :

$$R(t) = \sum_{k=1}^{t} \varphi(k)\varphi^T(k) = R(t-1) + \varphi(t)\varphi^T(t) \quad \text{(I.51)}$$

D'après les équations (I.47) et (I.48), on a :

$$\hat{\theta}(t) = R^{-1}(t)\big[R(t)\hat{\theta}(t-1) - \varphi(t)\varphi^T(t)\hat{\theta}(t-1) + \varphi(t)y(t)\big] \quad \text{(I.52)}$$

$$\hat{\theta}(t) = \hat{\theta}(t-1) + R^{-1}(t)\varphi(t)\big[y(t) - \hat{\theta}^T(t-1)\varphi(t)\big] \quad \text{(I.53)}$$

$$\hat{\theta}(t) = R^{-1}(t)\sum_{k=1}^{t} \varphi(k)y(k) \quad \text{(I.54)}$$

$$\hat{\theta}(t) = R^{-1}(t)[\sum_{k=1}^{t-1} \varphi(k)y(k) + \varphi(t)y(t)] \quad \text{(I.55)}$$

$$\hat{\theta}(t) = R^{-1}(t)\big[R(t-1)\hat{\theta}(t-1) + \varphi(t)y(t)\big] \quad \text{(I.56)}$$

D'après cette dernière équation, on remarque que la solution des moindres carrés récursive contient le terme $R^{-1}(t)$ qui nécessite une inversion matricielle à chaque instant *t*.
Rappelons, le lemme d'inversion matricielle qui se présente sous la forme [21] :

$$[A + BCD]^{-1} = A^{-1} - A^{-1}B[DA^{-1}B + C^{-1}]^{-1}A^{-1} \quad \text{(I.57)}$$

Nous posons:

$$A = R(t-1), B = \varphi(t)\,, C = 1\,, D = \varphi^T(t) \tag{I.58}$$

Or

$$R^{-1}(t) = [R(t-1) + \varphi(t)\varphi^T(t)]^{-1} \tag{I.59}$$

En utilisant le lemme d'inversion matricielle [21], l'équation (I.59) peut se réécrire :

$$R^{-1}(t) = R(t-1) - \frac{R^{-1}(t-1)\varphi(t)\varphi^T(t)R^{-1}(t-1)}{1+\varphi^T(t)R^{-1}(t-1)\varphi(t)} \tag{I.60}$$

L'introduction de la matrice du gain d'adaptation $P(t) = R^{-1}(t)$, permet la mise en œuvre de l'algorithme des moindres carrés récursif (Recursive Least Squares(RLS)) de la forme suivante :

$$\hat{\theta}(t) = \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \hat{\theta}^T(t-1)\varphi(t)\right] \tag{I.61}$$

$$P(t) = P(t-1) - \frac{P(t-1)\varphi(t)\varphi^T(t)P(t-1)}{1+\varphi^T(t)P(t-1)\varphi(t)} \tag{I.62}$$

I.8.CONCLUSION :

Nous avons présenté deux notions très importantes, la modélisation et l'identification. Différents modèles non linéaires ainsi que différentes structures de représentations permettant leur implantation ont également été présentés. Les méthodes d'estimation paramétrique les plus fréquemment utilisées ont aussi été présentées.

II.1.INTRODUCTION :

Inspirés du fonctionnement du cerveau humain, les réseaux de neurones artificiels (RNA) occupent aujourd'hui une place prépondérante dans plusieurs domaines des sciences de l'ingénieur. C'est à partir de l'hypothèse que le comportement intelligent émerge de la structure et du comportement des éléments de base du cerveau, que les réseaux de neurones artificiels se sont développés. Les progrès accomplis dans la compréhension du fonctionnement du cerveau humain ont contribué dans une large mesure au développement des RNA. Cependant les scientifiques sont toujours impressionnés par l'architecture du système neuronal humain, sa complexité, son efficacité et sa rapidité dans le traitement de certains problèmes complexes que le plus puissant des ordinateurs actuels ne peut résoudre avec la même efficacité.

Ce chapitre est structuré en deux parties. Une première partie est consacrée à la présentation des réseaux de neurones artificiels. Nous commençons par donner une brève présentation de l'évolution historique de cet axe de recherche, avant de présenter le principe de fonctionnement des neurones artificiels, nous décrivons les bases essentielles des neurones biologiques ainsi que les différentes architectures de réseaux de neurones. Nous concluons cette partie en présentant les réseaux de neurones les plus utilisés. La deuxième partie de ce chapitre sera consacrée aux architectures neuronales utilisées dans notre mémoire qui sont le Perceptron Multi Couches et Fonctions Radiales de base. Nous présentons ainsi leur principe de fonctionnement avec leurs algorithmes d'apprentissage et nous terminerons par les différentes applications des RNA.

II.2.HISTORIQUE :

L'origine de l'inspiration des réseaux de neurones artificiels remonte à 1890 quand W. James, célèbre psychologue américain, introduit le concept de mémoire associative. Il propose ce qui deviendra une loi de fonctionnement pour l'apprentissage des réseaux de neurones, connu plus tard sous le nom de loi de Hebb. Il a fallu attendre l'année 1943 pour que J. Mc Culloch chercheur en neurologie et W. Pitts mathématicien proposent le premier modèle mathématique et informatique du neurone (un neurone au comportement binaire). Ce sont les premiers à montrer que des réseaux de neurones formels simples peuvent réaliser des fonctions logiques, arithmétiques et symboliques complexes. C'est avec D. Hebb, neuropsychologue canadien, qu'arrive une nouvelle vague de développement. Il

publia en 1949 son livre intitulé 'The Organization of behavior' [22] dans lequel il expose ses idées sur l'apprentissage pour la première fois. Cet ouvrage sera considéré comme source d'inspiration pour le développement des modèles informatiques d'apprentissage et des systèmes adaptatifs ultérieurs.

Un des premiers succès remarquable de cette discipline remonte à 1957 lorsque F. Rosenblatt développe le modèle du Perceptron [23]. Il construit le premier neuro ordinateur basé sur ce modèle et l'applique au domaine de la reconnaissance des formes. En 1960, B.Widrow et T.Hoff développent le modèle Adaline (Adaptative Linear Neuron). Dans sa structure, le modèle ressemble au Perceptron, cependant la loi d'apprentissage est différente. Ils proposent la minimisation des erreurs quadratiques en sortie comme algorithme d'apprentissage du réseau. Vers la fin des années 1960, M. Minsky et S. Papert montrent dans leur livre intitulé Perceptrons [24] les limitations théoriques du Perceptron. Ces limitations concernent l'impossibilité de traiter des problèmes non linéaires en utilisant ce modèle. Quelques années d'ombre se sont ensuite succédé de 1967 à 1982.

Le renouveau de cette discipline reprend en 1982 grâce à J. J. Hopfield, un physicien reconnu. Au travers d'un article court, clair et bien écrit [25], il présente une théorie du fonctionnement et des possibilités des réseaux de neurones. On peut citer encore les travaux de Kohonen et J. Anderson qui ont mis au point de nouveaux réseaux pouvant servir de mémoires associatives, ainsi que S. Grossberg qui a mis au point ce que l'on appelle aujourd'hui les réseaux auto-organisés. C'est ensuite, en 1985 que la rétropropagation du gradient apparaît. C'est un algorithme d'apprentissage adapté au Perceptron Multi Couches. Sa découverte est réalisée par trois groupes de chercheurs indépendants. Dès cette découverte, nous avons la possibilité de réaliser une fonction non linéaire d'entrée/sortie sur un réseau en décomposant cette fonction en une suite d'étapes linéairement séparables. Enfin, en 1989 Moody et Darken [26] exploitent quelques résultats de l'interpolation multi variables pour proposer le Réseau à Fonctions de base Radiales (RFR), connu sous l'appellation anglophone 'Radial Basis Function network' (RBF). Ce type de réseau se distingue des autres types de réseaux de neurones par sa représentation locale.

II.3. ELEMENTS DE BASE ET TERMINOLOGIE:

II.3.1.MODELISATION BIOLOGIQUE :

Le cerveau humain est composé de cellules distinctes appelées neurones formant un

ensemble dense d'environ 10 à 100 milliards d'unités interconnectées. Le neurone est une cellule composée d'un corps cellulaire et d'un noyau comme montré sur la figure (II.2). Le corps cellulaire se ramifie pour former ce que l'on nomme les dendrites. C'est par les dendrites que l'information est acheminée de l'extérieur vers le soma (corps du neurone). L'information est traitée alors par le corps cellulaire. Si le potentiel d'action dépasse un certain seuil, le corps cellulaire répond par un stimulus. Le signal transmis par le neurone est cheminé ensuite le long de l'axone (unique) pour être transmis aux autres neurones. La transmission entre deux neurones n'est pas directe. En fait, il existe un espace intercellulaire de quelques dizaines d'Angströms entre l'axone du neurone afférent et les dendrites du neurone efférent. La jonction entre deux neurones est appelée synapse comme schématisé sur la figure (II.1) [27].

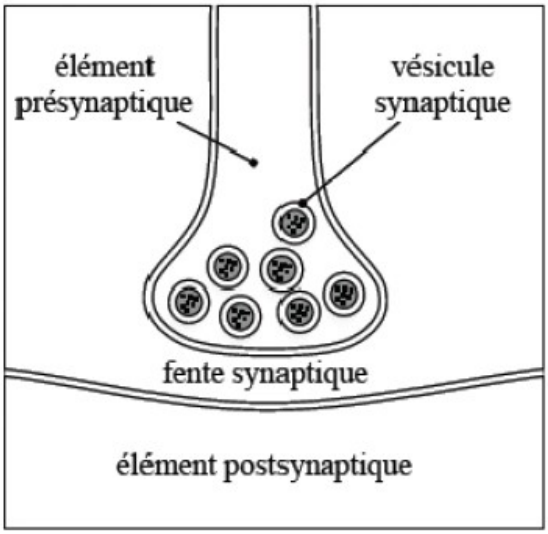


Figure II.1.Schéma d'une synapse.

Les cellules nerveuses, appelées neurones, sont les éléments de base du système nerveux centrale. Elles sont constituées de trois parties essentielles [28][29][30] : le corps cellulaire les dendrites et l'axone figure (II.2).

II.3.1.1.Le corps cellulaire :

Parfois appelé soma, il contient le noyau du neurone et effectue les transformations biochimique nécessaires à la synthèse des enzymes et d'autres molécules qui assurent la vie du neurone.

II.3.1.2.Les dendrites :

Chaque neurone possède une « chevelure » de dendrites qui entourent le corps cellulaire. Celles-ci se ramifient, ce qui les amène à former une espèce d'arborescence autour du corps cellulaire. Elles sont les récepteurs principaux du neurone pour capter les signaux qui lui parviennent.

II.3.1.3.L'axone :

L'axone qui est à proprement parler la fibre nerveuse, sert de moyen de transport pour les signaux émis par le neurone. Les neurones sont connectés les uns aux autres suivant des répartitions spatiales complexes [29]. Les connexions entre deux neurones par un petit espace synaptique de l'ordre du centième de micron.

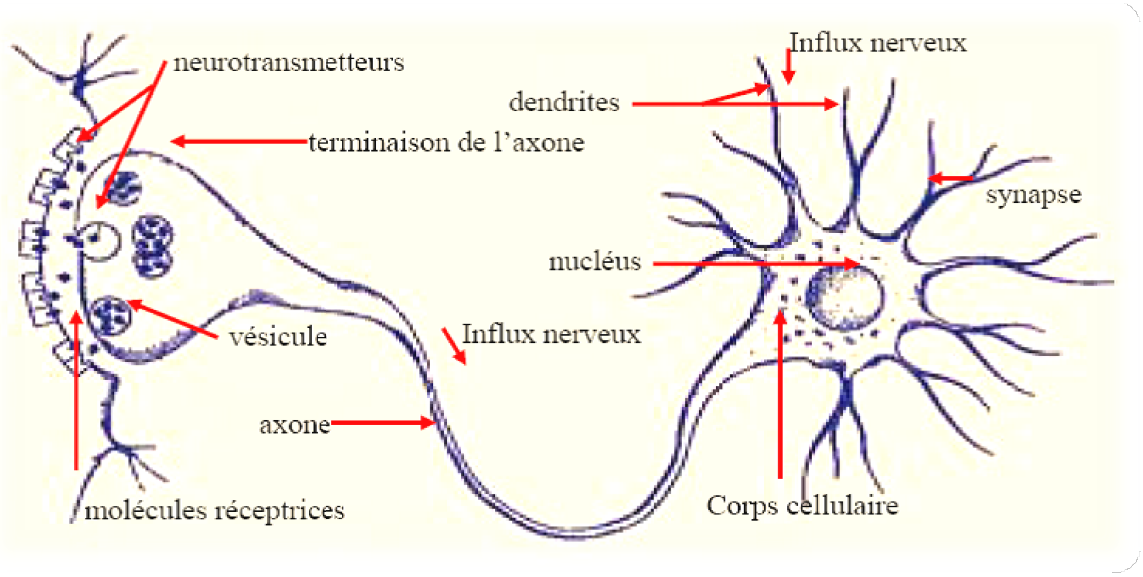


Figure II.2. Anatomie d'un neurone.

II.3.2.FONCTIONNEMENT DES NEURONES :

D'une façon générale, on peut dire que le soma du neurone traite les courants électriques qui lui proviennent de ses dendrites, et qu'il transmet le courant électrique résultant de ce traitement aux neurones auxquelles il est connecté par l'intermédiaire de son axone [29][30].

Le modèle classique présenté par les biologistes est celui d'un soma effectuant une sommation des influx nerveux transmis par ses dendrites. Si la sommation dépasse un seuil, le neurone répond par un influx nerveux ou potentiel d'action qui se propage le long

de son axone. Si la sommation est inférieure à ce seuil, le neurone reste inactif [30][31].

Les neurones peuvent engendrer les potentiels d'action dans une large gamme de fréquences, mais tous ont la même amplitude. L'information transmise est donc représentée par le nombre de potentiels d'action produits par unité de temps. Ceci correspond à un codage en fréquence des signaux transmis [29].

Lorsqu'un potentiel d'action est parvenu à l'extrémité d'un axone relié à une dendrite par une synase, il provoque à travers la membrane la libération d'un médiateur chimique au niveau de cette synapse. Ce médiateur se diffuse jusqu'à la membrane post synaptique de la dendrite ou, il provoque la naissance d'un potentiel appelé potentiel post synaptique. Ce potentiel se propage ensuite le long de la dendrite vers le soma du neurone.

II.3.3.LE NEURONE FORMEL :

Le premier modèle du neurone formel date des années quarante. Il a été présenté par Mc Culloch et Pitts. S'inspirant de leurs travaux sur les neurones biologiques ils ont proposé le modèle suivant [28][29] :

Un neurone forme lest un somme pondérée des potentiels d'action qui lui parviennent. Puis s'active suivant la valeur de cette sommation pondérée. Si cette somme dépasse un certain seuil, le neurone est activé et transmet une réponse dont la valeur est celle de son activation. Si le neurone n'est pas activé, il ne transmet rien.

On pourra résumer cette modélisation par le tableau ci-dessous (tableau II.1), qui nous permettra de voir clairement la transition entre le neurone biologique et le neurone formel.

Neurone biologique	**Neurone artificiel**
Synapse	Poids de connexion
Axone	Signal d'entrée
Dendrite	Signal de sortie
Somma	Fonction d'activation

Tableau II.1 Transition entre le neurone biologique et le neurone formel.

D'une façon plus générale, un neurone formel est un élément de traitement possédant *n*

(II.1)

entrées $x_1, x_2, \ldots, x_n$ (qui sont les entrées externes ou les sorties des autres neurones), et une seule sortie. Son traitement consiste à affecter à sa sortie y_j le résultat d'une fonction de seuillage f (dite aussi fonction d'activation) du soma pondérée.

$$y_j = f\left(\sum_{i=1}^{n} w_{ij} x_i\right)$$

Où w_{ij} est la pondération (coefficient synaptique ou poids) associée à la i^{eme} entrée du neurone j .

Parfois, il y a un terme b_j représentant le seuil interne du neurone, ce terme est considéré comme un poids w_{0j} associé à une entrée constante figure (II.3).

L'expression (II.1) devient donc :

$$y_j = f\left(\sum_{i=1}^{n} w_{ij} x_j - b_j\right) \qquad \text{(II.2)}$$

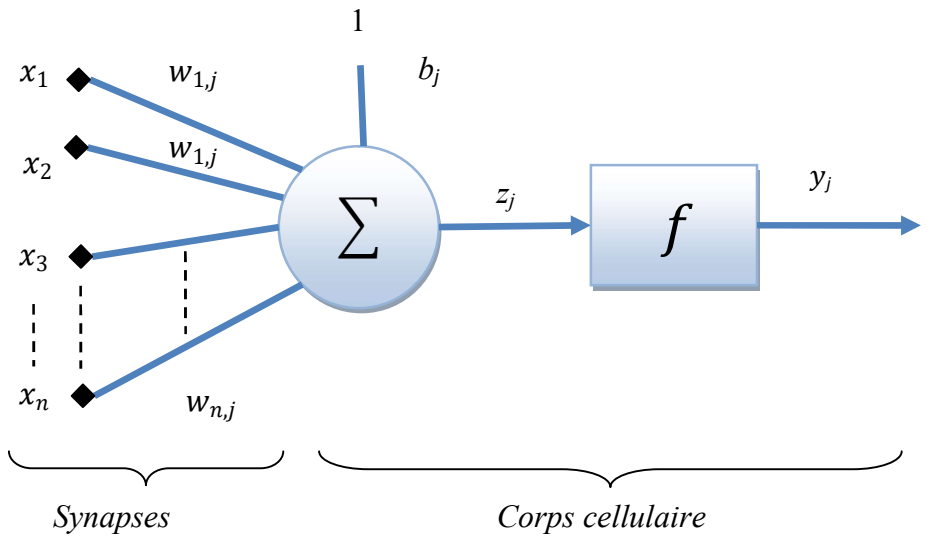


Figure II.3. Le neurone formel**.**

II.3.4.Fonction d'activation :

L'utilisation d'une fonction d'activation non linéaire permet au RNA de modéliser des équations dont la sortie n'est pas une combinaison linéaire des entrées. Cette caractéristique confère au RNA de grandes capacités de modélisation fortement appréciées pour la résolution de problèmes non linéaires. La figure (III.4) représente quelques

exemples de fonction d'activation:

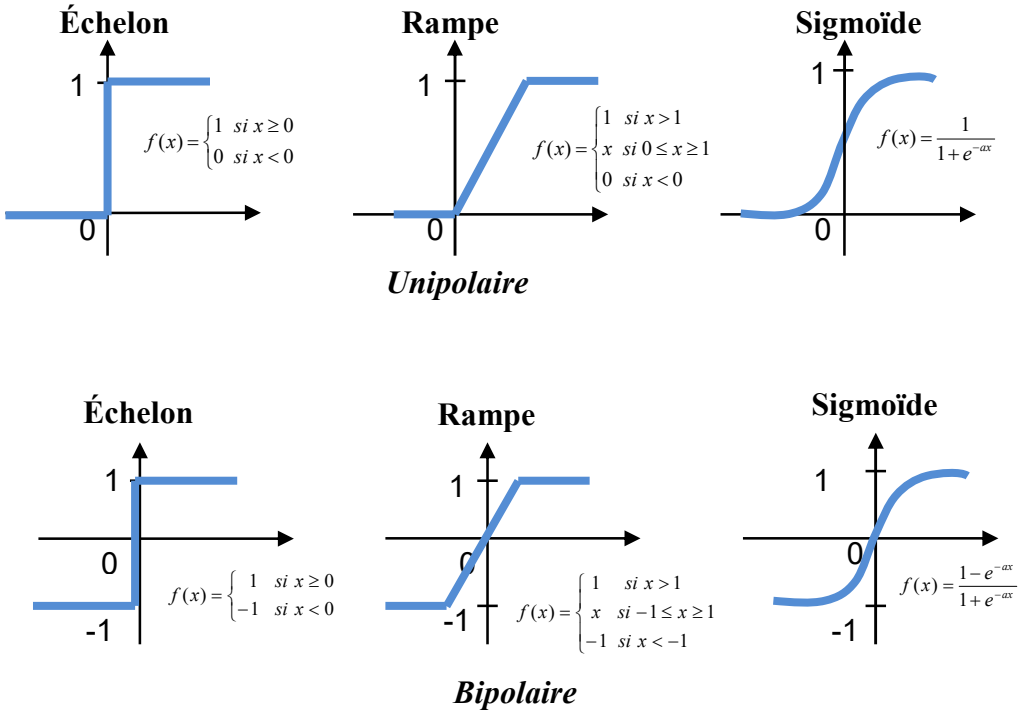


Figure II.4. Les fonctions d'activation (décision) les plus utilisées.

II.4.DEFINITION ET PROPRIETES DES RESEAUX DE NEURONES :

Un réseau de neurones est un ensemble d'éléments de traitement de l'information, avec une topologie spécifique d'interconnexions entre ces éléments et une loi d'apprentissage pour adapter les pois de connexions, il est caractérisé par un parallélisme à grain très fin et à forte connectivité. Nous entendons par la que dans un réseau de neurones donné, l'information est traitée par un grand nombre de processeurs élémentaires très simples, chacun étant relié à un nombre très important d'autres processeurs. En général le processeur très simple est un neurone formel désigné ainsi car son fonctionnement s'inspire d'une modélisation des cellules neuronales biologiques [28][30][32].

D'une manière générale un réseau de neurones possède les propriétés suivantes [29][33] :

II.4.1. Le parallélisme :

Cette notion se situe à la base de l'architecture des réseaux de neurones considérés comme ensembles d'entités élémentaires qui travaillent simultanément.

II.4.2. La capacité d'adaptation :

Celle-ci se manifeste tout d'abord dans les réseaux de neurones par la capacité d'apprentissage qui permet au réseau de tenir compte des nouvelles contraintes ou de nouvelles données du monde extérieur. De plus elle se caractérise dans certains réseaux par leur capacité d'auto-organisation qui assure leur stabilité en tant que systèmes dynamiques.

II.4.3. La mémoire distribuée :

Dans les réseaux de neurones, la mémoire d'un fait correspond à une carte d'activation des neurones. Cette carte est en quelque sortie un codage du fait mémorisé.

II.4.4. La résistance aux pannes :

A cause de l'abondance des entrées et la structure du réseau, les données bruitées ou les pannes locales dans certain nombre de ses éléments n'affectent pas ses fonctionnalités. Cette propriété résulte essentiellement du fonctionnement collectif et simultané des neurones qui le composent.

II.4.5. La généralisation :

La capacité de généralisation d'un réseau de neurones est son aptitude de donner une réponse satisfaisante à une entrée qui ne fait pas partie des exemples à partir desquels il a été adapté.

II.5. STRUCTURES DE CONNEXIONS :

II.5.1.CAS GENERAL :

Les structures qui peuvent être utilisées sont très variées. Si l'on se réfère aux études biologique du cerveau, on constante que le nombre de connexions est énorme. Par exemple, des chercheurs ont montré que le cortex était divisé en différents couches. A l'intérieur d'une même couche les interactions entre neurones sont très grandes, les neurones d'une couche sont aussi reliés aux neurones d'autres couches, le tout forme un système d'une complexité gigantesque.

D'une manière générale, l'architecture des réseaux de neurones formels peut aller d'une connectivité totale (tous les neurones sont reliés les uns aux autres), à une connectivité locale où les neurones ne sont reliés qu'à leurs proches voisins [34].

II.5.2.LES RESEAUX A COUCHES :

Dans les réseaux à couches, les neurones qui appartiennent à une même couche ne sont

pas connectés entre eux chacune des couches recevant des signaux de la couche précédente, et transmettant le résultat de ses traitements à la couche suivante [29][30]. Les couches extrêmes correspondent à la couche qui reçoit ses entrées du milieu extérieur, cette couche est appelée couche d'entrée, d'une part, et la couche qui fournit les résultats des traitements effectués, on appelle cette couche, couche de sortie, d'autre part. Les couches internes sont appelées couches cachées (c'est-à-dire non reliées au monde extérieur), leur nombre est variable. En général la couche d'entrée est une couche passive, ces neurones n'effectuent aucun traitement, leur rôle étant simplement, de recevoir et transmettre les configurations à mémoriser.

II.5.3.LES RESEAUX ENTIEREMENT CONNECTES :

Dans ce type de réseaux, chaque neurone est relié à tous les autres et passe de même un retour sur lui-même [35]. L'exemple typique de cette structure est le réseaux de Kohonen [28][36]. Ce réseau est structuré sous forme d'une matrice de neurones complètement connectées. Chaque neurone possède deux ensembles de poids :un ensemble pour calculer la somme des entrées externes pondérées, et l'autre pour contrôler les interactions entre les neurones du réseau.

II.6. L'APPRENTISSAGE:

Le besoin d'apprentissage se manifeste lorsque l'information à priori est incomplète, et son type dépend du degré de plénitude de cette information [35], Comme l'information que peut acquérir un réseau de neurones est représentée dans les poids des connexions entre les neurones, l'apprentissage consiste donc à ajuster ces poids de telle façon que le réseau présente certains comportements désirés. Mathématiquement l'apprentissage est défini par [30]:

$$\frac{d\,w}{dt} \neq 0 \qquad \text{(II.3)}$$

Où w est la matrice des poids.

On peut distinguer trois types d'apprentissage, un apprentissage supervisé, apprentissage non supervisé et apprentissage par renforcement.

II.6.1.L'apprentissage supervisé :

Ce type d'apprentissage nécessite que la réponse désirée du système à entrainer soit

connue à priori (présence d'un maître qui fournit la réponse désirée), et il est effectué de la façon suivante [37]: on présente au réseau les valeurs d'entrée et on calcule sa sortie actuelle correspondante ensuite les poids sont ajustés de façon à réduire l'écart entre la réponse désirée et celle du réseau en utilisant l'erreur de sortie (la différence entre 1a réponse du réseau et celle désirée). Cette procédure est répétée jusqu'à ce qu'un critère de performance soit satisfait. Une fois 1a procédure d'apprentissage est achevée, les coefficients synaptiques prennent des valeurs optimales au regard des configurations mémorisées et le réseau peut être opérationnel.

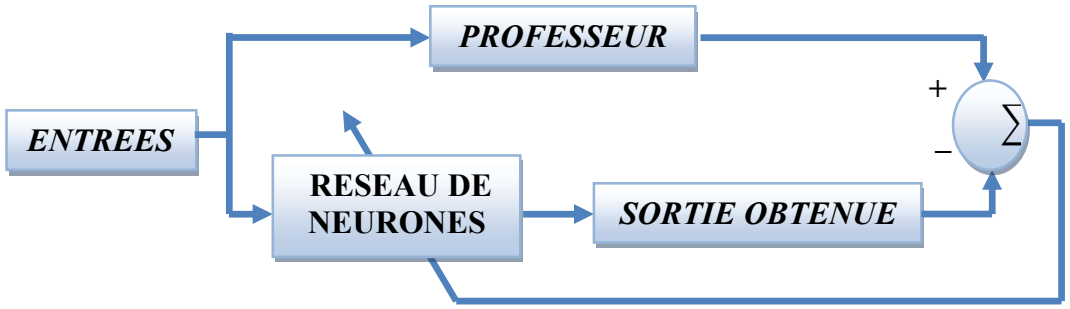


Figure II.5. Apprentissage supervisé.

II.6.2.Apprentissage non supervisé :

Dans ce cas, la connaissance à priori de la sortie désirée n'est pas nécessaire, et la procédure d'apprentissage est basée uniquement sur les valeurs d'entrées. Le réseau s'auto organise de façon à optimiser une certaine fonction de coût, sans qu'on lui fournit 1a réponse désirée [37][38]. Cette propriété est appelée auto-organisation.

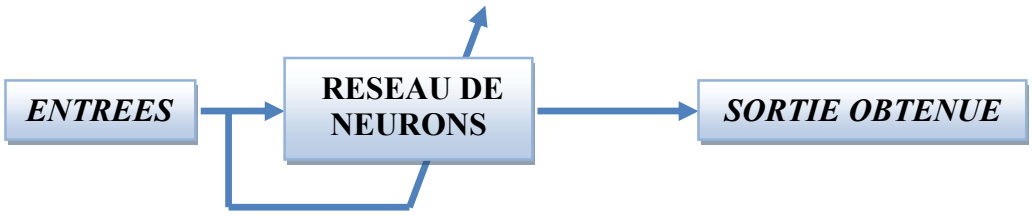


Figure II.6. Apprentissage non supervisé.

II.6.3 Apprentissage par renforcement :

L'idée de base de l'apprentissage par renforcement est inspirée des mécanismes d'apprentissage chez les animaux. Dans ce type d'apprentissage on suppose qu'il n'existe

pas de maître (superviseur) qui peut fournir la réponse correcte, mais le système à entrainer est informé, d'une manière indirecte, sur l'effet de son action choisie. Cette action est renforcée si elle conduit à une amélioration des performances du système entrainé, et les éléments qui contribuent dans La génération de cette action sont soit récompenses ou punis [39][40].

II.7.ARCHITECTURE DES RESEAUX DE NEURONES :

L'évolution d'un réseau de neurones dépend des interactions entre ces constituants, ces interactions sont définies par les poids des connexions entre les divers neurones. Selon la nature de ces connexions on distingue deux groupes des réseaux de neurones : les réseaux statiques et les réseaux dynamiques.

II.7.1. LES RESEAUX STATIQUE :

Dans les réseaux statiques, dits aussi réseaux non récurrents, les sorties dépendent des entrées actuelles, et non des entrées ou sorties passées. La sortie de n'importe quel neurone ne peut pas être appliquée directement sur son entrée ni indirectement à travers d'autres neurones. Dans ce type de réseaux, lorsqu'un stimulus est présenté en entrée du réseau, la réponse de ce dernier est calculée instantanément, après la traversée de ses couches, de l'entrée vers la sortie (dans un sens unique).

Un réseau statique réalise une transformation non linéaire de la forme [33] :

$$y = f(x) \tag{II.4}$$

Où $x \in R^n$, $y \in R^m$ sont les vecteurs d'entrée et de sortie.

Le premier modèle des réseaux statiques monocouche fut présenté par Rosenblatt en 1958, c'était le perceptron [29][33][41]. Ce modèle utilise comme élément de base neurone de Mc-Culloch dont la fonction d'activation est celle de Heaviside [28]. Le deuxième modèle est l'Adaline (Adaptive Linear Element) de B.Widrow et M.Hoff [28] ce modèle est composé d'un seul neurone de type Mc-Culloch dont la fonction d'activation est linéaire. L'association de plusieurs perceptrons (une couhe des perceptrons) avec des portes logiques forme ce qu'on appelle Madaline (Many Adalines) [28][31].

Deux règles d'apprentissage adaptées aux deux modèles précédents ont été développées. L'une d'elles, due à B.Widrow et M.Hoff est appelée aujourd'hui règle de Widrow-Hoff (ou règle Delta). Cette règle réalise une approximation de la descente du

gradient [31]. L'autre, due à F.Rosenblatt est appelée règle du perceptron [42]. Ces deux règles permettent de soumettre les réseaux monocouche à ce qu'on appelle un apprentissage supervisé.

Quoi qu'ils en soient, ces réseaux ne pouvaient résoudre que des problèmes simples de classification. Pour des problèmes complexes, une solution consiste à organiser le réseau en plusieurs couches.

II.7.1.1.RESEAUX MULTICOUCHE :

Les réseaux multicouche, appelés aussi réseaux MLP (une abréviation de multilayere perceptron, c'est-à-dire perceptron multicouche), consistent à cascader un certain nombre de perceptrons en plusieurs couches [30][33][42].

Les réseaux MLP sont organisés de la façon suivante (figure II.7) : une couche d'entrée, une ou plusieurs couches cachées et une couche de sortie. Les neurones des deux couches consécutives sont entièrement connectés. Comme la couche d'entrée est passive (elle reçoit seulement les entrées). Le nombre total des couches d'un réseau est en général donné par celui des couches cachées et de la couche de sortie. La figure (II.7) représente un réseau MLP à trois couches : deux couches cachées et une couche de sortie.

Même si les chercheurs avaient songé à des telles architectures « multicouches » celles-ci étaient restées inutilisables car on ne connaissait pas de règle d'apprentissage adaptée. En effet, l'utilisation directe des règles d'apprentissage du perceptron ou de Widrow-Hoff, pour ajuster les poids était impossible à cause de l'existence des neurones cachés. Puis une généralisation de le règle de Widrow-Hoff, pour ce type de réseaux, connue sous le nom de l'algorithme de la rétropropagation a permis de surmonter cet obstacle.

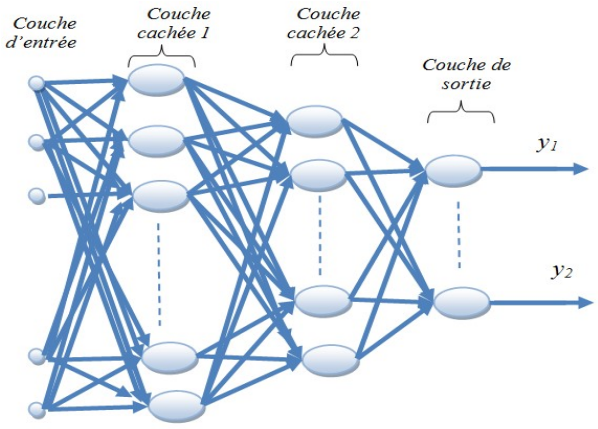


Figure II.7.Structure d'un réseau MLP.

a.Equation du réseau :

Les états des différentes neurones dans un réseau multicouches à *L* couches (couches cachées et couches de sortie) ayant *n* entrées et *m* sorties sont données par les équations suivantes [43]:

$$y_i^k(t) = f(Z_i^k(t)) \tag{II.5}$$

$$Z_i^k(t) = \sum_{j=0}^{n_{k-1}} w_{ij}^k \cdot y_j^{k-L}(t) \tag{II.6}$$

Ou $i=1,2....n_K$

$k=1,2.....L$

n_K: nombre de neurone de la couche k .

L : nombre de couche.

$y_i^k(t)$:La sortie de neurone i de la couche k.

$Z_i^k(t)$: Potentiel somatique du neurone i de la couche k.

w_{ji}^k: cœfficient synaptique (poids) de l'entrée i de la couche k .

$$y_0^k(t) = 1 \qquad k=1,2,....L \tag{II.7}$$

$$y_I^0(t) = x_j i=1,2,.....n \tag{II.8}$$

f: fonction d'activation.

$y_i^L(t)$: sortie du réseau correspondant au neurone i.

$$y_I^L(t) = y_j\,(t) i=1,2,....m \tag{II.9}$$

Ou $x_I^0(t)$et$y_i^L(t)$ sont respectivement les entrées et les sorties du réseau.

b. Méthode de rétropropagation :

La rétropropagation est l'algorithme d'apprentissage supervisé le plus utilisé pour ajuster les poids d'un réseau MLP. L'idée simple qui est à la base de cette méthode, est l'utilisation d'une fonction dérivable (fonction sigmoïde) en remplacement de la fonction de seuil utilisée dans le modèle original de Rosenblatt.

Mathématiquement, cette méthode est basée sur l'algorithme de la descente du gradient,

et utilise simplement les règles de dérivation composée. Dans cette méthode, de même que l'on est capable de propager un signal provenant des cellules d'entrée vers la couche de sortie, on peut, en suivant le chemin inverse, rétropropager l'erreur commise en sortie vers les couches cachées d'où le nom rétropropagation.

b.1.Principe de la méthode :

L'apprentissage utilise le même principe que la règle de Widrow-Hoff. On dispose d'un ensemble d'exemples qui sont des couples (entrées-sorties). A chaque étape, un exemple est présenté en entrée du réseau. Une sortie réelle *y(t)* est calculée en utilisant les équations (II.5), (II.6). Ensuite l'erreur de sortie *e(t)* est rétropropagée dans le réseau. Ceci permet de calculer les erreurs sur chaque neurone dans les couches cachées.

Les poids des connexions sont ajustés de telle sorte à minimiser la fonction de coût donnée par :

$$J(W) = \frac{1}{2}\sum_{p=1}^{n} J_p\,(W) \tag{II.10}$$

Où *n* est le nombre des exemples d'entraînement et $J_p(W)$ est le carré de l'erreur associée au p^{eme} exemple, et qui donnée par la relation suivante :

(II.11)

Où $Y^d(X_p)$ est le vecteur de sortie désiré, $Y(X_p)$ celui du réseau et X_p le p^{eme} échantillon d'entraînement. La loi d'adaptation des poids est donnée par [28] :

$$W_{ij}^k(t) = W_{ij}^k(t-1) - \mu \sum_{p=1}^{n} \frac{dJ_p(W)}{dW_{ij}^k(t)} \tag{II.12}$$

Dans le cas d'un apprentissage récursif on utilise la loi d'adaptation suivante :

$$W_{ij}^k(t) = W_{ij}^k(t-1) - \mu \frac{dJ_p(W)}{dW_{ij}^k(t)} \tag{II.13}$$

Où μ est une constante positive appelée taux ou pas d'apprentissage, et t est l'indice des

itérations. La dérivée partielle de $J_p(W)$ par rapport à chaque poids est donnée par :

$$\frac{dJ_p(W)}{dW_{ij}^k(t)} = \frac{dJ_p(W)}{dy_j^k(t)} \cdot \frac{dy_j^k(t)}{dW_{ij}^k} \tag{II.14}$$

Où :

$$\frac{dy_j^k(t)}{dW_{ij}^k} = F'\left(\sum_{m=0}^{n_{k-1}} W_{mj}^k \, y_m^{k-1}\right) y_i^{k-1} \tag{II.15}$$

Le terme $\frac{dJ_p(W)}{dy_j^k(t)}$ représente la sensibilité de $J_p(W)$ par rapport à la sortie$y_j^k(t)$, et elle donnée par l'expression [33] :

$$\frac{dJ_p(W)}{dy_j^k} = \sum_{m=1}^{n+1} \frac{dJ_p(W)}{dy_m^{k+1}} F'\left(\sum_{q=0}^{n_{k-1}} W_{qm}^{k+1} \, y_q^{k-1}\right) W_{mj}^{k-1} \tag{II.16}$$

Ce processus peut être continu jusqu'à ce qu'on arrive à la couche de sortie. Pour cette couche la sensibilité est donnée par :

$$\frac{dJ_p(W)}{dy_j^k} = y_j^L - y_j^d \tag{II.17}$$

L'expression (II.17) est appelée erreur de sortie, et l'expression (II.16) est souvent appelée erreur des couches cachées ou erreur équivalente.

Ce processus est répété, en présentant successivement chaque exemple d'entraînement. Si pour tous les exemples l'erreur de sortie est inférieure à seuil choisi, on dit alors que le réseau a convergé [44].

c. Les difficultés des MLP :

Bien que les réseaux MLP aient prouvé leur efficacité pratique dans de nombreux problèmes, ils présentent un certain nombre des difficultés encore non résolues :

- **Architecture du réseau** : il n'existe pas de résultat théorique, ni même de règle empirique satisfaisante, qui permet de dimensionner correctement un réseau en

fonction du problème à résoudre [29]. Faut-il utiliser une ou deux couches cachées.

- **Convergence de l'algorithme de la rétropropagation :** le problème de minimisation que la méthode de la rétropropagation tente de résoudre n'est pas simple [29][45]. En effet, la surface de la fonction d'erreur présente des caractéristiques peu satisfaisantes pour effectuer une descente de gradient minima locaux, qui empêchent la convergence de l'algorithme, plateau où les pentes sont très faibles, etc.…

Dans sa version complète, l'algorithme comporte un certain nombre de paramètres, qui sont difficile à régler, comme par exemple le pas du gradient (taux d'apprentissage). Il est clair que ce dernier paramètre a une importance comme dans toute descente de gradient. S'il est très faible, la convergence risque d'être très lente. S'il est trop élevé, un problème d'oscillation peut survenir.

Pour remédier à ces problèmes, plusieurs techniques ont été proposées. Yamada et Yabuta [46] ont proposé une méthode pour ajuster la pente de la fonction sigmoïde utilisée. Ceci permet d'améliorer les performances et d'accélérer la convergence de l'algorithme de la rétropropagation. Afin de dimensionner correctement le réseau R.Azimi Sadjadi et al [47] ont développé la technique de création dynamique des neurones sur la ou les couche cachées. Dans le but d'éviter les problèmes d'oscillation, certains auteurs ont proposé de modifier la loi d'apprentissage donnée par l'expression (II.18) devient en lui ajoutant un autre terme appelé moment. La loi (II.18) devient alors :

$$W_{ij}^{k}(t) = W_{ij}^{k}(t-1) - \mu \sum_{p=1}^{n} \frac{dJ_p(W)}{dW_{ij}^{k}(t)} + \alpha\left[W_{ij}^{k}(t-1) - W_{ij}^{k}(t-2)\right] \qquad \text{(II.18)}$$

Où $0 \leq \alpha < 1$.

II.7.1.2.RESEAUX RBF (fonctions radiales de base):

Contrairement aux réseaux MLP (qui sont non linéaires par rapport aux poids des connexions), les réseaux RBF (fonctions radiales de base) sont linéaires par rapport aux paramètres (poids des connexions) à estimer. Par conséquent ces paramètres sont ajustés en utilisant les techniques d'optimisation linéaire, ceci constitue l'avantage essentiel de ces réseaux.

a. Structure d'un réseau RBF :

Un réseau RBF (figure II.8) comporte deux couches de neurones [33][48][49] une couche caché dont les neurones sont connectés à ceux de la couche d'entrée par des connexions non pondérées, et une couche de sortie dont les neurones sont connectés à ceux de la couche caché par des connexions pondérées,. Les sorties z_j des neurones cachés sont données par l'expression suivante :

$$Z_j = \phi\left(\|X - C_j\|, \rho_j\right) \quad j= 1,\ldots,n_h \tag{II.19}$$

Où ϕ est une fonction de base non linéaire et C_j son centre.

ρ_j est une constante positive associée à, et n_h est le nombre des neurones (ou des RBFs) de la couche cachée.

La réponse y_i de chaque neurone de la couche de sortie est une combinaison linéaire des Z_j . En effet y_i est donnée par :

$$y_i = \sum_{j=1}^{n_h} \eta_{ij} Z_j = \sum_{j=1}^{n_h} \eta_{ij} \phi\left(\|X - C_j\|, \rho_j\right) \qquad i=1,\ldots\ldots,m \tag{II.20}$$

Où m est le nombre des neurones de la couche de sortie et η_{ij} est le poids de la connexion entre le i^{eme} neurone de la couche de sortie et le j^{eme} neurone de la couche cachée.

Certains choix typiques des fonctions de base sont donnés par les expressions suivantes :

$$\phi(Z, 1) = Z^2 \log(Z) \tag{II.21}$$

$$\phi(Z, \rho) = (Z^2 + \rho^2)^{\frac{1}{2}} \tag{II.22}$$

$$\phi(Z, \rho) = \frac{1}{(Z^2 + \rho^2)^{\frac{1}{2}}} \tag{II.23}$$

$$\phi(Z, \rho) = \exp\left(-\frac{Z^2}{\rho^2}\right) \tag{II.24}$$

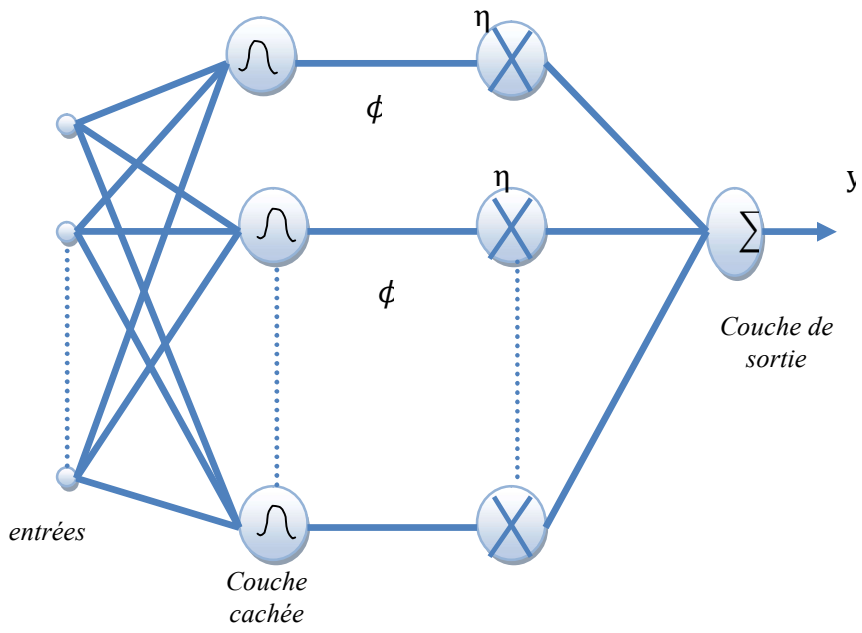


Figure II.8. Structure d'un réseau RBF.

b.L'apprentissage dans les réseaux RBF :

La plupart des algorithmes d'apprentissage des réseaux RBF procèdent à une sélection aléatoire d'un certain nombre d'échantillons de l'espace d'entrée comme étant les centres des fonctions de base. Ensuite une méthode d'optimisation linéaire est utilisée pour déterminer les poids des connexions. Cependant, dans la plupart des cas, ils évident qu'une s élection aléatoire des centres n'est pas satisfaisante. Pour remédier à cet inconvénient, S.Chen et al [49] ont proposé un autre algorithme basé sur la méthode des moindres carrés orthogonaux. Cet algorithme effectue une sélection un par un des centres à partir de l'ensemble des données, jusqu'à ce qu'un réseau adéquat soit construit.

Bien que cet algorithme permet de déterminer, d'une manière automatique, le nombre des RBFs (nombre des neurones sur la couche cachée) nécessaires au problème considéré, son utilisation dans des applications en temps réel ou dans une identification récursive est impossible, car il nécessite la disposition a priori de toutes les données entrées-sorties.

Dans d'autres algorithmes, la procédure d'apprentissage est divisée en deux étapes : un apprentissage non supervisé dans la couche cachée suivi par un apprentissage supervisé dans la couche de sortie [33]. L'algorithme de K-means et la méthode des moindres carrés sont souvent utilisées pour la sélection des centres et l'adaptation des poids des connexions, respectivement. C'est cette approche que nous avons adopté pour entraîner le

modèle neuronal lorsqu'une structure RBF est utilisée.

II.7.3. LES RESEAUX DYNAMIQUE :

Appelés aussi "réseaux récurrents", ce sont des réseaux dans lesquels il y a retour en arrière de l'information. (Figure II.9)

C'est l'architecture la plus générale pour un réseau de neurones et la plus répandue ; les réseaux bouclés possèdent un graphe cyclique tel que lorsqu'on se déplace dans le réseau suivant le sens des connexions, on peut toujours revenir vers notre point de départ.

La sortie d'un neurone du réseau peut donc être fonction d'elle-même ; la notion de retard est donc introduite.

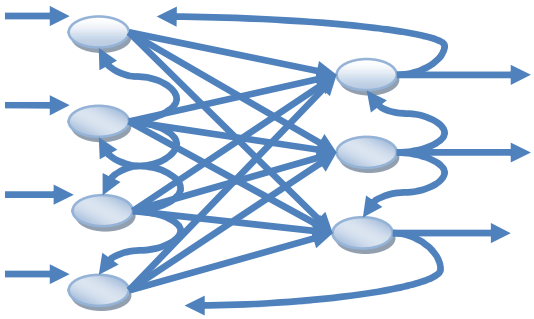


Figure II.9.Réseau à connexions récurrentes.

II.7.3.1.Les cartes auto-organisatrices de Kohonen (CAOK) :

Ce sont des réseaux à apprentissage non-supervisé qui établissent une carte discrète, ordonnée topologiquement, en fonction de patterns d'entrée. Le réseau forme ainsi une sorte de treillis dont chaque nœud est un neurone associé à un vecteur de poids. La correspondance entre chaque vecteur de poids est calculée pour chaque entrée. Par la suite, le vecteur de poids ayant la meilleure corrélation, ainsi que certains de ses voisins, vont être modifiés afin d'augmenter encore cette corrélation.

II.7.3.2.Les réseaux de Hopfield :

Les réseaux de Hopfield sont des réseaux récurrents et entièrement connectés. Dans ce type de réseau, chaque neurone est connecté à chaque autre neurone et il n'y a aucune différenciation entre les neurones d'entrée et de sortie. Ils fonctionnent comme une mémoire associative non-linéaire et sont capables de trouver un objet stocké en fonction de

représentations partielles ou bruitées. L'application principale des réseaux de Hopfield est l'entrepôt de connaissances mais aussi la résolution de problèmes d'optimisation. Le mode d'apprentissage utilisé ici est le mode non-supervisé.

II.7.3.3.Le ART :

Les réseaux ARTs ("Adaptative Resonnance Theorie") sont des réseaux à apprentissage par compétition. Le problème majeur qui se pose dans ce type de réseaux est le dilemme « stabilité/plasticité ». En effet, dans un apprentissage par compétition, rien ne garantit que les catégories formées aillent rester stables. La seule possibilité, pour assurer la stabilité, serait que le coefficient d'apprentissage tende vers zéro, mais le réseau perdrait alors sa plasticité. Les ART ont été conçus spécifiquement pour contourner ce problème. Dans ce genre de réseau, les vecteurs de poids ne seront adaptés que si l'entrée fournie est suffisamment proche, d'un prototype déjà connu par le réseau. On parlera alors de résonance. A l'inverse, si l'entrée s'éloigne trop des prototypes existants, une nouvelle catégorie va alors se créer, pour prototype, l'entrée qui a engendrée sa création. Il est à noter qu'il existe deux principaux types de réseaux ART : les ART-1 pour des entrées binaires et les ART-2 pour des entrées continues. Le mode d'apprentissage des ARTs peut être supervisé ou non.

II.8. AVANTAGES DES RESEAUX DE NEURONES :

- Capacité de représenter n'importe quelle dépendance fonctionnelle. Le réseau découvre la dépendance lui-même sans avoir besoin qu'on lui « souffle » quoi que ce soit. Pas besoin de postuler un modèle, de l'amender, etc.
- Résistance au bruit ou au manque de fiabilité des données.
- Simple à manier, beaucoup moins de travail personnel à fournir que dans l'analyse statistique classique. Aucune compétence en maths, informatique ou statistiques requises.
- Comportement moins mauvais en cas de faible quantité de données.
- Pour l'utilisateur novice, l'idée d'apprentissage est plus simple à comprendre que les complexités des statistiques multi variables [30].
- Conclusion rapide.

II.9. INCONVENIENTS DES RESEAUX DE NEURONES:

Bien sûr, les réseaux de neurones ne dispense pas de bien connaître son problème, de définir ses classes avec pertinence, de ne pas oublier de variables importantes, etc. Surtout, le réseau est une «boîte noire », qui n'explique pas ses décisions.

II.10. DOMAINES D'APPLICATION:

Les réseaux de neurones sont utilisés dans plusieurs domaines, citons la classification, la reconnaissance de formes, l'identification et la commande de processus. Le choix d'utiliser tel ou tel type de réseau de neurones dépend de l'application mais aussi des capacités de traitement du processeur sur lequel ils s'exécutent.

II.10.1.Reconnaissance des formes :

C'est un domaine privilégie d'application pour les RNA et c'est lui qui a marqué leur début. Le premier modèle dans cette thématique a été développé aux laboratoires AT&Bell dans la reconnaissance des codes postaux.

Le terme de reconnaissance est un terme général qui désigne les processus traitant des données pour en extraire des informations afin de parvenir à leur classification. Une partie non négligeable des applications neuronales actuelles appartiennent à cette catégorie, parmi lesquelles nous pouvons citer:

- Reconnaissance des formes.
- Reconnaissance des caractères.
- Classification et compression d'images.
- Diagnostic des pannes.

II.10.2.Modélisation :

L'une des applications répondu des RNA est la modélisation, nous pouvons citer comme exemples: Modélisation financière, modélisation des processus de fabrication et de production, modélisation en biomédecine.

II.10.3.Traitement de la parole :

Le traitement de la parole est une discipline scientifique localisée au croisement du traitement du signal numérique et du traitement du langage. Depuis une vingtaine d'années, les réseaux de neurones artificiels constituent une technique utilisée dans les systèmes de traitement automatique de la parole. Ils peuvent être employés à de nombreux

niveaux dans un système de traitement automatique de la parole. De nombreuses études ont été menées pour les utiliser pour le traitement du signal parole (filtrage, annulation d'échos, séparation de sources), la modélisation acoustique mais aussi pour des tâches de plus haut niveaux telles que la modélisation linguistique.

II.10.4.Traitements dépendant du temps :

Dans ce cas, on se retrouve devant des problèmes de prédiction, d'identification et de commande de processus

- Prédiction.
- Identification et commande de processus.

II.10.5.Détection d'anomalies en médecine :

Ceci est une dérivation des reconnaissances de formes. On apprend à un réseau une image du fonctionnement *normal* d'un système et celui-ci sera ainsi capable d'indiquer tout état de fonctionnement quand certains paramètres engendrent une image *anormale*. Parmi les applications en médecine nous pouvons citer:

- Cardiologie.
- Traitement des images MRI (Magnetic Resonance Imager).
- Psychologie, psychiatrie et sociologie.
- Ophtalmologie.

II.11.CONCLUSION :

Ce chapitre introduit les notions de base sur les réseaux de neurones artificiels. Après un historique sur l'apparition des RNA, et la présentation des modèles biologique et mathématique du neurone, les différentes architectures des réseaux de neurones ainsi que leurs types d'apprentissage sont présentés.

Nous avons aussi présenté deux structures des réseaux de neurones le plus utilisée pour l'identification : les réseaux MLP et les réseaux RBF. Les algorithmes d'apprentissages adaptés à ces structures ont été étudiés en détail. Nous avons terminé ce chapitre par les différentes applications des réseaux de neurones artificiels.

Le chapitre suivant traite le problème d'identification des systèmes non linéaires en utilisant les deux architectures (MLP et RBF) déjà présentées.

III.1.INTRODUCTION :

L'identification des systèmes non linéaires par réseaux de neurones a fait l'objet de nombreux travaux de recherche depuis une trentaine d'années à cause de la capacité d'apprentissage, d'approximation et de généralisation que possèdent ces réseaux [50][51]. En effet, cette nouvelle approche fournit une solution efficace à travers laquelle de larges classes des systèmes non linéaires peuvent être modélisés sans une description mathématique précise.

Dans ce chapitre, nous présentons la résolution des problèmes d'identification par les techniques neuronales en utilisant deux architectures différentes des réseaux de neurones (MLP et RBF), et de montrer l'apport de cette approche dans les problèmes d'identification des systèmes non linéaires dynamique.

III.2.ETAPES D'IDENTIFICATION :

L'identification d'un processus consiste à déterminer un modèle capable, lorsqu'il est soumis aux mêmes entrées que le processus de restituer une sortie suffisamment proche du celle du processus, au sens d'un certain critère [51]. Souvent, la construction de ce modèle est effectuée en deux étapes :

- La première étape est qualitative (caractérisation), et elle consiste à fixer les formes des équations décrivant le processus.
- La deuxième étape est quantitative (estimation des paramètres et elle consiste à déterminer les valeurs numériques des coefficients qui interviennent dans ces équation en minimisant un critère de mesure.

III.2.1. Structure d'erreur de prédiction (série-parallèle) :

La structure générale d'identification série-parallèle d'un système non linéaire est donnée par la figure (III.1) [51][52][53]. Le modèle neuronale placé en parallèle avec le système est un réseau statique dont le comportement entrée-sortie est décrit par l'expression suivante :

$$\hat{y}(k+1) = \hat{f}[x(k)] \qquad \text{(III.1)}$$

Où$x(k)$est le vecteur d'entrée des réseaux, $\hat{y}(k+1)$ sa sortie et $\hat{f}(.)$ est une approximation de $f(.)$.

L'entrée du réseau (modèle) est donnée par :

$$x(k) = [y_s(k), y_s(k-1), \cdots, y_s(k-(n+1)); u(k), \cdots, u(k-(m+1))] \quad \text{(III.2)}$$

La prise en compte de l'ordre du système s'effectue par l'intermédiaire de lignes de retard mémorisant les valeurs passées de la sortie et de la commande dont dépendra la sortie courante du système.

La différence entre la sortie y_s et la sortie $\hat{y}$ que le modèle neuronal prédit est utilisée à travers un algorithme d'apprentissage pour ajuster les poids des connexions du modèle.

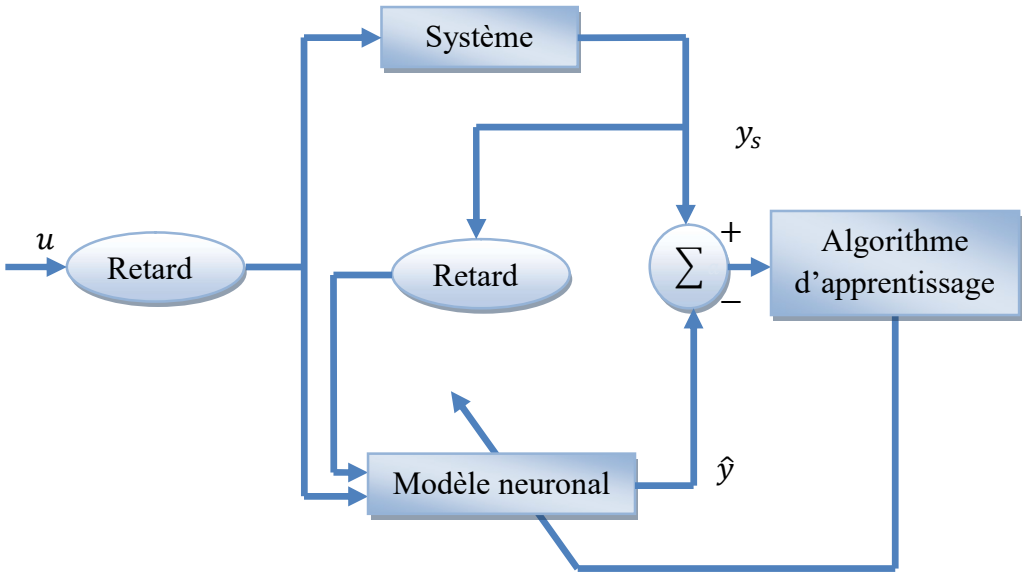


Figure.III.1. Erreur de prédiction.

III.2.2. Structure d'erreur de sortie (parallèle) :

La structure générale d'identification parallèle d'un système non linéaire est donnée par la figure (III.2) [53]. Le modèle placé en parallèle avec le système est un réseau de neurones dont la relation entrée-sortie est donnée par l'expression suivante :

$$\hat{y}(k+1) = \hat{f}[x(k)] \quad \text{(III.3)}$$

Où

$$x(k) = [\hat{y}(k), \hat{y}(k-1), \cdots, \hat{y}(k-(n-1)); u(k)u(k-1), \cdots, u(k-(m-1)] \quad \text{(III.4)}$$

$x(k)$ est le vecteur d'entrée du modèle neuronal.

Dans ce cas, la sortie courante $\hat{y}(k)$ du modèle neuronal dépend des *n* valeurs passées de sa sortie, au lieu des valeurs passées de la sortie du système dans le cas de la structure précédente.

La différence entre la sortie du système et celle du modèle est utilisée à travers un algorithme d'apprentissage pour ajuster les paramètres du modèle.

L'utilisation de cette structure d'identification souffre des problèmes de convergence et de stabilité, par conséquent l'utilisation de la structure série-parallèle est préférable [52].

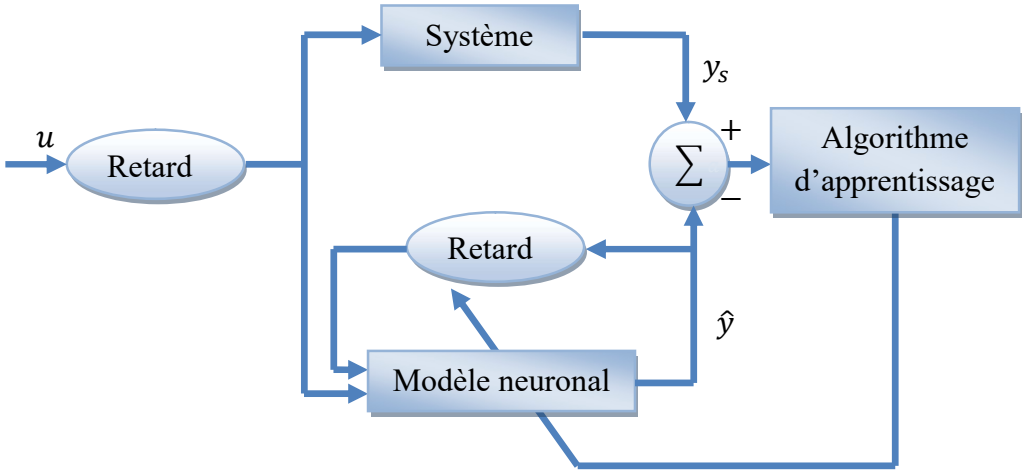


Figure.III.2. Erreur de sortie.

III.3. IDENTIFICATION DE SYSTEMES NON LINEAIRES PAR RESEAUX DE NEURONES :

L'utilisation des réseaux de neurones pour l'identification des systèmes non linéaires découle naturellement des aptitudes de ces derniers à l'approximation et la généralisation. La détermination du modèle dynamique d'un système comporte en général les étapes suivantes [54]:

- **Acquisition des données d'apprentissage et de test :**

Cette étape fournit les données entrées/sorties susceptibles de permettre l'extraction d'un modèle de procédé significatif.

- **Choix de la structure du modèle *:***

La deuxième étape consiste à choisir la structure du modèle susceptible de représenter la dynamique du système, l'architecture du réseau de neurones et ses entrées. Les réseaux multicouches statiques sont les plus utilisés à cause de la simplicité de leurs algorithmes d'apprentissage et leurs aptitudes à l'approximation et à la généralisation. Il n'existe pas de méthodes générales pour le choix du nombre de neurones sur chaque couche cachée ainsi que le nombre de ces dernières. Cependant, un réseau à une seule couche cachée est dans la majorité des cas suffisant.

➢ **Estimation des paramètres du modèle *:***

Après avoir choisi la structure du modèle, il faut estimer les paramètres de ce dernier. Ces paramètres sont les poids de connexions entres les neurones qui sont adaptés de telle sorte à minimiser un critère de performance .

➢ **Validation du modèle identifié *:***

La dernière étape doit permettre de mettre en évidence si le modèle identifié est représentatif des comportements entrées/sorties du système.

En référence à la théorie des systèmes linéaires, plusieurs modèles non linéaires ont été proposés [55][56] :

III.3.1.Le modèle NFIR (Réponse non linéaire impulsionnelle finie) :

La régression est composée uniquement des entrées passées.

$$\hat{y}(k) = f(u(k-1),...,u(k-n)) \qquad \text{(III.5)}$$

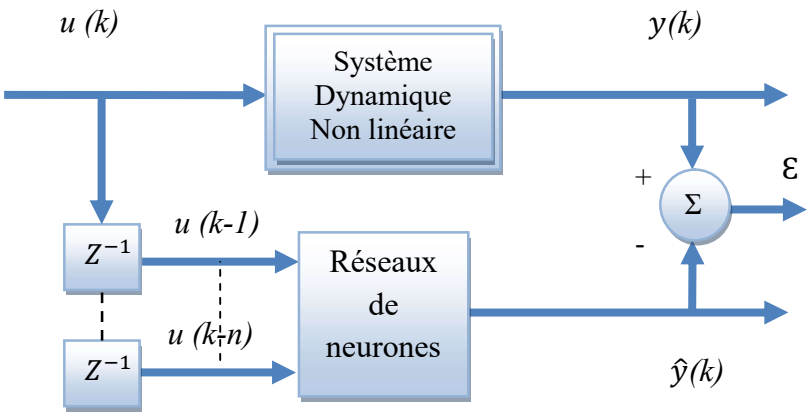


Figure.III.3. Structurelle modèle NFIR par réseaux de neurones.

III.3.2. Le modèle NARX (Non linéaire autorégressif avec entrée exogène) :

Dans ce cas la régression est composée de sorties et entrées passées.

$$\hat{y}(k) = f(u(k-1),...,u(k-n), y(k-1),..., y(k-m)) \qquad \text{(III.6)}$$

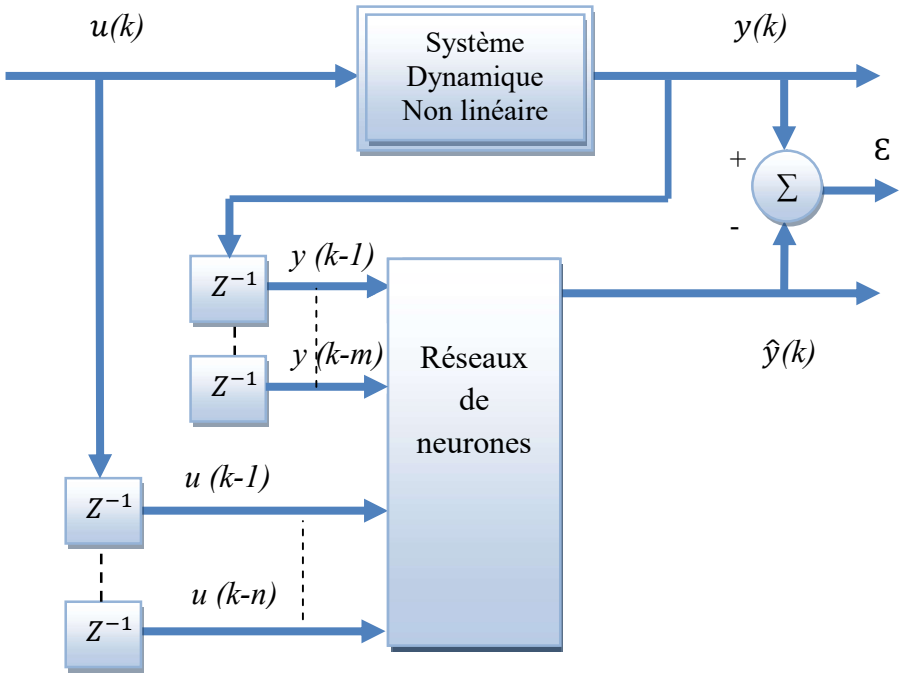


Figure.III.4. Structurelle modèle NARX par réseaux de neurones.

III.3.3.Le modèle NOE (Erreur de sortie non linéaire) :

La régression est composée d'entrées et sorties estimées passées.

$$\hat{y}(k) = f(u(k-1),...,u(k-n), \hat{y}(k-1),..., \hat{y}(k-m)) \quad \text{(III.7)}$$

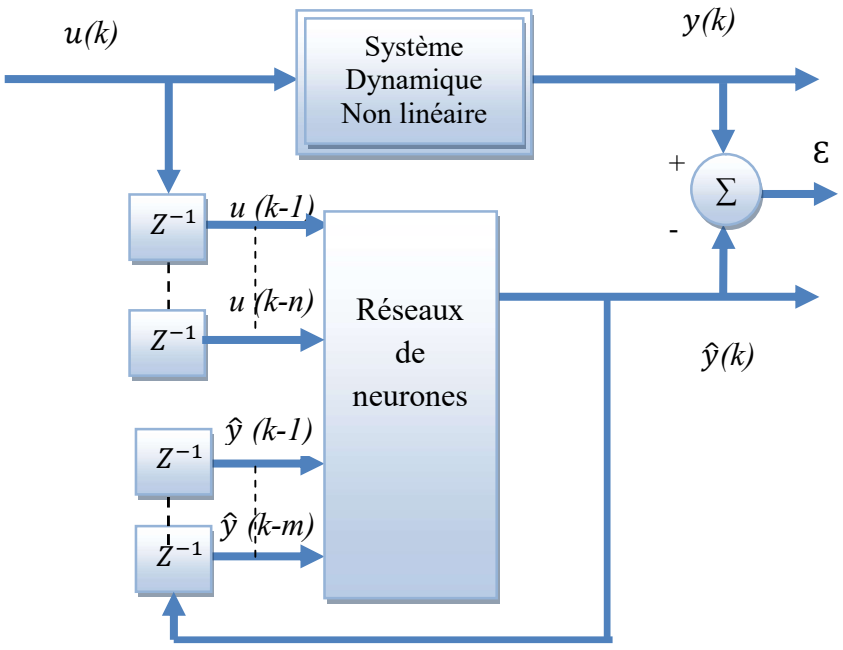


Figure.III.5. Structurelle modèle NOE par réseaux de neurones.

III.3.4.Le modèle NARMAX (Non linéaire Moyenne mobile auto régressive avec entrée exogène) :

La régression est composée de sorties et entrées passées ainsi que d'erreurs d'estimation.

$$\hat{y}(k) = f(u(k-1),...,u(k-n), y(k-1),..., y(k-m), e(k-1),..., e(k-l)) \quad \text{(III.8)}$$

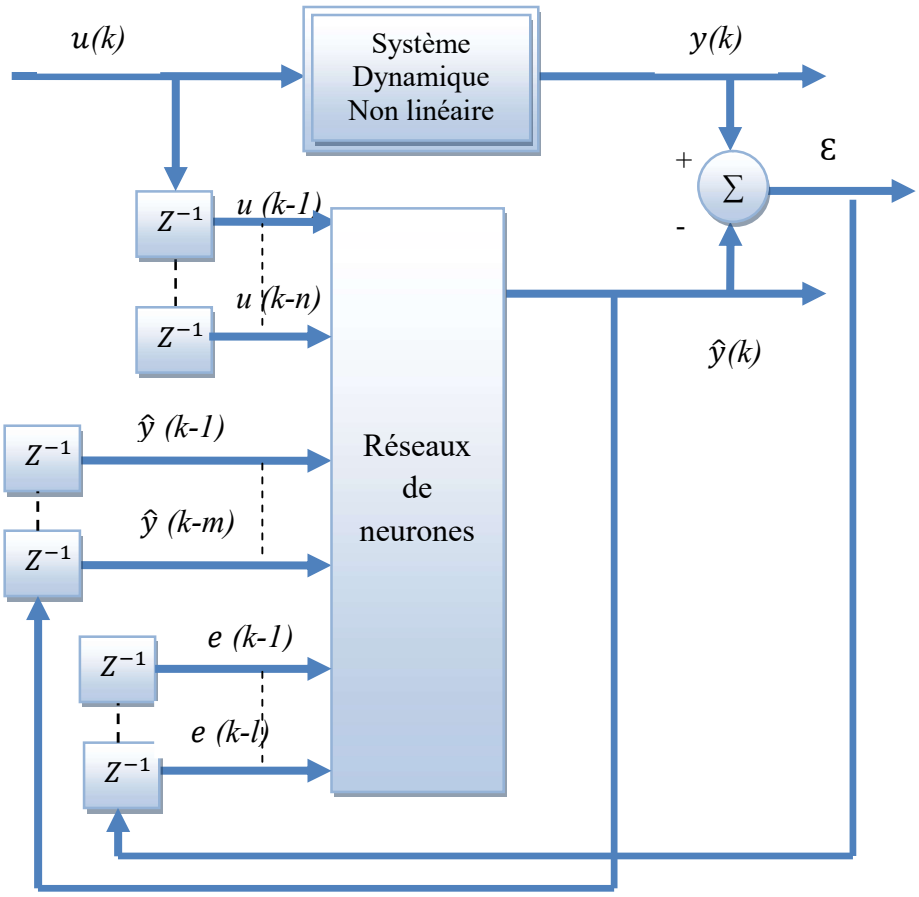


Figure.III.6. Structurelle modèle NARMX par réseaux de neurones.

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. Le choix des régresseurs est un problème combinatoire complexe. En fait, s'il existe N régresseurs possibles, alors 2^N hypothèses de modèles doivent être considérées. Plusieurs algorithmes de sélection des régresseurs sont décrits dans les références [57][58].

Après avoir choisi la régression et le modèle, il faut estimer les paramètres de ce dernier. Ces paramètres sont les poids de connexions entre les neurones qui sont adaptés de

telle sorte à minimiser un critère de performance; ceci est appelé dans la littérature des réseaux de neurones « Apprentissage ».

III.4.IDENTIFICATION PAR UN RESEAU MLP (PERCEPTRONS MULTICOUCHES) :

En général, un système dynamique est décrit par des équations différentielles s'il est à temps continu ou par des équations aux différences s'il est à temps discret. Dans la pratique, il est rare qu'un système complexe puisse être entièrement décrit par un modèle de connaissance. On a souvent recours aux modèles entrées-sortie de type « boîtes noires », pour lesquels aucune connaissance sur le système n'est nécessaire, mais des mesures sur les variables régissant le fonctionnement du système sont indispensables et en quantité suffisante. Le problème de l'identification devient alors un problème de régression non linéaire.

Dans ce chapitre , l'étude est limitée à des MLP possédant deux couches cachées avec des fonctions d'activation sigmoïde et une couche de sortie ayant un neurone à fonction d'activation linéaire, vue que cette structure a la propriété d'approximateur universel parcimonieux [59]. Cependant, la principale difficulté reste la détermination du nombre de neurones dans la couche cachée.

Les réseaux MLP utilisent un mode d'apprentissage supervisé. Dans ce mode d'apprentissage, un ensemble de données constitué des entrées du système à modéliser et des sorties correspondantes est présenté au réseau qui doit adapter ses paramètres suivant un algorithme d'apprentissage de façon à ce que la différence entre la sortie du système et celle du modèle soit suffisamment faible. Les algorithmes d'apprentissage pour les réseaux multicouches sont très nombreux l'algorithme Levenberg Marquardt qui nous avons utilisé dans ce chapitre car il est beaucoup plus rapide et plus robuste.

III.4.1.ETUDES ET SIMULATIONS :

Exemple 1 :

Le système à identifier d'ordre deux est décrit par une équation aux différences suivantes:

$$y_s(k+1) = \frac{y_s(k)y_s(k-1)(y_s(k)+2.5)}{1+{y_s}^2(k)+{y_s}^2(k-1)} + u(k)$$

Le modèle d'identification série-parallèle utilisé pour identifier le système est décrit par :

$$y_m(k+1) = \hat{f}_M[u(k), y_s(k), y_s(k-1)] \quad \text{(III.10)}$$

Où $\hat{f}_M$ est un réseau MLP à deux couches cachées, 3 neurones dans la première couche cachée et2 neurones dans la deuxième couche cachée. Un seul neurone dans la couche de la sortie et la fonction d'activation utilisée dans la couche cachée est la fonction sigmoïde. L'entrée du système et du modèle est donnée par :

$$u(k) = \sin(2\pi k\ /25) \quad \text{(III.11)}$$

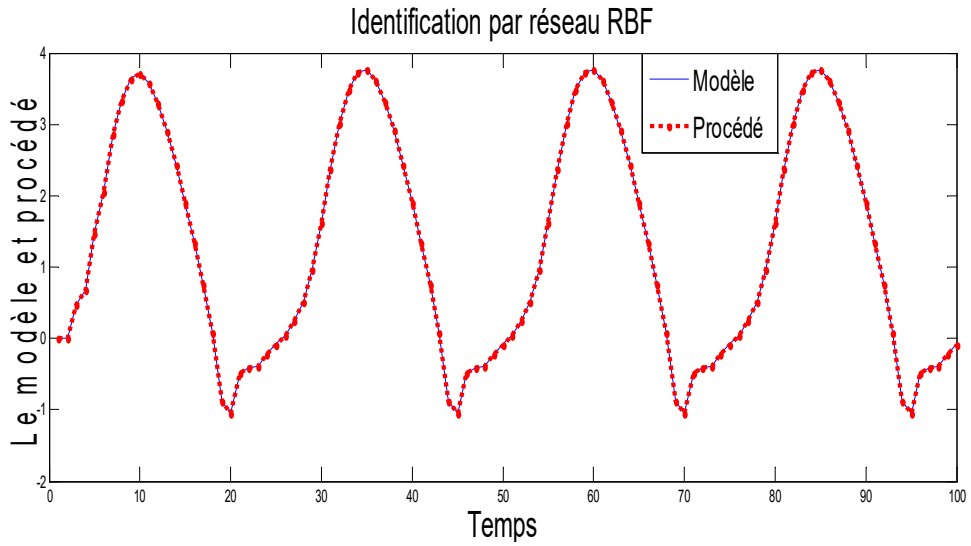


Figure III.7. Sortie du procédé et sortie du modèle.

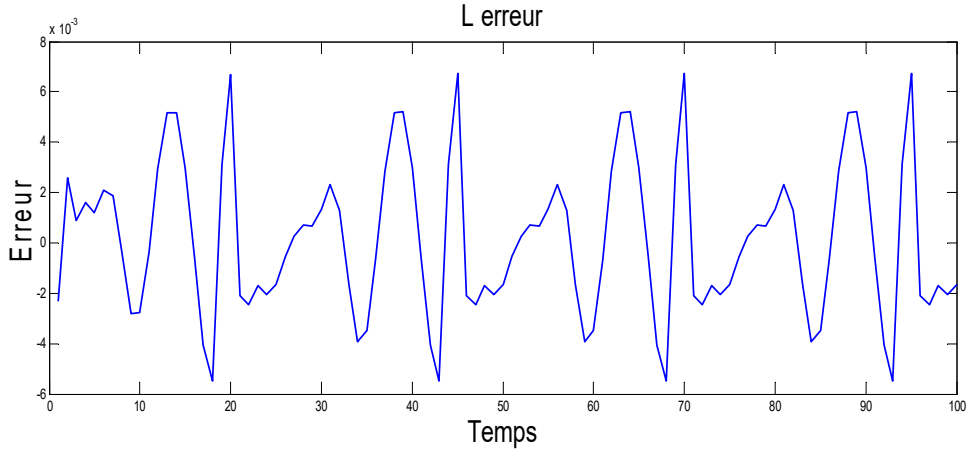


Figure III.8.Erreur d'identification par MLP.

On remarque que les deux courbes sont superposées et que l'erreur d'identification est faible.

Exemple 2 :

On a un système à identifier d'ordre un représenté par l'équation aux différences suivantes:

$$y_s(k+1) = \frac{y_s(k)}{1+{y_s}^2(k)} + u(k) \tag{III.12}$$

Le modèle d'identification série-parallèle utilisé pour identifier le système est décrit par :

$$y_m(k+1) = \hat{f}_M[u(k), y_s(k)] \tag{III.13}$$

Où $\hat{f}_M$ est un réseau MLP à deux couches cachées, 2 neurones dans la première couche cachée et2 neurones dans la deuxième couche cachée. Un seul neurone dans la couche de la sortie et la fonction d'activation utilisée dans la couche cachée est la fonction sigmoïde. L'entrée du système et du modèle est donnée par :

$$u(k) = \sin(2\pi k\ /25) + \sin(2\pi k\ /50) \tag{III.14}$$

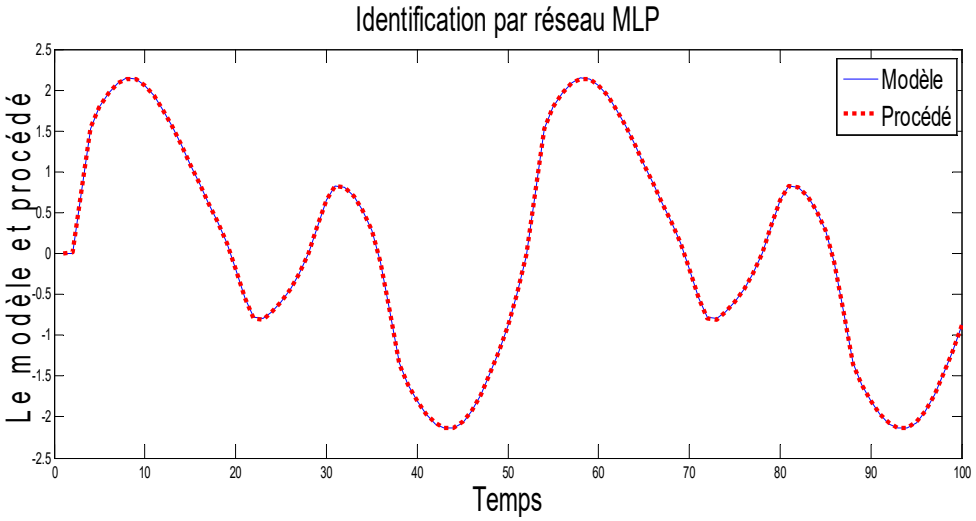


Figure III.9. Sortie du procédé et sortie du modèle.

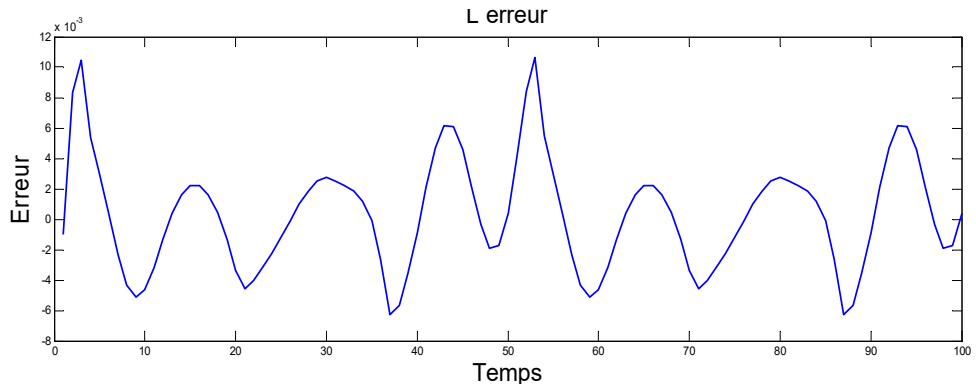


Figure III.10.Erreur d'identification par MLP.

On remarque qui l'écart entre les deux réponses est faible.

Exemple 3 :

Le système à identifier est décrit par une équation aux différences suivantes:

$$y_s(k+1) = \frac{y_s(k)}{1+{y_s}^2(k)} + u^3(k) \qquad \text{(III.15)}$$

Le modèle d'identification série-parallèle utilisé pour identifier le système est décrit par :

$$y_m(k+1) = \hat{f}_M[u(k), y_s(k), y_s(k-1)] \qquad \text{(III.16)}$$

Où $\hat{f}_M$ est un réseau MLP à deux couches cachées, 4 neurones dans la première couche cachée et3 neurones dans la deuxième couche cachée. Un seul neurone dans la couche de la sortie et la fonction d'activation utilisée dans la couche cachée est la fonction sigmoïde. L'entrée du système et du modèle est donnée par :

$$u(k) = \sin(2\pi k\ /25) + \sin(2\pi k\ /10) \qquad \text{(III.17)}$$

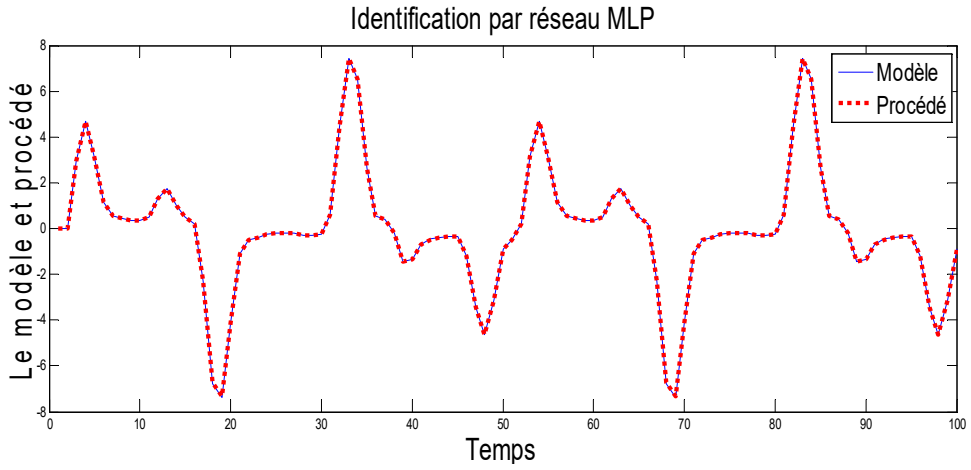


Figure III.11. Sorties du procédé et sorties du modèle.

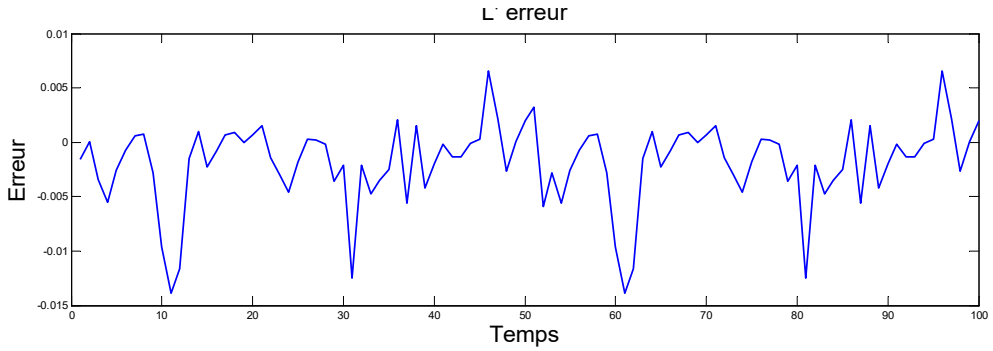


Figure III.12.Erreur d'identification par MLP.

Exemple 4 :

Le système à identifier sera donné par l'équation récurrente suivante :

$$y_s(k+1) = \frac{y_s(k)y_s(k-1)y_s(k-2)u(k-1)\big(y_s(k-2)-1\big)+u(k)}{1+y_s^2(k-1)+y_s^2(k-2)} \quad \text{(III.18)}$$

Le modèle d'identification série-parallèle utilisé pour identifier le système est décrit par :

$$y_m(k+1) = \hat{f}_M[u(k), u(k-1), y_s(k), y_s(k-1), y_s(k-2)] \quad \text{(III.19)}$$

Où $\hat{f}_M$ est un réseau MLP à deux couches cachées, 5 neurones dans la première couche cachée et4 neurones dans la deuxième couche cachée. Un seul neurone dans la couche de la sortie et la fonction d'activation utilisée dans les deux couches cachées est la fonction sigmoïde. L'entrée du système et du modèle est donnée par :

$$u(k) = \sin(2\pi k\ /250) \text{Pour} k \leq 500 \quad \text{(III.20)}$$

Et $$u(k) = 0.8\sin(2\pi k\ /250) + 0.2\sin(2\pi k\ /25) \ \text{ pour } \ k \geq 500 \quad \text{(III.21)}$$

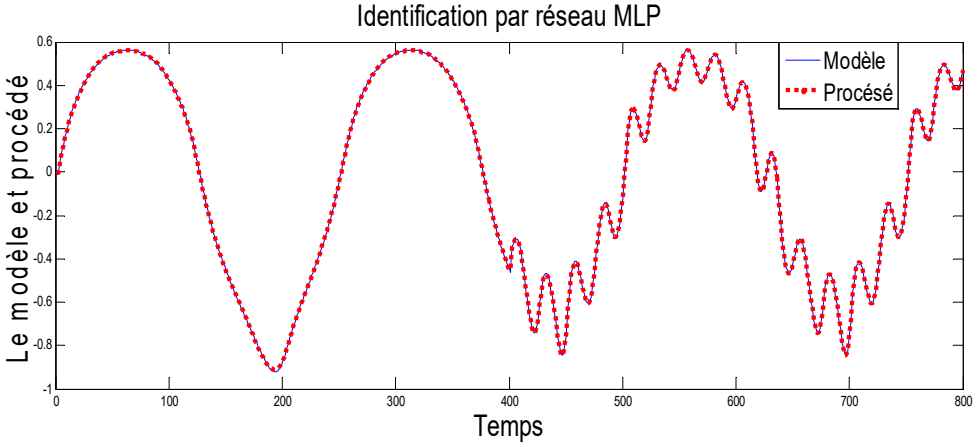


Figure III.13. Sorties du procédé et sortie du modèle.

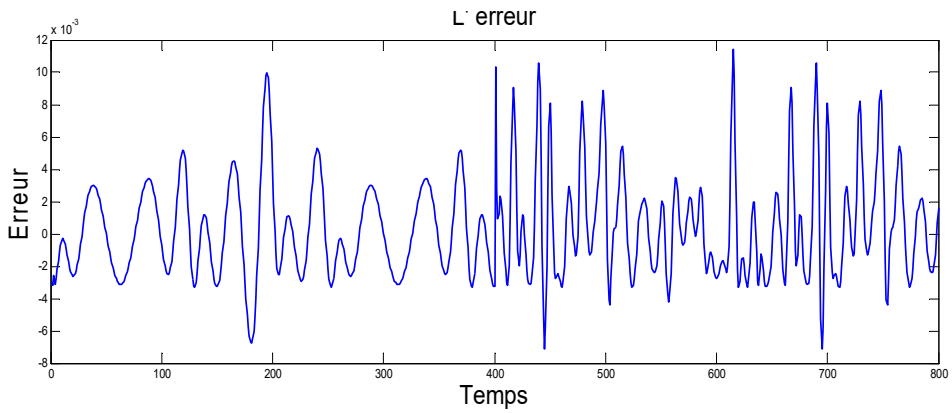


Figure III.14.Erreur d'identification par MLP.

On remarque que les deux courbes sont superposées et que l'erreur d'identification est faible malgré qu'on a changé la commande lorsque$k$>500.

Exemple 5 :

La plupart des systèmes pratiques sont de nature multivariable on a choisi un système MIMO décrit par l'équation aux différences :

$$\begin{pmatrix} y_{s1}(k+1) \\ y_{s2}(k+1) \end{pmatrix} = \begin{pmatrix} \frac{y_{s1}(k)}{1+{y_{s2}}^2(k)} + u_1(k) \\ \frac{y_{s1}(k)y_{s2}(k)}{1+{y_{s2}}^2(k)} + u_2(k) \end{pmatrix} \quad \text{(III.22)}$$

Le modèle d'identification neuronal est ainsi décrit par :

$$\begin{pmatrix} y_{m1}(k+1) \\ y_{m2}(k+1) \end{pmatrix} = \begin{pmatrix} \hat{f}_{M1}[u_1(k), y_{s1}(k), y_{s2}(k)] \\ \hat{f}_{M2}[u_2(k), y_{s1}(k), y_{s2}(k)] \end{pmatrix} \quad \text{(III.23)}$$

Où $\hat{f}_M$ est un réseau MLP à deux couches cachées, 6 neurones dans la première couche cachée et 5 neurones dans la deuxième couche cachée. Deux neurones dans la couche de la sortie et la fonction d'activation utilisée dans les deux couches cachées est la fonction sigmoïde. Les entrées du système et du modèle sont données par :

$$[u_1(k), u_2(k)] = [\sin(2\pi k\ /25)\ , \cos(2\pi k\ /25)] \quad \text{(III.24)}$$

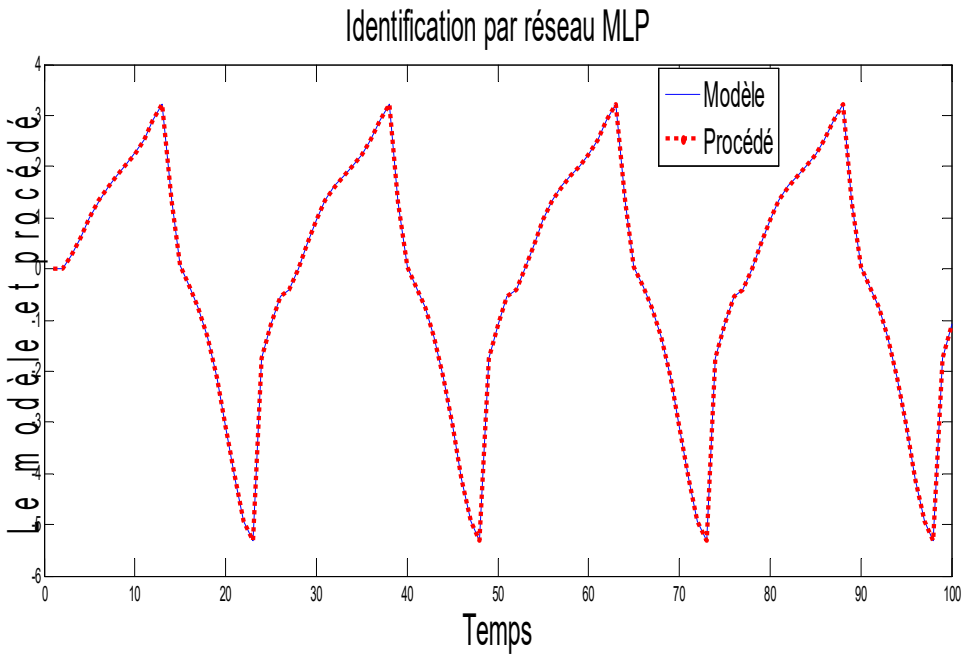


Figure III.15. Sorties du procédé et sortie du modèle (première sortie).

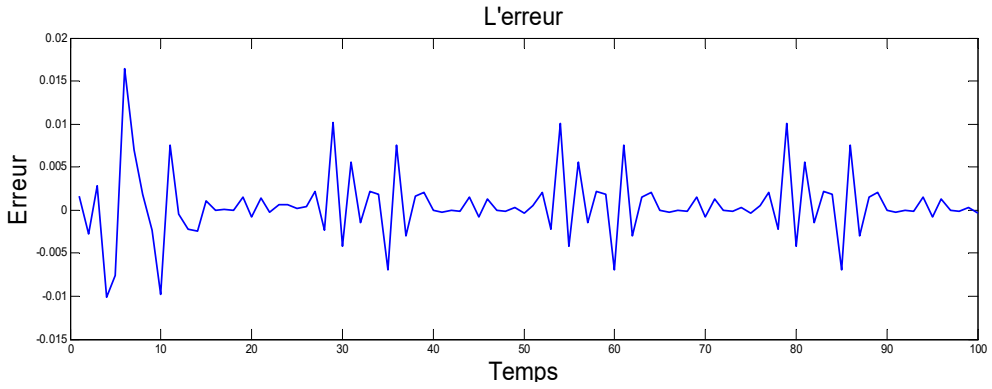


Figure III.16.Erreur d'identification par MLP.

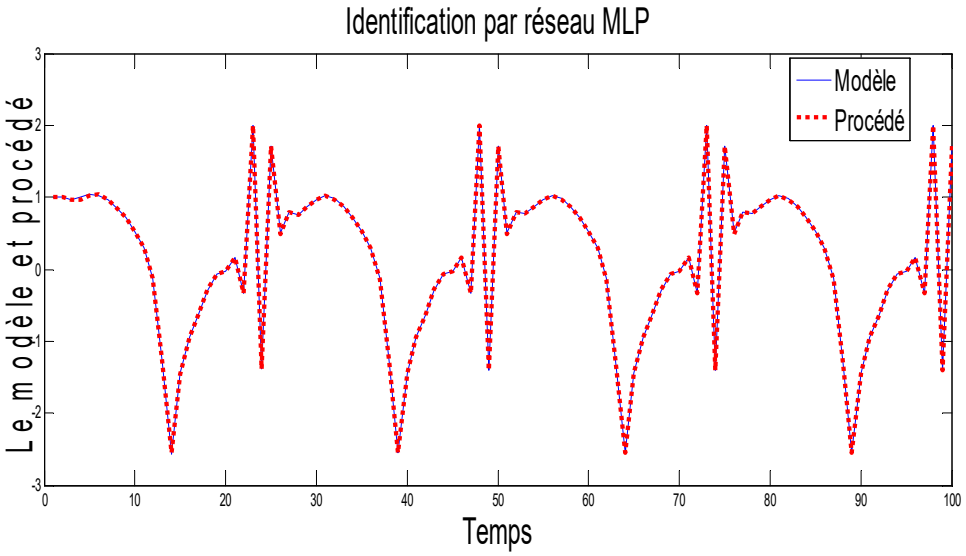


Figure III.17. Sorties du procédé et sortie du modèle (deuxième sortie).

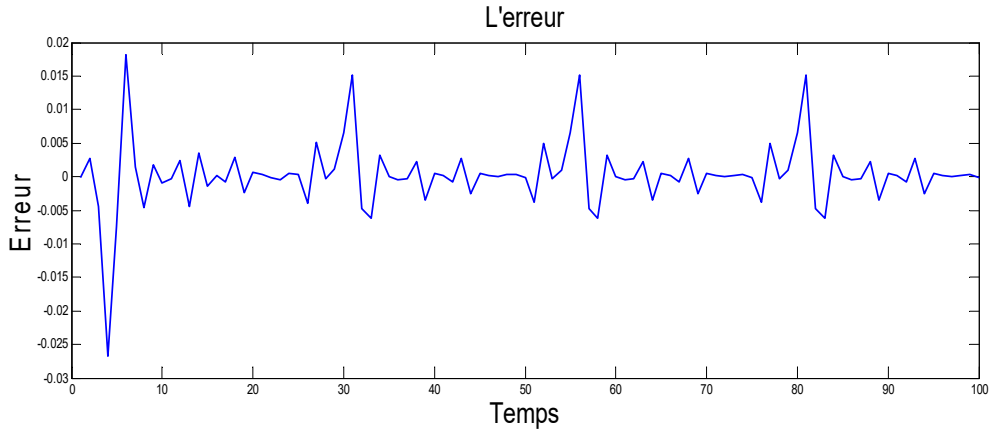


Figure III.18.Erreur d'identification par MLP.

Les résultats de simulation montrent que les réseaux MLP sont des approximateurs universels pour les systèmes non linéaires inconnus. Cependant, plusieurs difficultés

peuvent se présenter pendant la phase d'apprentissage, ceci revient au fait que les réseaux MLP sont non linéaires par rapport aux paramètres.

III.5.IDENTIFICATION PAR UN RESEAU RBF (Fonctions de base radiale) :

Vu les similitudes existant entre un réseau MLP et un réseau RBF, on s'attend une présentation similaire des capacités d'approximation et de généralisation entre ces deux réseaux. En effet ceci a été prouvé par plusieurs auteurs [51], et il a été démontré qu'un réseau RBF dont les centres et les poids des connexions sont correctement choisis est capable d'approximer avec une précision arbitraire n'importe quelle fonction continue.

En plus de leurs capacités d'approximation et de généralisation, un réseau RBF présente à sa sortie une réponse linéaire par rapport aux poids des connexions. Cette dernière propriété des réseaux RBF est d'une importance capitale, car c'est elle qui nous permet d'utiliser un algorithme d'optimisation linéaire (méthode des moindres carrés) pour ajuster les poids des connexions. L'utilisation de cette architecture peut donc fournir une solution aux problèmes soulevés par l'algorithme de la rétropropagation. L'utilisation des réseaux RBF dans les problèmes d'identification et de contrôle des systèmes non linéaires a été envisagée [52].

III.5.1.ETUDES ET SIMULATIONS :

Exemple 1 : Le système à identifier est décrit par l'équation (III.9).Le modèle d'identification utilisé est donné par l'équation suivante:

$$y_m(k+1) = \hat{f}_R[u(k), y_s(k), y_s(k-1)] \qquad \text{(III.25)}$$

Où $\hat{f}_R$ est un réseau RBF.L'entrée du système et du modèle est donnée par l'équation (III.11).

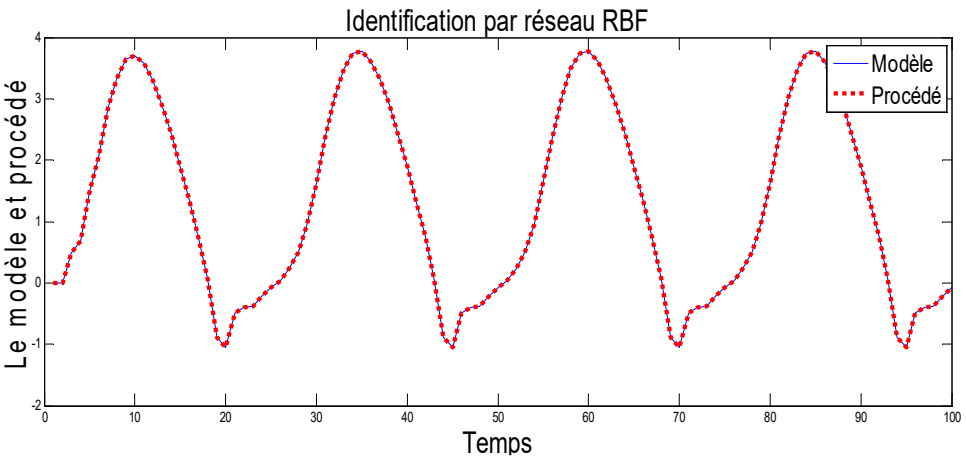


Figure III.19. Sorties du procédé et du modèle.

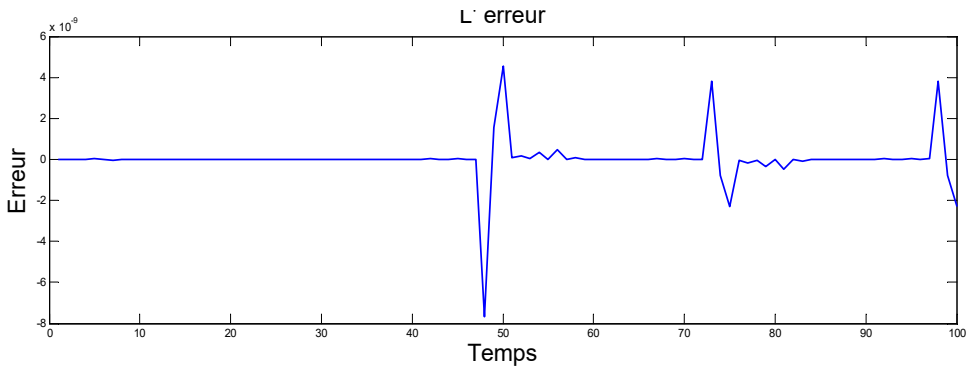


Figure III.20.Erreur d'identification par RBF.

On constate que les deux courbes sont superposées, l'erreur d'identification est très faible.

Exemple 2 :

Le système à identifier est décrit par l'équation (III.12).Le modèle d'identification utilisé est donné par l'équation suivante :

$$y_m(k+1) = \hat{f}_R[u(k), y_m(k)] \qquad \text{(III. 26)}$$

Où $\hat{f}_R$ est un réseau RBF.L'entrée du système et du modèle est donnée par l'équation (III.14).

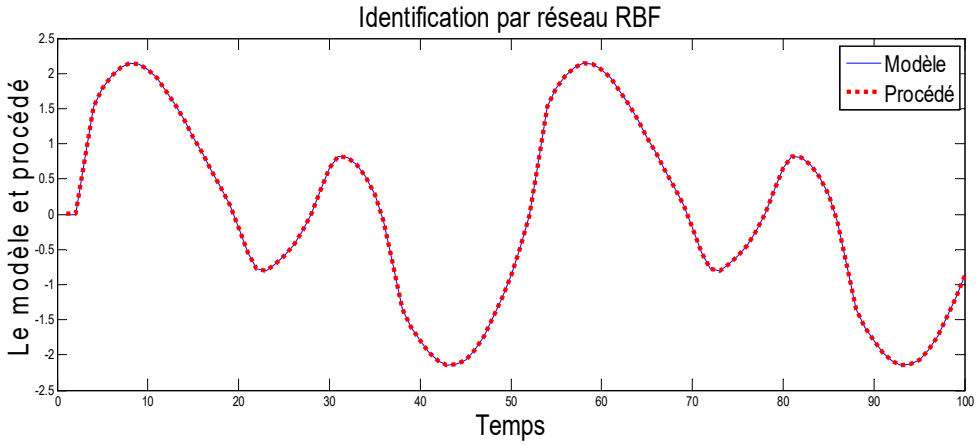


Figure III.21. Sorties du procédé et sortie du modèle.

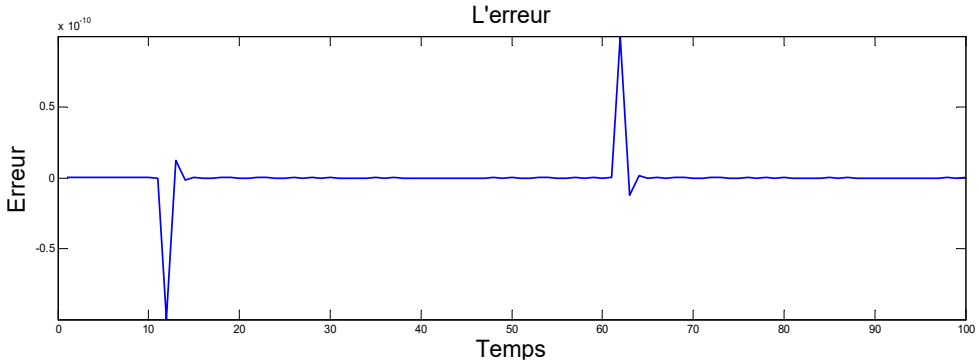


Figure III.22.Erreur d'identification par RBF.

On remarque que les deux courbes sont superposées et que l'erreur d'identification est très faible.

Exemple 3 :

Le système à identifier est décrit par l'équation (III.15). Le modèle d'identification utilisé est donné par l'équation suivante :

$$y_m(k+1) = \hat{f}_R[u(k), y_s(k)] \qquad \text{(III. 27)}$$

Où $\hat{f}_R$ est un réseau RBF. L'entrée du système et du modèle est donnée par l'équation (III.17).

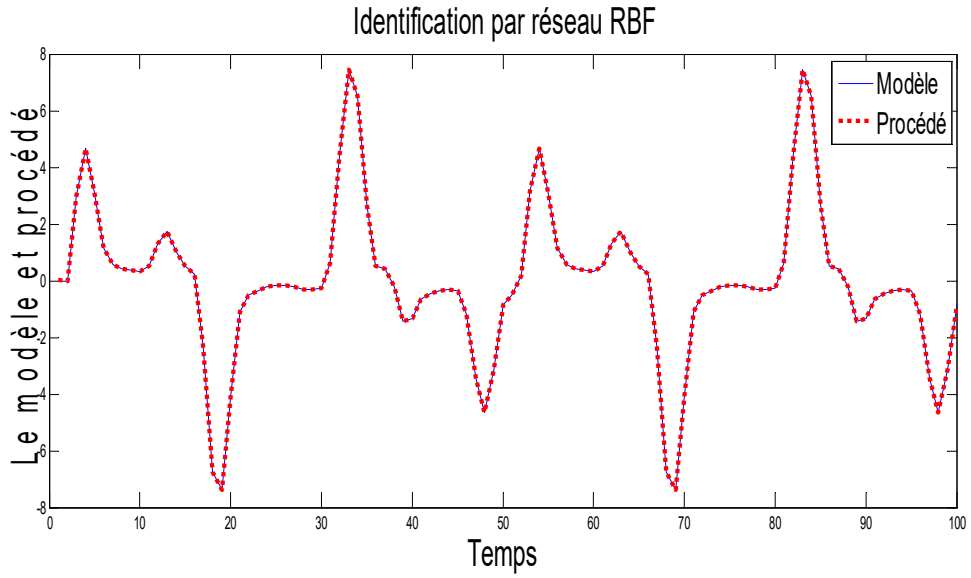


Figure III.23. Sorties du procédé et sortie du modèle.

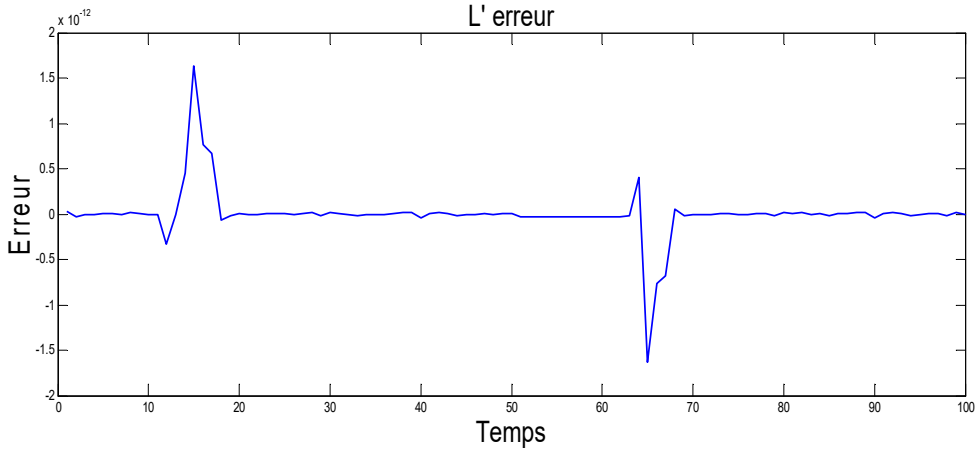


Figure III.24.Erreur d'identification par RBF.

On remarque que les deux courbes sont superposées et que l'erreur d'identification est très faible.

Exemple 4:

Le système à identifier est décrit par l'équation (III.18). Le modèle d'identification utilisé est donné par l'équation suivante :

$$y_m(k+1) = \hat{f}_R[y_s(k), y_s(k-1), y_s(k-2), u(k), u(k-1)] \qquad \text{(III.28)}$$

Où $\hat{f}_R$ est un réseau RBF. Les entrées du système et du modèle sont données par l'équation (III.20) et (III.21).

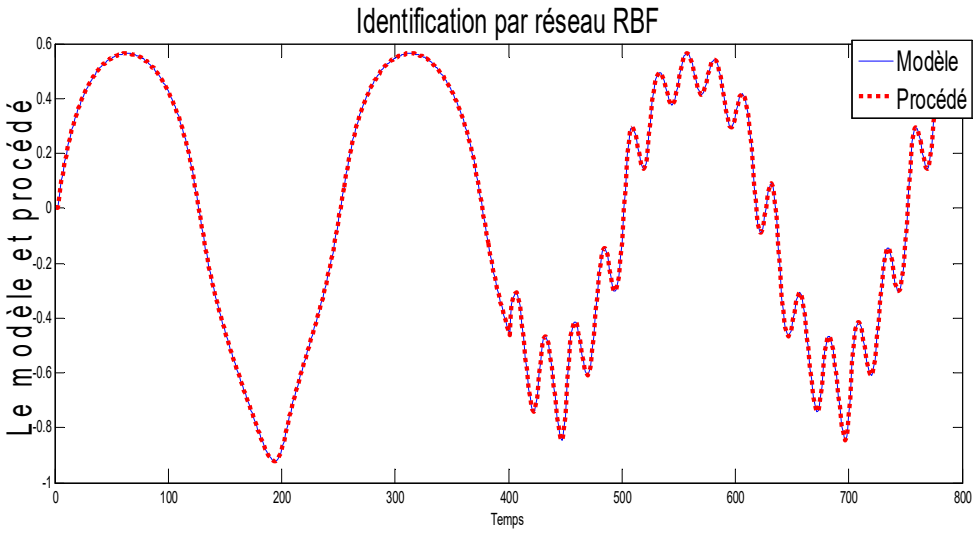


Figure III.25. Sorties du procédé et sortie du modèle.

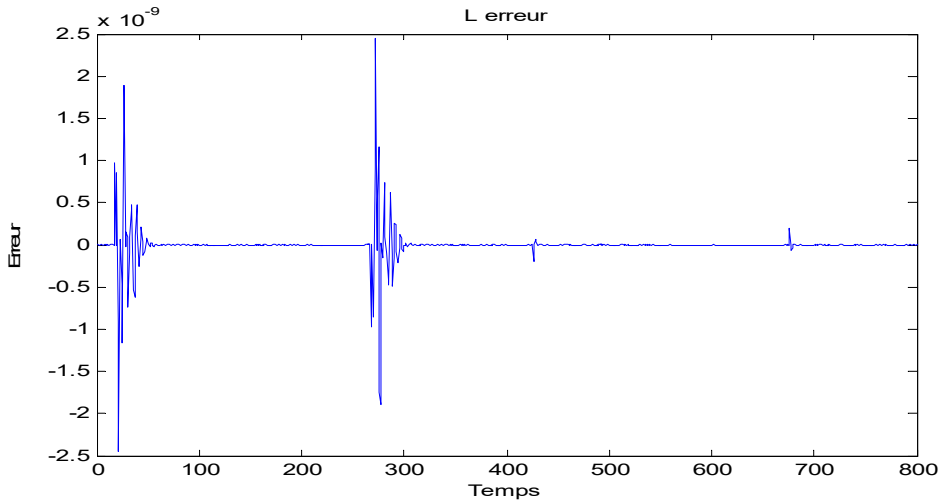


Figure III.26.Erreur d'identification par RBF.

On remarque que les deux courbes sont superposées et que l'erreur d'identification est faible.

Exemple 5:

Le système à identifier est un système multivariable décrit par l'équation (III.22). Le modèle d'identification utilisé est donné par l'équation suivante :

$$\begin{pmatrix} y_{m1}(k+1) \\ y_{m2}(k+1) \end{pmatrix} = \begin{pmatrix} \hat{f}_{R1}[u_1(k), y_{s1}(k), y_{s2}(k)] \\ \hat{f}_{R2}[u_2(k), y_{s1}(k) y_{s2}(k)] \end{pmatrix} \quad \text{(III. 29)}$$

Où $\hat{f}_R$ est un réseau RBF.Les entrées du système et du modèle sont données par l'équation (III.24).

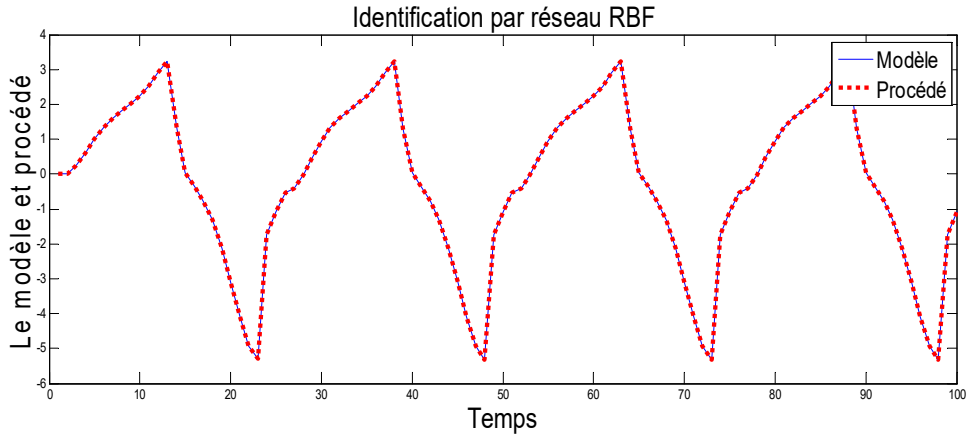


Figure III.27. Sorties du procédé et sortie du modèle (première sortie).

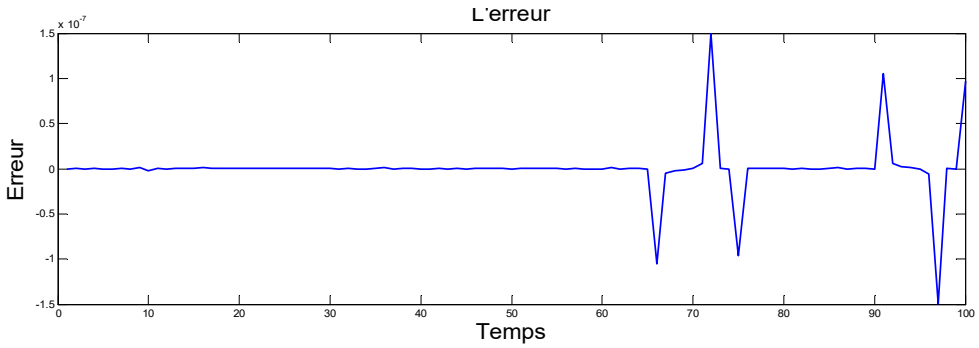


Figure III.28.Erreur d'identification par RBF.

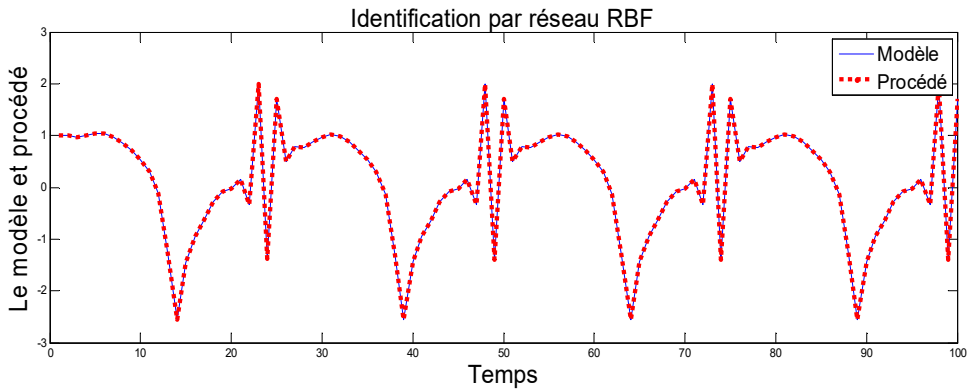


Figure III.29. Sorties du procédé et sortie du modèle (deuxième sortie).

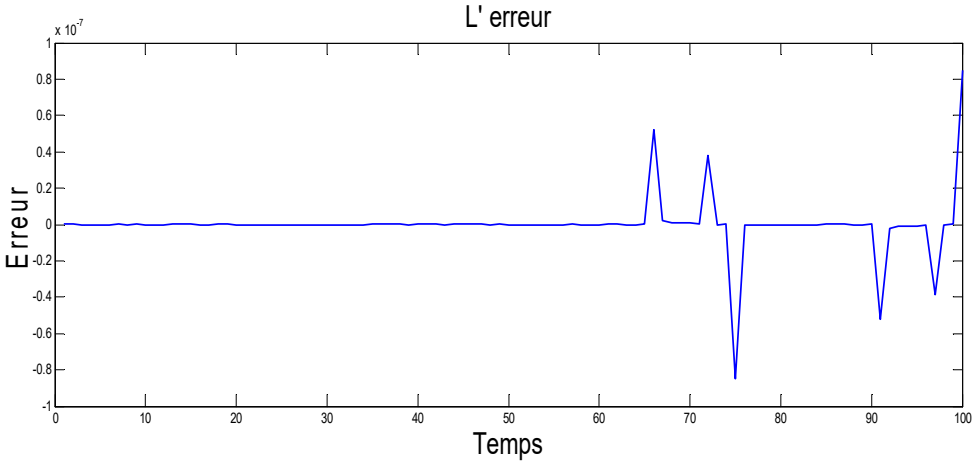


Figure III.30.Erreur d'identification par RBF.

III.5.CONCLUSION :

Dans ce chapitre nous avons présenté les structures d'identification en utilisant les réseaux de neurones, on a considéré ensuite le problème d'identification des systèmes non linéaires avec deux architectures des réseaux de neurones MLP et RBF.

Les résultats de simulation prouvent que les réseaux de MLP sont des approximateurs universels pour les systèmes non linéaires inconnus. Cependant, plusieurs difficultés peuvent surgir pendant la phase d'apprentissage ceci rejoint au fait que les réseaux de MLP sont non linéaires en ce qui concerne des paramètres. Par conséquent l'utilisation d'un algorithme d'optimisation non linéaire (rétropropagation) est nécessaire pour l'entraînement.

Les réseaux de RBF ont une réponse linéaire par rapport aux paramètres. Ceci laisse surmonter les difficultés rencontrées dans des réseaux de MLP, en utilisant des techniques d'optimisation linéaire pour leur entraînement. Les résultats de simulation prouvent que pour les systèmes relativement d'ordre réduit, l'entraînement des réseaux de RBF est beaucoup plus rapide que les réseaux MLP.

IV.1.INTRODUCTION :

Dans l'industrie, particulièrement dans les pays développés, plus de la moitié de l'énergie électrique totale produite est convertie en énergie mécanique dans les moteurs électriques. Parmi plusieurs types de moteurs électriques, les machines asynchrones triphasées occupent une place prépondérante. En effet, Au moins 90% des systèmes de commande industriels utilisent des moteurs asynchrones, qui ont, petit à petit, pris la place des machines à courant continu en raison de leurs bonnes performances : fiabilité, robustesse, faible coût et maintenance simple.

Dans ce chapitre, on va appliquer la technique d'identification par réseaux de neurones sur un moteur asynchrone (la machine asynchrone est un système dynamique non linéaire et multi-variables). Pour cela, en utilisant deux architectures différentes des réseaux de neurones (MLP et RBF).

IV.2.MODELE D'ETAT DE LA MACHINE ASYNCHRONE :

La simulation de la machine asynchrone est fondée sur l'intégration numérique d'une représentation d'état continue de la machine. Elle nécessite :

1) La détermination manuelle de l'ensemble des équations différentielles indépendantes régissant l'évolution de la machine et de son alimentation.

2) La détermination des valeurs numériques des paramètres du modèle ou le cahier des charges de la machine.

Le modèle de la machine asynchrone [60], est un modèle d'état de la forme:

$$\begin{cases} [\dot{X}] = [A].[X] + [B].[U] \\ [Y] = [C].[X] \end{cases} \quad \text{(VI.1)}$$

est composé de cinq équations non linéaires, où on a choisi le vecteur d'état de la façon suivante:

$$[X] = \left[i_{sd} i_{sq} i_{rd} i_{rq} w_r\right]^T \quad \text{(VI.2)}$$

Le vecteur d'entrée $[U]$ est donné par :

$$[U] = \left[V_{sd}\ V_{sq}\right]^T \quad \text{(VI.3)}$$

Le vecteur de sortie est donné par :

$$[Y] = [i_{sd} i_{sq} w_r]^T \tag{VI.4}$$

$$\text{Avec : } A(X) = \begin{pmatrix} \xi(-L_r.R_s.i_{sd} + M^2.P.w_r.i_{sq} + M.R_r.i_{rd} + P.L_r.M.w_r.i_{rq}) \\ \xi(-M^2.P.w_r.i_{sd} - L_r.R_s.i_{sq} - P.L_r.M.w_r.i_{rd} + M.R_r.i_{rq}) \\ \xi(M.L_s.w_r.i_{sd} + M.R_s.i_{sq} + L_s.L_rP.w_r.i_{rd} + L_r.R_r.i_{rq}) \\ \xi(P.i_s.w_r.i_{sd} + M.R_s.i_{sq} + L_s.L_rP.w_r.i_{rd} - L_r.R_r.i_{rq}) \\ \frac{1}{J}(P.M.i_{sq}.i_{rd} - P.M.i_{sd}i_{rq} - fv.w_r - C_r) \end{pmatrix} \tag{VI.5}$$

$$[B] = \begin{bmatrix} \xi L_r & 0 \\ 0 & \xi L_r \\ -\xi M & 0 \\ 0 & -\xi M \\ 0 & 0 \end{bmatrix}, \quad [C] = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{VI.6}$$

Avec : $\xi = 1/(L_r.L_s - M^2)$

$i_{sd}, i_{sq}, i_{rd}, i_{rq}$ Sont les courants de Park stator et rotor dans le repère (d, q) lié au stator.

w_r : La vitesse mécanique du moteur..

IV.3.IDENTIFICATIONDE LA MACHINE ASYNCHRONE PAR RESEAUX DE NEURONES :

Dans ce qui suit nous allons s'intéresser identification de la machine asynchrone par réseaux de neurones. La figure (IV.1) illustre le schéma de principe d'identification. Le réseau reçoit par ses entrées les trois tensions triphasées appliquées au moteur asynchrone. Il s'agit de la structure d'identification série parallèle.

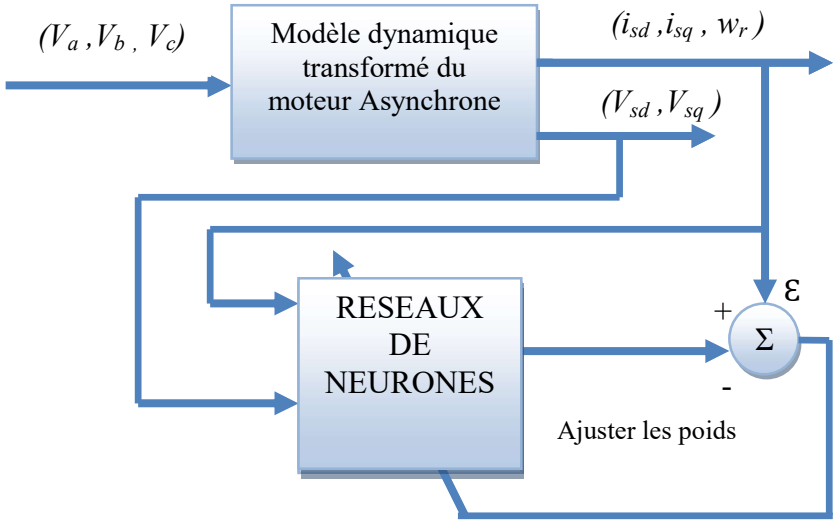


Figure. IV.1. Identification de la machine asynchrone par réseaux de neurones.

Nous allons développer le modèle neuronal utilisé pour l'identification de la machine. Le choix du modèle neuronal à apprentissage pour l'identification du modèle comportemental (comportement dynamique) de la machine dépend strictement de la complexité du modèle d'état traduisant le fonctionnement sain.

Le modèle neuronal de la machine asynchrone basé sur la structure NARX est un modèle de cinq entrées (V_{sd} , V_{sq}, i_{sd}, i_{sd}, w_r) et trois sorties (vitesse rotorique $\widehat{W}_r$ et les courants de Park stator dans le repère (d, q) ($\hat{I}_{sd}$, $\hat{I}_{sd}$)), ce modèle est donné par la figure suivante :

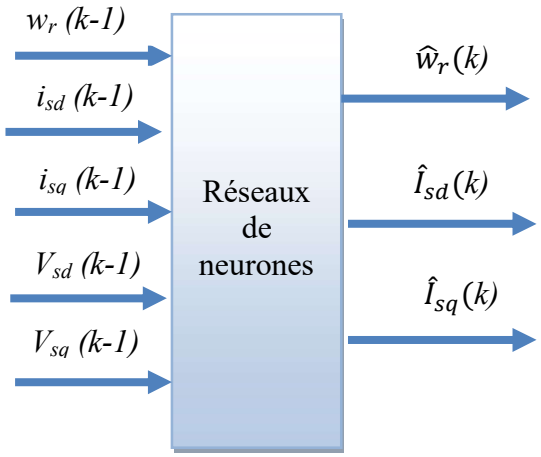


Figure.VI.2. Modèle neuronal de la machine asynchrone.

IV.3.1.Identification par perceptron multicouche (Multilayer Perceptron MLP) :

Le but de l'identification de la machine asynchrone par un réseau de neurones est de construire un modèle non linéaire, les valeurs de sorties (vitesse rotorique w_r et les courants de Park stator dans le repère (d, q) (i_{sd},i_{sq})) à partir des informations antérieures sur la vitesse, et les courants de stator.

Les sorties $\hat{w}_r$, $\hat{I}_{sd}$, $\hat{I}_{sq}$ du réseau sont définies par les relations suivantes:

$$\hat{w}_r = \psi_1(Z_1.h + b_1) \quad \text{(VI.10)}$$

$$\hat{I}_{sd} = \psi_2(Z_2.h + b_2) \quad \text{(VI.11)}$$

$$\hat{I}_{sq} = \psi_3(Z_3.h + b_3) \quad \text{(VI.12)}$$

Avec :

$$h = \varphi(w_1\hat{w}_r(k-1) + w_2\hat{I}_{sd}(k-1) + w_3\hat{I}_{sq}(k-1) + w_4V_{sd}(k-1) + w_5V_{sq}(k-1) \quad \text{(VI.13)}$$

Où : $w_i \quad i = 1,..,3$ sont les poids reliant la couche cachée à la couche d'entrée.

$Z_i \quad i = 1,..,3$ sont les poids reliant la couche cachée à la couche de sortie.

ψ est une fonction d'activation de type sigmoïde ($f(x) = \frac{1}{1+e^{-ax}}$) et φune fonction d'activation de type linéaire ($f(x) = x$). $b_i \quad i = 1,..,3$ est le biais.

Le modèle non linéaire réalisé par un réseau MLP dépend des valeurs des coefficients de pondération (poids et biais des neurones) de ce réseau. Pour qu'un réseau effectue une tâche donnée, il faut ajuster les valeurs de ses coefficients. La procédure d'ajustement des coefficients, de telle sorte que les sorties du réseau soient proches des sorties désirées, est appelée apprentissage. Un apprentissage supervisé a pour point de départ un ensemble d'apprentissage, c'est-à-dire un ensemble d'exemple, ou couples (valeurs des entrées-valeurs des sorties désirées correspondantes). Le principe général des algorithmes d'apprentissage repose sur la minimisation d'une fonction de coût quadratique des différences entre les sorties du réseau et celles désirées.

IV.3.1.1.Choix de l'architecture du réseau MLP:

Cette tâche est difficile et il n'y a aucune règle systématique pour choisir le nombre de neurones par couche cachée. Se rappeler que ce nombre est particulièrement important parce qu'il détermine la capacité du calcul de réseau. Un nombre insuffisant de neurones cachés peut compromettre la capacité du réseau de résoudre le problème. Inversement, un nombre élevé de neurones fait mémoriser le réseau le détriment de la généralisation. Ainsi nous avons procédé par la méthode d'essai et d'erreur. Nous avons commencé par des structures avec une seule couche cachée et un nombre réduit de neurones. Chaque fois, nous avons augmente graduellement le nombre de neurones jusqu'à ce qu'on obtient les performances désirée. Noter qu'il a passé plusieurs essais afin de réaliser des architectures d'architectures capables généralisé.

En appliquant ce procédé, nous pouvons étudier et évaluer plusieurs structures avec un ou deux couches cachées. On a également utilisé des réseaux de neurones avec des fonctions d'activation sigmoïdes tangentielles pour les neurones des couches cachées, et des fonctions d'activation linéaires pour la sortie du réseau.

IV.3.1.2.Choix de l'algorithme d'adaptation :

Parmi les nombreux algorithmes d'adaptation, celui de rétropropagation du gradient de l'erreur est sans doute l'algorithme connexionniste le plus utilisé dans l'apprentissage des

réseaux multicouches. Dans le logiciel de Matlab, on y trouve plusieurs algorithmes d'entraînement des réseaux multicouches. Les plus importants sont l'algorithme de rétropropagation de gradient standard, l'algorithme de rétropropagation avec momentum et l'algorithme de Levenberg-Marquardt. Les deux premiers sont basés sur la descente de gradient. Le troisième est basé sur l'approximation de Newton et il est plus puissant que les deux premiers. Il converge mieux mais demande plus de mémoire surtout lorsque la taille du réseau devient assez grande. On a constaté que les autres algorithmes sont parfois très rapides. Cependant, ils ne donnent pas des résultats acceptables.

IV.3.1. 3.SIMULATIONS :

On a utilisé des réseaux avec des fonctions d'activation sigmoïdes tangentielles pour les neurones des couches cachées, et des fonctions d'activation linéaires pour la sortie du réseau. Les poids w_i, Z_i et les biais b_i sont aléatoirement choisis par l'algorithme de Levenberg Marquardt.

D'après plusieurs choix d'architecture de réseaux MLP, nous avons trouvé que réseau à deux couches cachées ,12 neurones dans la première couche cachée et 10 neurones dans la deuxième couche cachée, donne meilleur résultat que d'autre architecteurs.

D'autres algorithmes ont été testés, mais l'algorithme Levenberg-Marquardt qu'on a utilisée car il est beaucoup plus rapide et plus robuste.

a. Démarrage à vide

Le moteur étant alimenté par un système de tensions sinusoïdales, Va, Vb et Vc Pour un démarrage à vide ($Cr = 0\ N.m$).

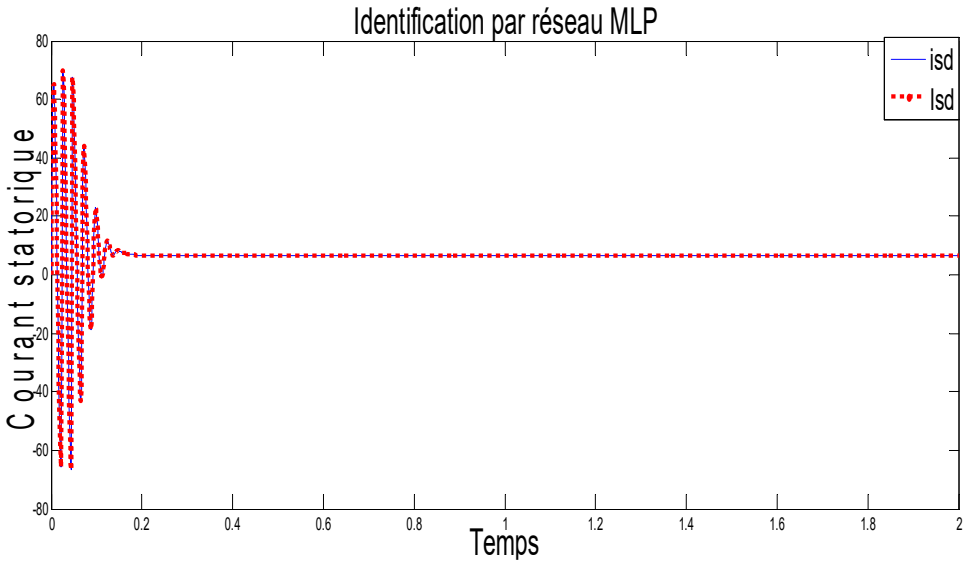


Figure.VI.3. Courant statorique suivant l'axe *d*.

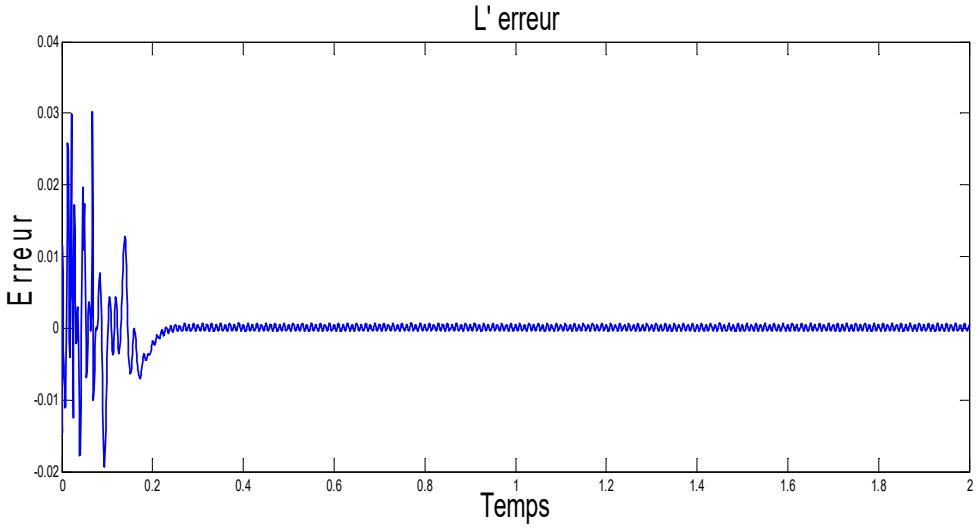


Figure.VI.4. Erreur d'identification par MLP de Courant statorique suivant l'axe *d*

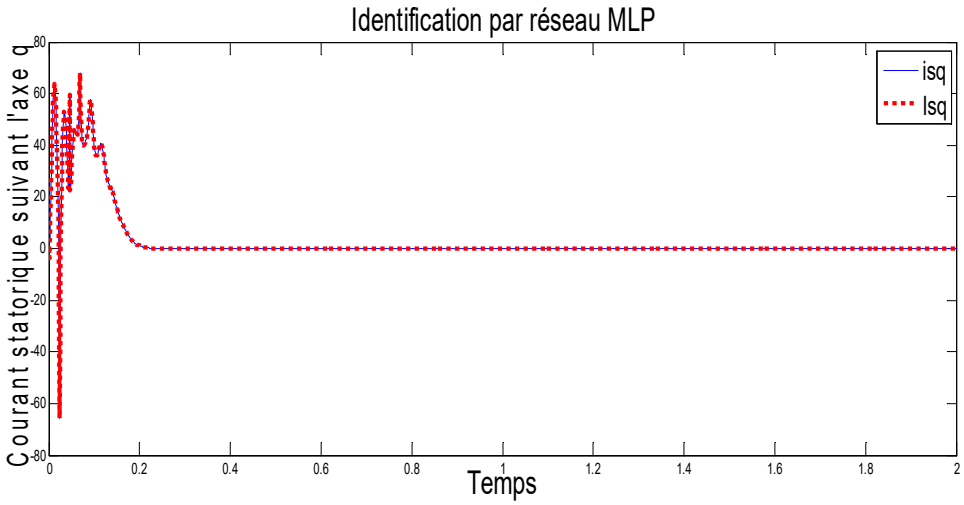


Figure.VI.5. Courant statorique suivant l'axe *q*.

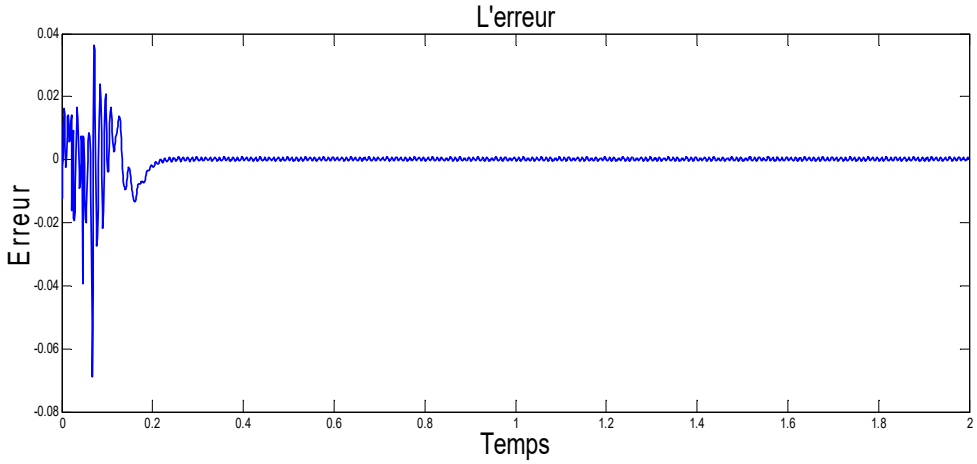


Figure.VI.6. Erreur d'identification par MLP de Courant statorique suivant l'axe *q*

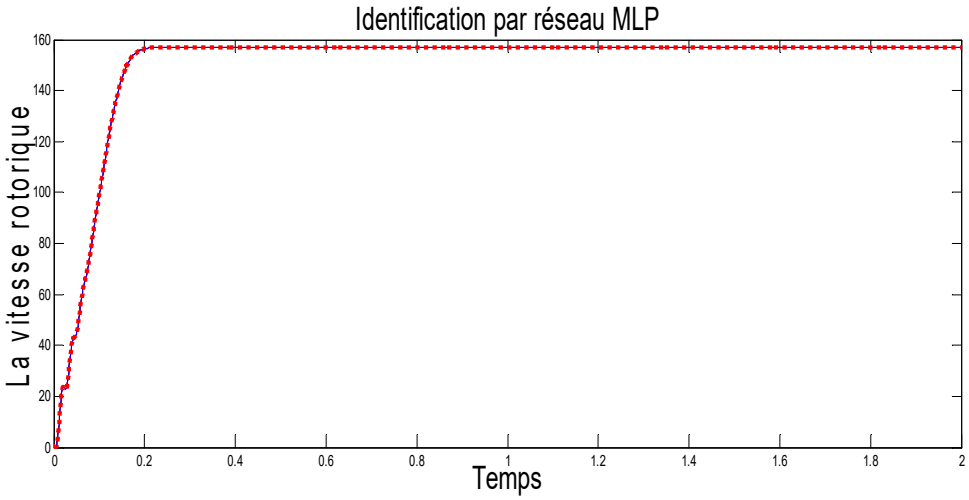


Figure.VI.7. La vitesse rotorique .

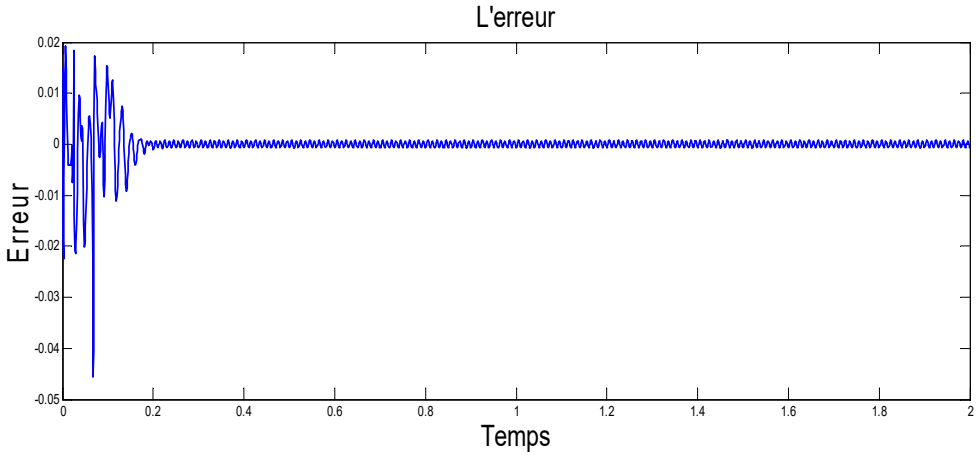


Figure.VI.8. Erreur d'identification par MLP de la vitesse rotorique.

b. Démarrage en charge :

Le moteur étant alimenté par un système de tensions sinusoïdales, Va, Vbet VcPour un démarrage en charge ($Cr = 60\ N.m$).

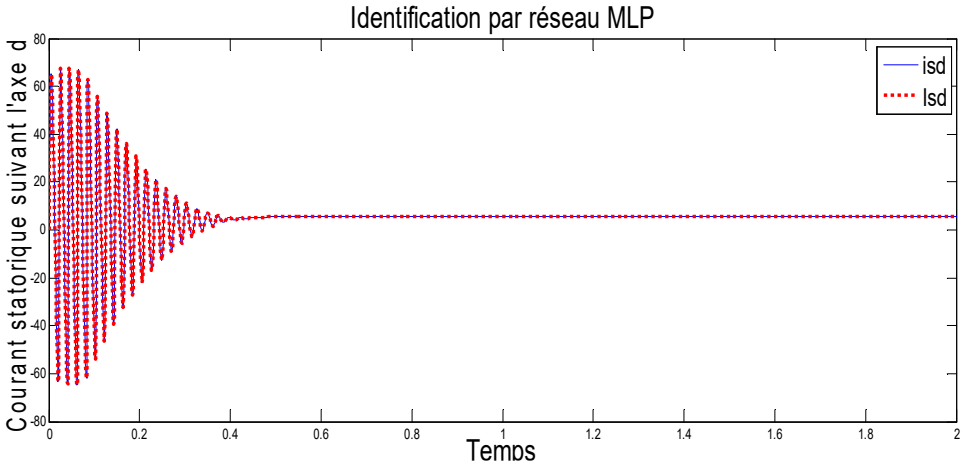


Figure.VI.9. Courant statorique suivant l'axe *d*.

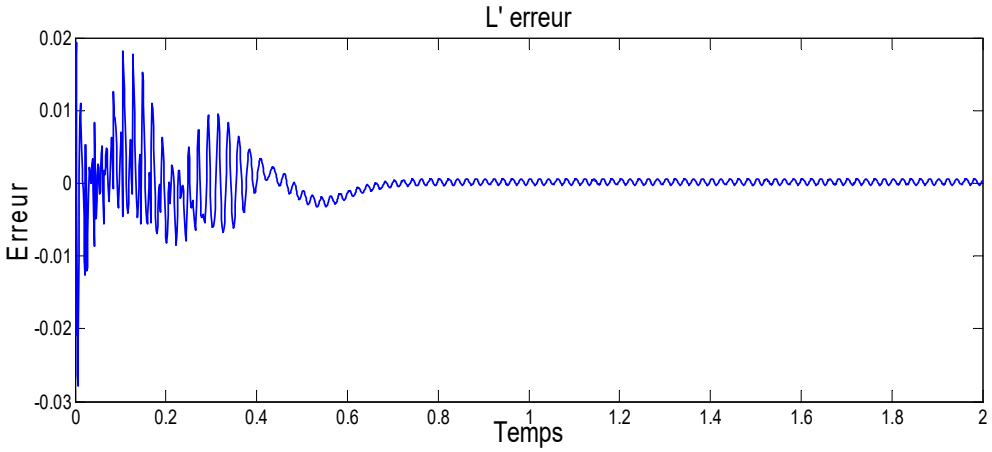


Figure.VI.10. Erreur d'identification par MLP de Courant statorique suivant l'axe *d*

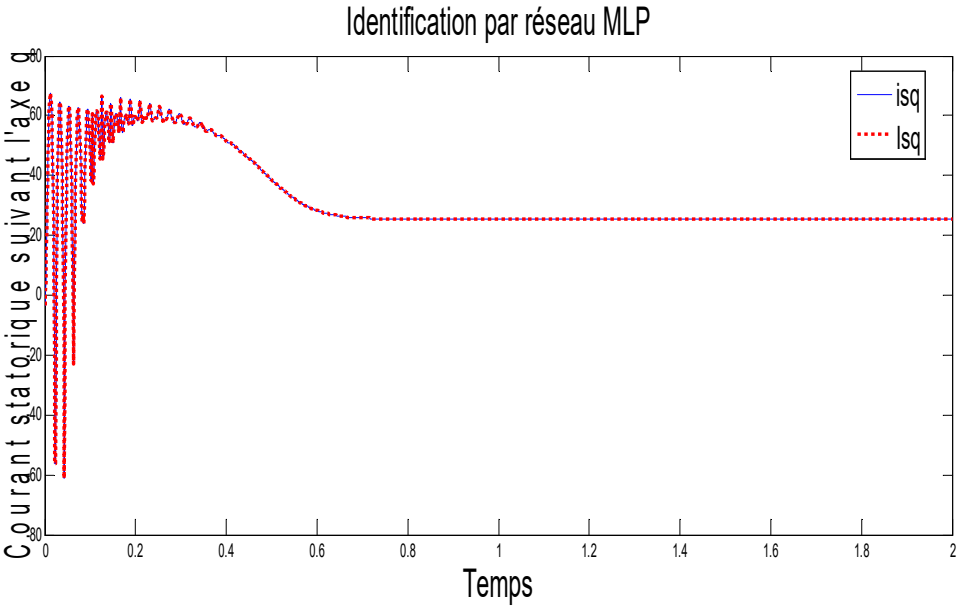


Figure.VI.11. Courant statorique suivant l'axe *q*.

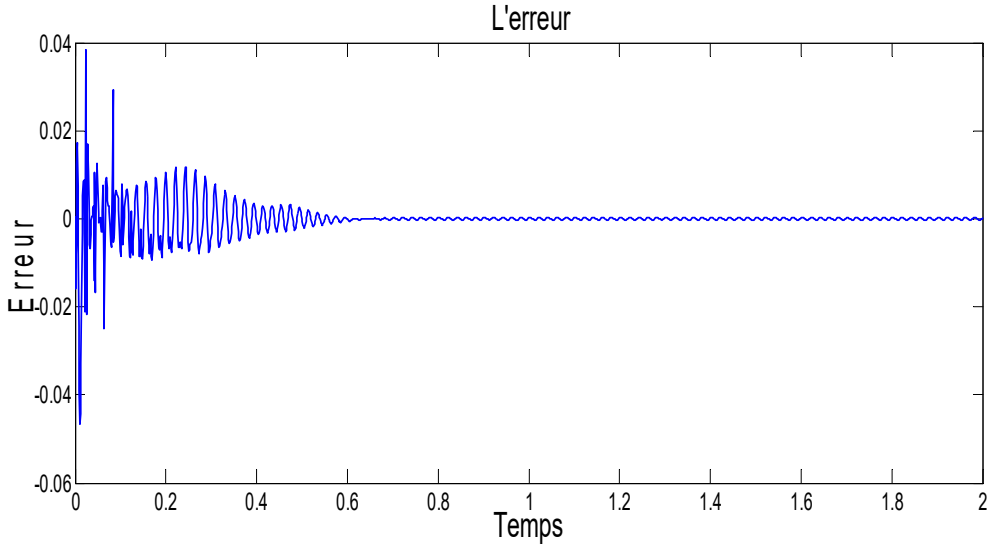


Figure.VI.12. Erreur d'identification par MLP de Courant statorique suivant l'axe *q*.

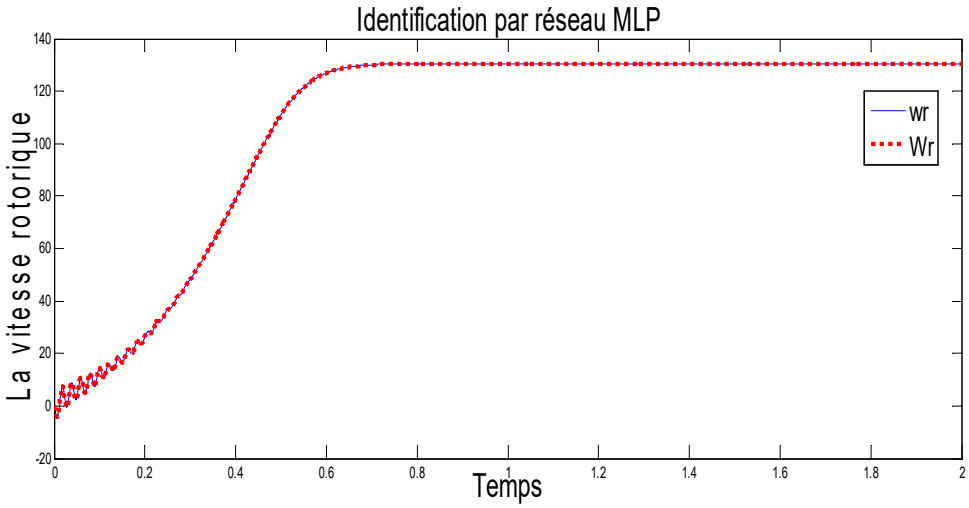


Figure.VI.13. La vitesse rotorique.

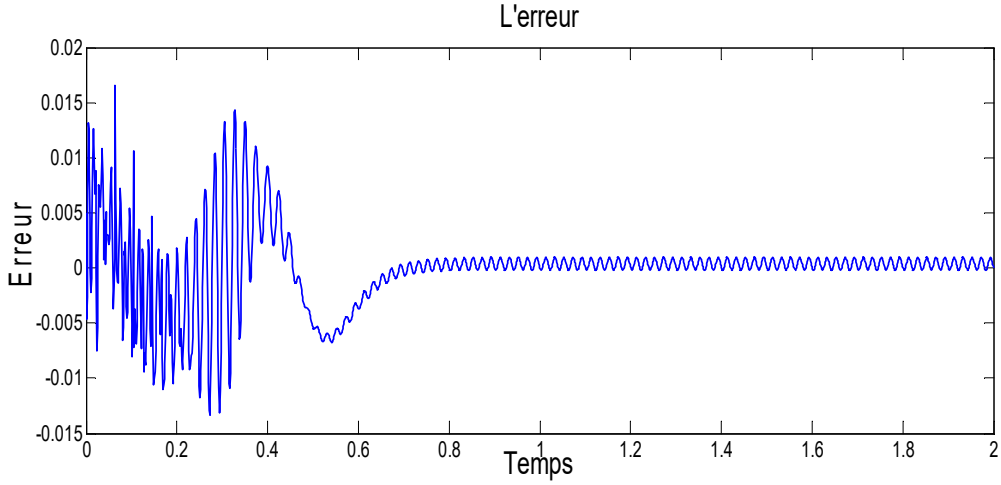


Figure.VI.14. Erreur d'identification par MLP de la vitesse rotorique.

b.1. Commentaire :

On remarque d'après les figures (VI.10, VI.12 et VI.14) que l'erreur diminue au fil de la séquence d'apprentissage cela montre que le réseau de neurone a bien assimilé les sortie qui les sorties de la machine.

Dans les figures (VI.9, VI.11 et VI.13) les sorties de la machine et du réseau sont presque identique, ce qui démontre l'efficacité du réseau à apprendre de son environnement.

IV.3.2.Identification par réseau à base de fonction radiale (RBF):

Le réseau RBF (Radial Basis Functions) est constitué de trois couches; une couche d'entrée qui retransmet les entrées, une seule couche cachée qui contient les neurones RBF qui sont généralement des gaussiennes ayant chacune deux paramètres ; vecteur prototype

μ et un coefficient d'étalement σstrictement positif et une couche de sortie.

Chaque couche est complètement connectée à la suivante et il n'y a pas de connexions à l'intérieur d'une même couche.

La fonction d'activation la plus utilisée est de forme gaussienne donnée par :

$$f(X) = e^{\frac{\|X-\mu\|^2}{2\sigma^2}} \quad \text{(VI.14)}$$

Le vecteur prototype μdéfinit un point dans l'espace d'entrée. La sortie s du neurone est égale à un pour une entrée x égale àμ, puis décroit vers zéro lorsque l'entrée s'éloigne deμ.La vitesse de décroissance est réglée par σ: plus le coefficient est petit et plus la fonctionsera concentrée autour du point μ et proche de zéro ailleurs.

Les neurones de la seconde couche quant à eux calculent la sortie du réseau en effectuant une combinaison linéaire des sorties de ceux de la première couche, avec un biais qui est ajouté au total. La fonction qu'ils réalisent est la suivante :

$$f_1(V) = V.W + b \quad \text{(VI.15)}$$

Où V est un vecteur composé des sorties de tous les neurones de la première couche, W est un vecteur de poids et b est le biais. W et b sont ajustés lors de l'apprentissage.

Chaque sortie du réseau RBF est donnée par la formule suivante:

$$s_i(x) = \sum_{j=1}^{N_g} w_{ij}.e^{\frac{\left\|x-\mu_j\right\|^2}{2\sigma_j^2}} + b_i \quad \text{(VI.16)}$$

Où i est le numéro de la sortie (c'est à dire le numéro du neurone de la seconde couche dont on calcule la sortie), N le nombre de neurones de la couche cachée, μ_j le vecteurprototype du neurone numéro j de la première couche,σ_j son coefficient d'étalement,w_{ij}, j=1,…..,N les Npoids du neurone de sortie i, et b_i son biais[33] .

Les sorties $\widehat{w}_r$, $\hat{I}_{sd}$, $\hat{I}_{sq}$ sorties (vitesse rotorique et les courants stators) du réseau RBF sont définies par les relations suivantes:

$$\widehat{W}_r(k) = \sum_{j=0}^{N} \varphi(k)\,.\,\theta \quad \text{(VI.17)}$$

$$\hat{I}_{sd}(k) = \sum_{j=0}^{N} \varphi(k)\,.\,\theta \quad \text{(VI.18)}$$

$$\hat{I}_{sq}(k) = \sum_{j=0}^{N} \varphi(k) \,.\, \theta \qquad \text{(VI.19)}$$

θ : représente les poids synaptiques w_{ij} .

N: Le nombre de neurones dans la couche cachée.

$\varphi(k)$: La sortie de la couche cachée.

Où

$$\varphi(k) = [\, f_1(k), f_2(k), \ldots\ldots\ldots\ldots f_N(k) \,] \qquad \text{(VI.20)}$$

La méthode des moindres carrés récursifs est l'une des techniques d'estimation récursive des paramètres les plus utilisées à cause de sa robustesse et sa facilité d'implémentation.

IV.3.2.1.SIMULATIONS :

a. Démarrage à vide

Le moteur étant alimenté par un système de tensions sinusoïdales, Va, Vb et Vc Pour un démarrage à vide ($Cr = 0\ N.m$).

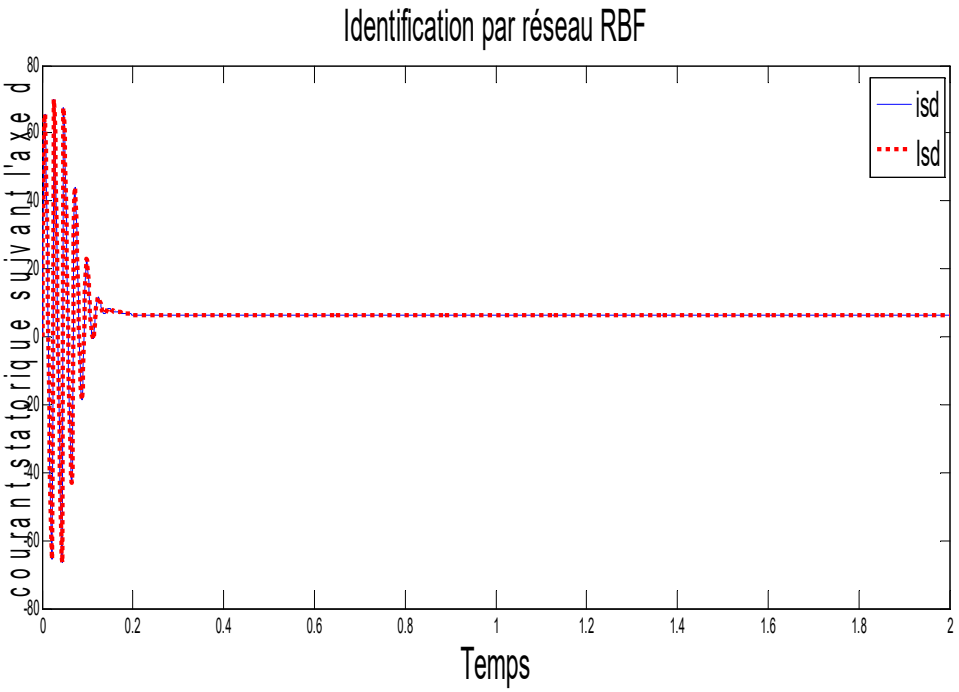


Figure.VI.15. Courant statorique suivant l'axe *d.*

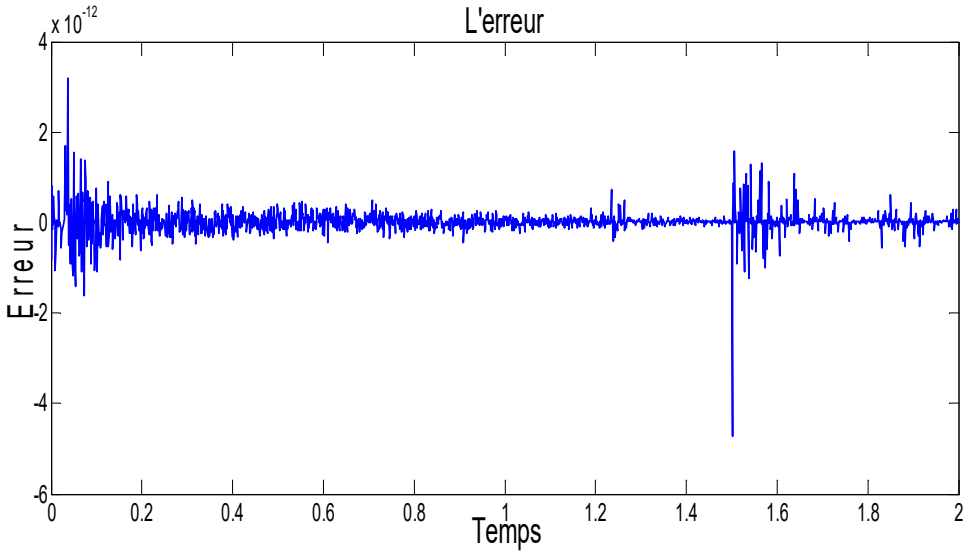


Figure.VI.16. Erreur d'identification par RBF de Courant statorique suivant l'axe *d*

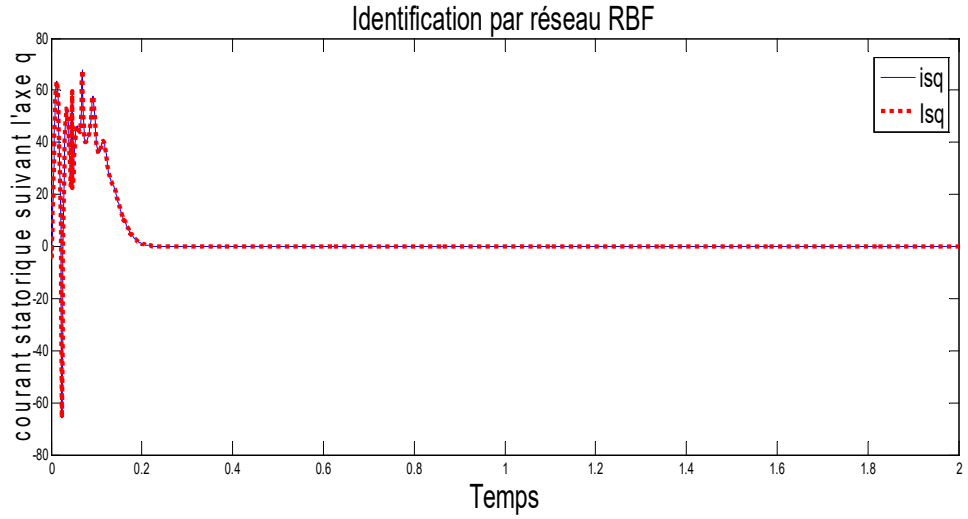


Figure.VI.17. Courant statorique suivant l'axe *q*.

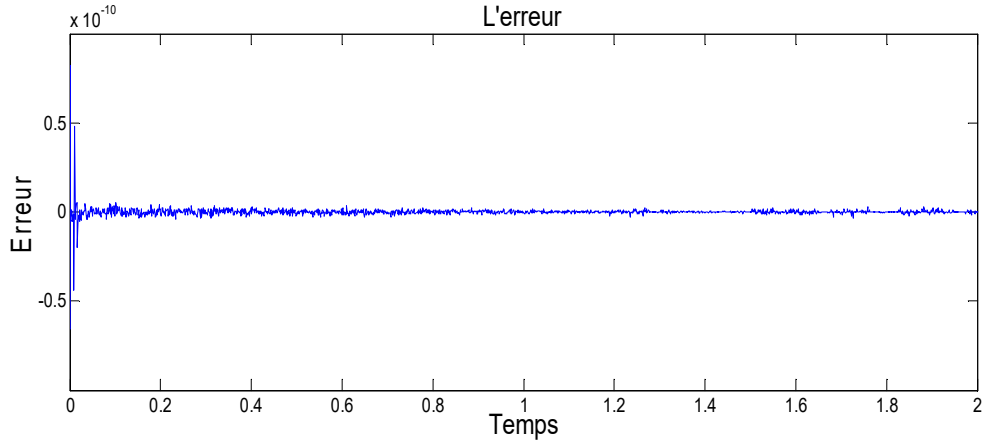


Figure.VI.18. Erreur d'identification par RBF de Courant statorique suivant l'axe *q*.

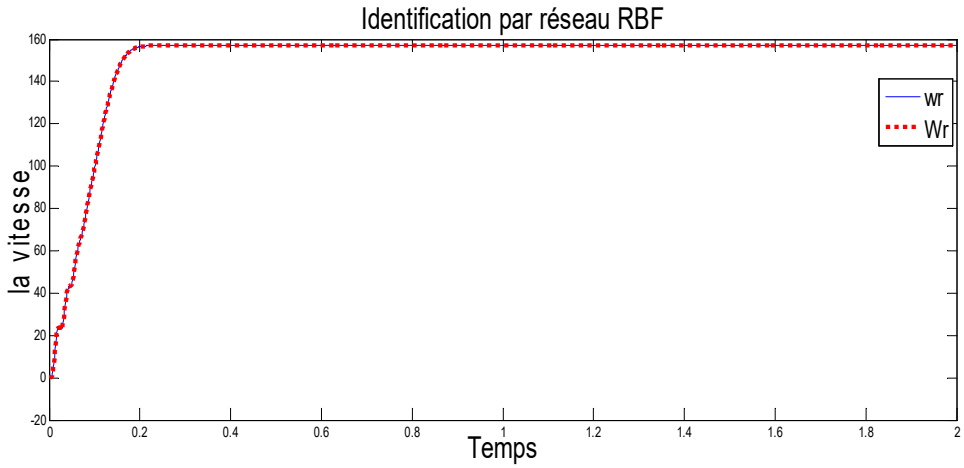


Figure.VI.19. La vitesse rotorique .

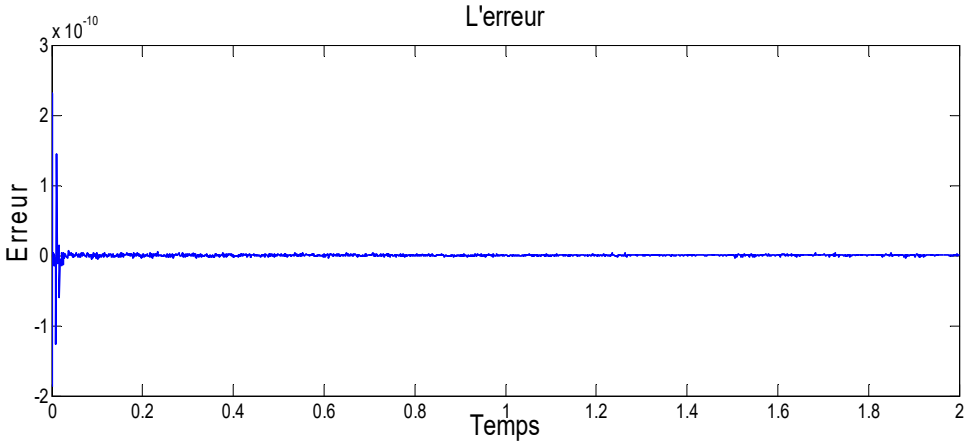


Figure.VI.20. Erreur d'identification par RBF de la vitesse rotorique.

b. Démarrage en charge :

Le moteur étant alimenté par un système de tensions sinusoïdales, Va, Vb et Vc Pour un démarrage en charge ($Cr = 60\ N.m$).

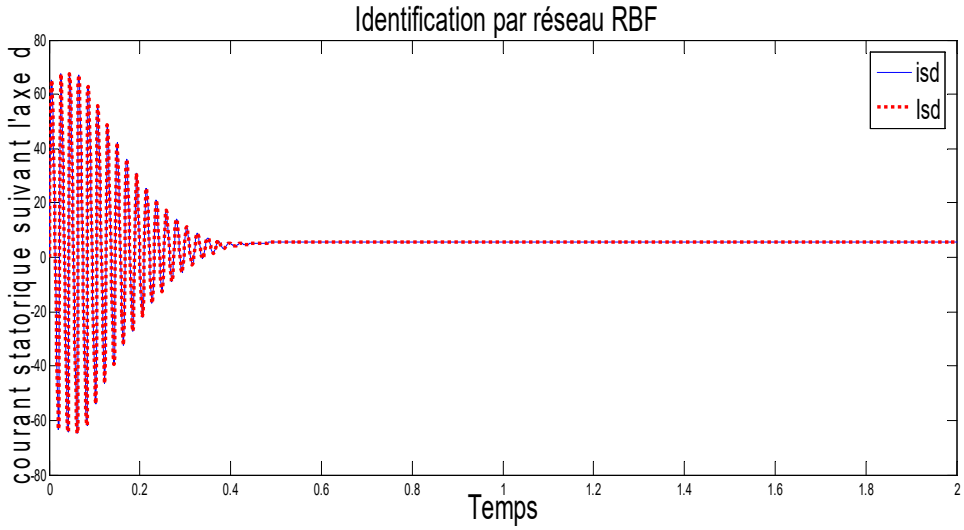


Figure.VI.21. Courant statorique suivant l'axe *d*.

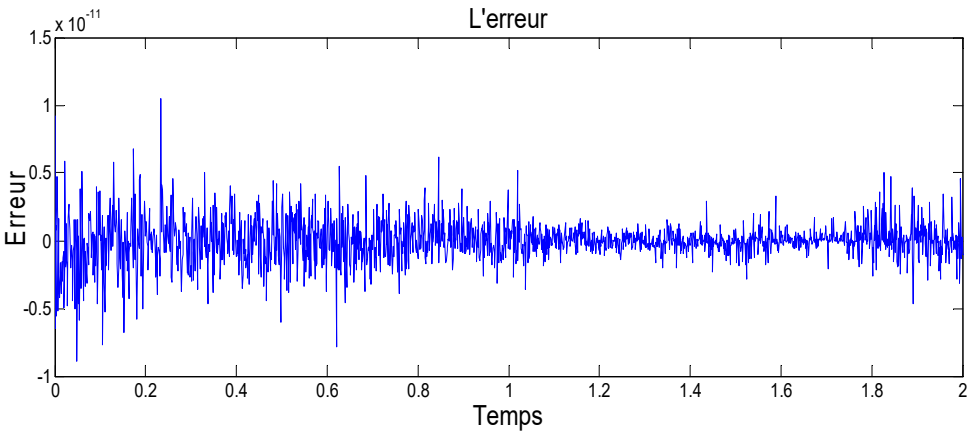


Figure.VI.22. Erreur d'identification par RBF de Courant statorique suivant l'axe *q*

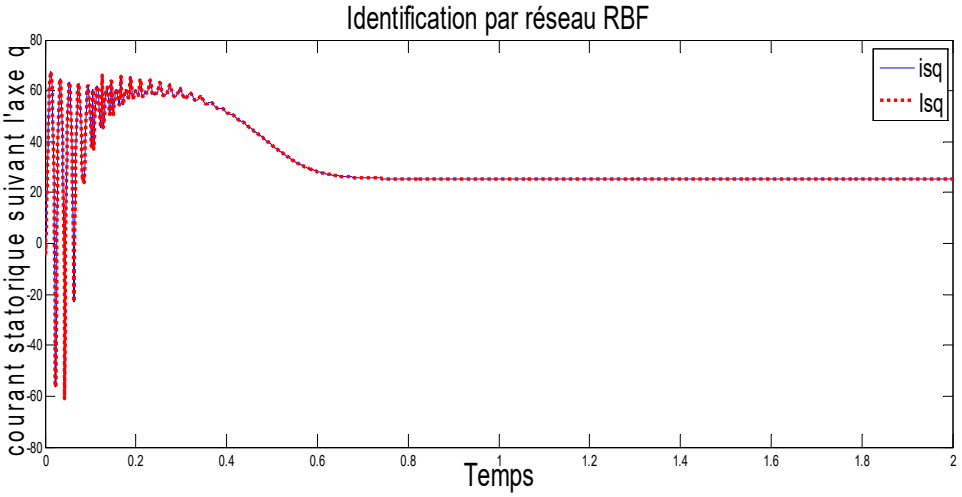


Figure.VI.23. Courant statorique suivant l'axe *d*

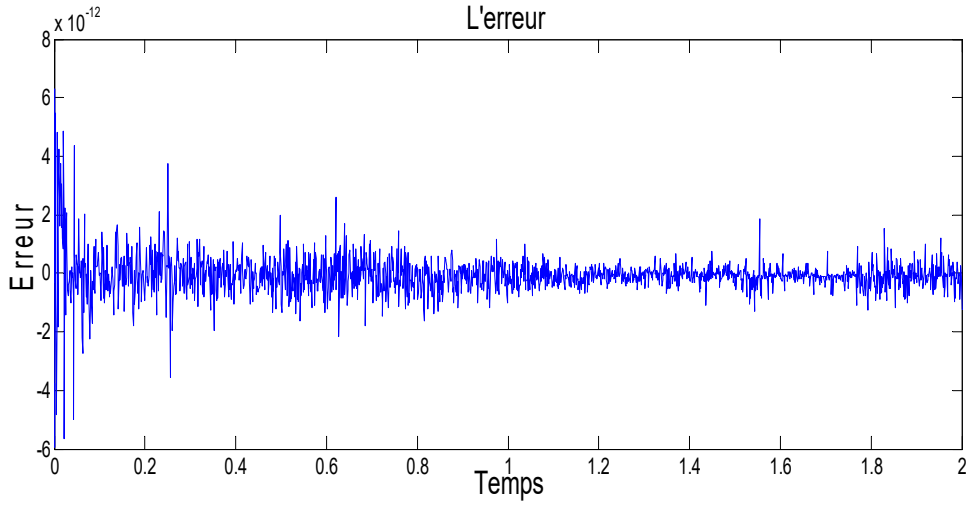


Figure.VI.24. Erreur d'identification par RBF de Courant statorique suivant l'axe *q*

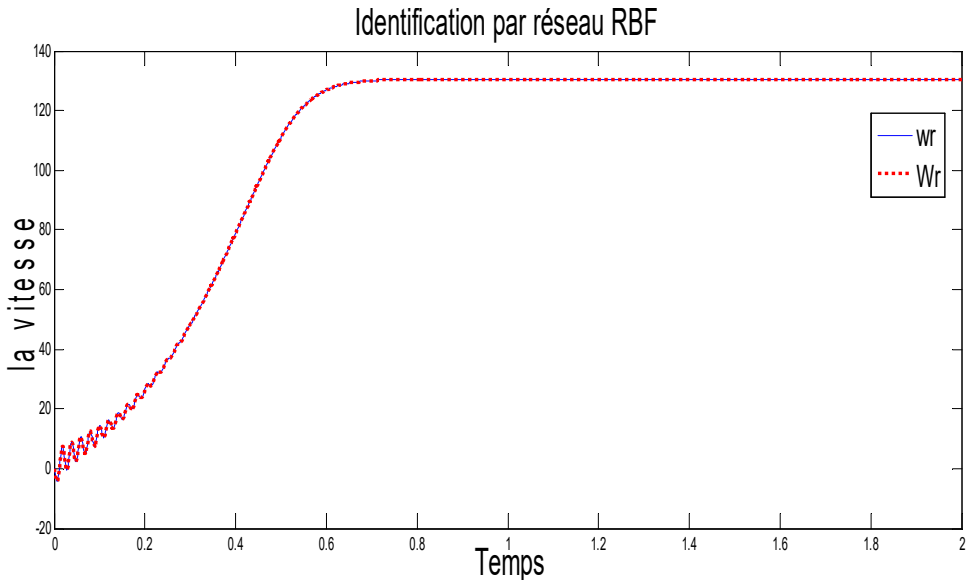


Figure.VI.25. La vitesse rotorique .

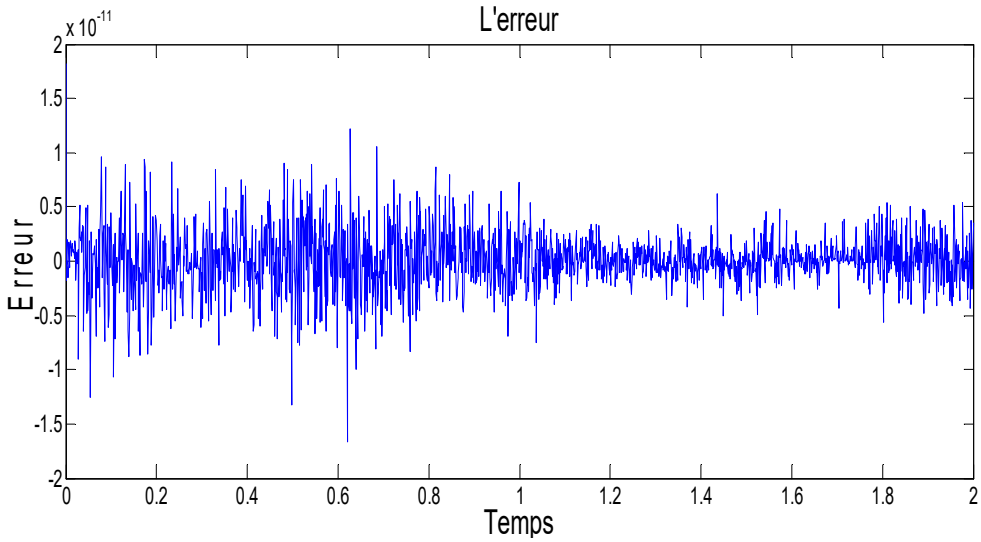


Figure.VI.26. Erreur d'identification par RBF de la vitesse rotorique.

b.1. Commentaire :

On remarque d'après les figures (VI.22, VI.24 et VI.26) que l'erreur diminue au fil de la séquence d'apprentissage cela montre que le réseau de neurone a bien assimilé les sortie qui les sorties de la machine.

Dans les figures (VI.21, VI.23 et VI.25) les sorties de la machine et du réseau sont presque identique, ce qui démontre l'efficacité du réseau à apprendre de son environnement.

VI.4.CONCLUSION :

Dans ce chapitre nous avons appliqué le technique d'identification par réseaux de neurones sur la machine asynchrone, et pour cela en utilisant deux architectures différentes des réseaux de neurones, (le perceptron multicouche et les réseaux de fonctions à base radiale). La capacité d'approximation et d'apprentissage des réseaux de neurones ont été utilisées pour l'identification du comportement dynamique des systèmes non linéaires.

L'apprentissage dans les réseaux MLP, repose sur les méthodes d'optimisation non linéaires. Cependant, la surface d'erreur pour ces architectures est souvent très complexe et présente des caractéristiques peu satisfaisantes pour effectuer une descente de gradient (minima locaux, plateau où les pentes sont très faibles,…). Ceci est l'inconvénient majeur des réseaux MLP.

L'utilisation d'un réseau RBF, pour lequel l'apprentissage est basé sur les méthodes d'optimisation linéaire dans les problèmes d'identification, fournit une autre alternative plus efficace. Malheureusement, le nombre de neurones cachés augmente, dans ce cas, avec le degré de complexité du problème traité d'une façon considérable ; c'est l'inconvénient des réseaux à une seule couche cachée par conséquent l'apprentissage devient très lent.

CONCLUSION GENERALE :

L'étude d'un système dynamique non linéaire, quelle que soit sa nature (industrielle, environnementale, financière, etc.) et quel que soit l'objectif visé (commande, optimisation du fonctionnement, prédiction, analyse du comportement, etc.), nécessite la mise en place d'une représentation capable de reproduire son comportement. Cette représentation, communément appelée modèle, permet dans certaines applications de simuler le comportement du système, notamment lorsque l'expérimentation est coûteuse.

Parmi les différents types de modélisation qui existent, la modélisation mathématique connaît plus de succès grâce au progrès de l'informatique qui a permis une avancée significative des méthodes de calcul numérique. La modélisation mathématique d'un système est une représentation mathématique sous forme d'une relation liant les différentes variables régissant son fonctionnement. Cependant, la détermination de ces variables et de la structure de la relation qui les lie constitue souvent un problème majeur.

Pour certains systèmes, il est possible d'établir ces relations à partir de connaissances physiques, chimiques, biologiques ou autres : une telle représentation est appelée modèle de connaissance, modèle boîte blanche ou modèle théorique. Il est cependant très difficile, voire parfois impossible, d'établir de tels modèles pour des systèmes complexes. La démarche la plus courante est l'établissement d'un modèle boîte noire (modélisation expérimentale), basé sur les informations recueillies sur le fonctionnement du système, notamment les mesures faites sur les variables. La difficulté d'une modélisation de type boîte noire est de trouver un compromis entre ces deux objectifs contradictoires.

Plusieurs approches ont été proposées pour la représentation des systèmes dynamiques non linéaires : méthode de linéarisation du système, représentation de Volterra Wiener, représentations de Hammerstein et de Wiener, etc. Plus récemment, d'autres techniques ont vu le jour notamment celles basées sur les Réseaux de Neurones Artificiels (RNA).

Les réseaux neuronaux artificiels ont été utilisés avec succès comme blocs structurels dans l'architecture de l'identification et la commande des systèmes dynamiques non linéaires. Cette architecture est basée sur l'entraînement du réseau utilisant les données d'entrée-sortie du système.

Nous avons organisé le contenu de notre travail comme suit :

En premier lieu la démarche présentée des généralités sur l'identification des systèmes non linéaires. Les différents types de modèles de systèmes sont définis ainsi que les étapes

de l'identification, et les algorithmes d'adaptation qui sont des éléments essentiels de l'identification.

Après avoir donné un aperçu général sur les réseaux de neurones, présenté les réseaux statiques et étudié en détail les structures neuronales MLP et RBF, ainsi que leurs algorithmes d'apprentissage, nous avons envisagé l'application de ces structures à l'identification des systèmes non linéaires.

Ensuite notre première application de l'approche neuronale était l'identification des systèmes non linéaires inconnus SISO et MIMO, les résultats de simulations obtenus ont démontrés que les modèles neuronaux sont de bons approximateurs universels pour les fonctions non linéaire inconnues, après cette étape nous avons appliqué la technique d'identification par réseaux de neurones sur la machine asynchrone, et pour cela en utilisant deux architectures différentes des réseaux de neurones (MLP et RBF).

L'apprentissage dans les réseaux MLP, repose sur les méthodes d'optimisation non linéaires. Cependant, la surface d'erreur pour ces architectures est souvent très complexe et présente des caractéristiques peu satisfaisantes pour effectuer une descente de gradient (minima locaux, plateau où les pentes sont très faibles,…). Ceci est l'inconvénient majeur des réseaux MLP.

L'utilisation d'un réseau RBF, pour lequel l'apprentissage est basé sur les méthodes d'optimisation linéaire dans les problèmes d'identification, fournit une autre alternative plus efficace. Malheureusement, le nombre de neurones cachés augmente, dans ce cas, avec le degré de complexité du problème traité d'une façon considérable ; c'est l'inconvénient des réseaux à une seule couche cachée par conséquent l'apprentissage devient très lent.

Finalement, dans ce mémoire, on a supposé, pour accomplir les tâches d'identification, que les systèmes étudiés sont stables. Tous les exemples considérés dans ce travail ont été pris de [54][58].

A.1. MODELISATION DE LA MACHINE ASYNCHRONE :

L'étude de cette machine traduit les lois de l'électromagnétisme dans le contexte habituel des hypothèses simplificatrices [60] **:**

- L'entrefer du moteur est d'épaisseur uniforme, négligeant ainsi l'effet des encoches.
- Le circuit magnétique non saturé et à une perméabilité constante, l'hystérésis et les courants de Foucault sont négligeables
- Les résistances des enroulements ne varient pas avec la température, en négligeant l'effet de peau et les pertes fer.
- Le bobinage triphasé est symétrique et la répartition de la force magnétomotrice dans l'entrefer est sinusoïdale.

Dans ces conditions, si on considère que le moteur à induction est triphasé au stator et au rotor. Les trois types d'équations traduisant le comportement du moteur sont :

- Les équations électriques.
- Les équations magnétiques.
- L'équation mécanique.

La représentation schématique de la MAS dans l'espace électrique est donnée sur la figure(A.1).

Elle est munie de six enroulements, [61].

Le stator de la machine est formé de trois enroulements fixes décalés de 120° dans l'espace et traversés par trois courants variables.

Le rotor peut être modélisé par trois enroulements identiques décalés dans l'espace de 120°. Ces enroulements sont en court-circuit et la tension à leurs bornes est nulle.

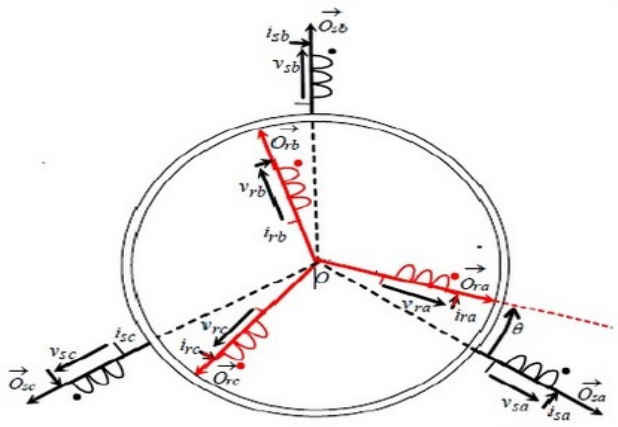


Figure A.1. Représentation schématique d'une machine asynchrone triphasée.

A.1.1 .Equations électriques :

Les six enroulements (a, b, c, A, B, C) représentés sur la figure (A.1) obéissent aux équations matricielles suivantes [60].

$$[V_s]=[R_s]\cdot[i_s]+\frac{d}{dt}[\phi_s] \tag{A.1}$$

$$[V_r]=[R_r]\cdot[i_r]+\frac{d}{dt}[\phi_r]=[0 \quad 0 \quad 0]^T \tag{A.2}$$

Avec :

$[V_s]$: Vecteur tension. $[i_s]$: Vecteur courant. $[\phi_s]$: Vecteur flux statorique.

$[R_s]$: Matrice résistance , s, r: Indices stator et rotor, respectivement.

A.1.2.Equations magnétiques :

Les hypothèses simplificatrices citées antérieurement conduisent à des relations linéaires entre les flux et les courants de la machine asynchrone, ces relations s'écrivent matricielle - ment comme suit, [62] :

Pour le stator :

$$[\phi_s]=[L_s]\cdot[i_s]+[M_{sr}][i_r] \tag{A.3}$$

Pour le rotor :

$$[\phi_r]=[L_r]\cdot[i_r]+[M_{rs}][i_s] \tag{A.4}$$

$[L_s]$, $[L_r]$: les matrices d'inductance statorique et rotorique .

$[M_{sr}]$: correspond à la matrice des inductances mutuelles stator-rotor.

On désigne par :

$$[L_s]=\begin{bmatrix} l_s & M_s & M_s \\ M_s & l_s & M_s \\ M_s & M_s & l_s \end{bmatrix} \tag{A.5}$$

$$[L_r]=\begin{bmatrix} l_r & M_r & M_r \\ M_r & l_r & M_r \\ M_r & M_r & l_r \end{bmatrix} \tag{A.6}$$

$$[L_s]=[M]^T=M\begin{bmatrix}\cos(\theta) & \cos(\theta+2\frac{\pi}{3}) & \cos(\theta-2\frac{\pi}{3})\\ \cos(\theta-2\frac{\pi}{3}) & \cos(\theta) & \cos(\theta+2\frac{\pi}{3})\\ \cos(\theta+2\frac{\pi}{3}) & \cos(\theta-2\frac{\pi}{3}) & \cos(\theta)\end{bmatrix} \quad (A.7)$$

θ : La position absolue entre le stator et le rotor.

l_r , l_s : Inductance propre du rotor et du stator, respectivement.

M : Inductance mutuelle cyclique entre stator-rotor.

Finalement les équations de tensions deviennent :

Pour le stator :

$$[V_{sabc}]=[R_s]\cdot[i_{sabc}]+\frac{d}{dt}\{[L_s][i_{sabc}]+[M_{sr}][i_{rabc}]\} \quad (A.8)$$

Pour le rotor :

$$[V_{rabc}]=[R_r]\cdot[i_{rabc}]+\frac{d}{dt}\{[L_r][i_{rabc}]+[M_{rs}][i_{sabc}]\} \quad (A.9)$$

A.1.3.Equations mécaniques :

L'étude des caractéristiques de la machine asynchrone fait introduire de la variation non seulement des paramètres électriques (tension, courant, flux) mais aussi des paramètres mécaniques (couple, vitesse) [63].

$$C_{em}=p[i_{sabc}]^T\frac{d}{dt}[M_{sr}][i_{rabc}] \quad (A.10)$$

L'équation du mouvement de la machine est :

$$J\frac{d}{dt}\Omega=C_{em}-C_r-f_r\Omega \quad (A.11)$$

Avec : J : moment d'inertie des masses tournantes.

C_r : Couple résistant impose à l'arbre de la machine.

Ω : vitesse rotorique.

C_{em} : Couple électromagnétique.

f_r : Coefficient de frottement visqueux.

$f_r\Omega$: Terme de couple de frottement visqueux.

A.2. Transformation du système triphasée :

La mise en équation des moteurs triphasés aboutit à des équations différentielles à coefficients variables. L'étude analytique du comportement du système est alors relativement laborieuse, vu le grand nombre de variable. On utilise alors des transformations qui permettent de décrire le comportement de la machine à l'aide d'équations différentielles à coefficients constants.

Les transformations utilises doivent conserver la puissance instantanée et la réciprocité des inductances mutuelles. Ceci permet d'établir une expression du couple électromagnétique dans le repère correspondant au système transforme [63].

A.2.1.Transformation CLARKE/CONCORDIA :

Le but de l'utilisation de cette transformation c'est de passer d'un système triphasé abc vers un système diphasé α, β. Il existe principalement deux transformations : Clarke et Concordia.

La transformation de CLARK conserve l'amplitude des grandeurs électriques. Tandis que celle de CONCORDIA, conserve la puissance.

Transformation de Concordia	**Transformation de Clarke**
Passer d'un système triphasé abc vers un système diphasé α β	
$\begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \xrightarrow{T_{23}} \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix}$ c.-à-d. $[x_{\alpha\beta}] = T_{23}[x_{abc}]$ Avec $T_{23} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}$	$\begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \xrightarrow{C_{23}} \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix}$ c.-à-d. $[x_{\alpha\beta}] = C_{23}[x_{abc}]$ Avec $C_{23} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}$
Passer d'un système diphasé α β vers un système abc	
$\begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} \xrightarrow{T_{32}} \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix}$ c.-à-d. $[x_{abc}] = T_{32}\,[x_{\alpha\beta}]$ Avec $T_{32} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}$	$\begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} \xrightarrow{C_{32}} \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix}$ c.-à-d. $[x_{abc}] = C_{32}\,[x_{\alpha\beta}]$ Avec $C_{32} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}$

Tableau A.1.Passage d'un système triphasé au système

Le choix de matrice de passage non norme (Clarke) est bien pratique en commande où l'on traite des grandeurs *d q* (I_{sd} , I_{qs} que l'on verra par la suite). En effet, cela permet, par exemple, d'apprécier directement le module du courant qui est absorbe par le moteur, sans avoir à passer par un coefficient multiplicateur. Mathématiquement parlant, le choix d'une matrice normée (Concordia) est souvent utilise pour des raisons de symétrie de transformation directe et inverse. Nous allons utiliser la transformation de Concordia dans notre modélisation. Son application aux équations de la machine écrites ci-dessous [63] donne :

$$T_{23}\left[v_{abcs}\right]=\left[v_{\alpha\beta s}\right]=T_{23}\left\{R_s\left[i_{abcs}\right]+\frac{d}{dt}\left[\phi_{abcs}\right]\right\} \tag{A.12}$$

$$\left[v_{\alpha\beta s}\right]=R_s T_{23}\left[i_{abcs}\right]++\frac{d}{dt}T_{23}\left[\phi_{abcs}\right] \tag{A.13}$$

$$\left[v_{\alpha\beta s}\right]=R_s\left[i_{abcs}\right]++\frac{d}{dt}\left[\phi_{\alpha\beta s}\right] \tag{A.14}$$

On a alors réduit le système de trois (3) équations à un système de deux (2) équations.

De même pour le rotor :

$$\left[v_{\alpha\beta r}\right]=R_r\left[i_{abcr}\right]++\frac{d}{dt}\left[\phi_{\alpha\beta r}\right] \tag{A.15}$$

Ainsi que pour l'écriture des flux en fonction des courants. L'intérêt pour les flux, c'est que les matrices 3*3 des inductances vont être réduites à des matrices 2*2. On a alors l'apparition des inductances cycliques :

$$L_s=l_s-m_s \quad , \quad L_r=l_r-m_r \quad , \quad M=\frac{3}{2}m_{sr}$$

Alors :

$$\begin{bmatrix}\phi_{\alpha\beta s}\\ \phi_{\alpha\beta r}\end{bmatrix}=\begin{bmatrix}L_s & 0 & M\cdot P(\theta) & \\ 0 & L_s & & \\ M\cdot P(-\theta) & & L_r & 0\\ & & 0 & L_r\end{bmatrix}\begin{bmatrix}i_{\alpha\beta s}\\ i_{\alpha\beta r}\end{bmatrix}$$

Où la matrice P (θ) est la matrice de rotation :

$$P(\theta)=\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}$$

On dispose à présent d'une modélisation de la machine asynchrone dans deux repères sépares : les grandeurs statoriques sont exprimées dans le repère α β stator et les grandeurs rotoriques dans le repère α β rotor. Il faut exprimer toute la modélisation dans un repère commun. En effet, si l'on examine de plus prés la matrice des inductances [63].

$$\begin{bmatrix} L_s & 0 & & \\ 0 & L_s & M\cdot P(\theta) & \\ & & L_r & 0 \\ M\cdot P(-\theta) & & 0 & L_r \end{bmatrix}$$

On s'aperçoit que les grandeurs statoriques sont liées aux grandeurs rotoriques à travers l'angle

On choisi alors de transformer les deux grandeurs statoriques et rotoriques vers un repère commun dit *d ,q* et ceci à l'aide de deux transformations dans le plan qui sont des rotations. Ce sont ces transformations ainsi que la transformation de Concordia ou de Clarke qui constitue la transformation de Park.

Alors on peut écrire toute grandeur dans le repère (*d,q*) on utilisant le produit matricielle suivant :

$$\begin{bmatrix}x_\alpha\\ x_\beta\end{bmatrix}=\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}x_d\\ x_q\end{bmatrix} \text{ avec : } \bar{x}=(\bar{I}_{s,r},\bar{v}_{s,r},\overline{\Psi}_{s,r})$$

A.2.2.TRANSFORMATIONDE PARK :

La transformation de Park à pour but de traiter une large gamme de machines de façon unifiée en un modèle unique. Cette conversion est appelée souvent transformation des axes, fait correspondant aux deux enroulements de la machine originale suivie d'une rotation, les enroulements équivalents du point de vue électrique et magnétique. Cette transformation ainsi, pour l'objectif de rendre les inductances mutuelles du modèle indépendantes de l'angle de rotation [64].

a.Différents repères :

L'isotropie du moteur asynchrone permet une souplesse dans la composition des équations de la machine selon deux axes à l'aide des composantes de Park, cela nécessite l'utilisation d'un repère qui permet de simplifier au maximum les expressions analytiques.

Il existe différentes possibilités pour le choix du repère d'axes, se ramène pratiquement à trois référentiels (systèmes biphasés) orthogonaux : [14]

- ✓ Référentiel immobile par rapport au stator : $(\alpha-\beta)\longrightarrow\omega=0$.
- ✓ Référentiel immobile par rapport au rotor : $(x-y)\longrightarrow\omega=\omega_r$
- ✓ Référentiel immobile par rapport au champ tournant : $(d-q)\longrightarrow\omega=\omega_r$

Où : ω : Vitesse angulaire de rotation du système d'axes biphasé par rapport au système d'axes triphasé.

La transformation de Park est souvent définir par la matrice normalisée [P] comme suit [64] :

$$[P]=\sqrt{\frac{2}{3}}\begin{bmatrix}\cos(\theta) & \cos(\theta-2\frac{\pi}{3}) & \cos(\theta+2\frac{\pi}{3})\\ -\sin(\theta) & -\sin(\theta-2\frac{\pi}{3}) & -\sin(\theta+2\frac{\pi}{3})\\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\end{bmatrix} \quad \text{(A.16)}$$

$\sqrt{\frac{2}{3}}$: Le facteur ($\sqrt{\frac{2}{3}}$) : pour la conservation de la puissance électrique instantanée.

$$\begin{bmatrix}X_a\\X_b\\X_c\end{bmatrix}=[M]^T=\begin{bmatrix}X_U\\X_W\\X_0\end{bmatrix} \quad \text{Avec :} \quad [P]^{-1}=\begin{bmatrix}\cos(\theta) & -\sin(\theta) & \frac{1}{\sqrt{2}}\\ \cos(\theta-2\frac{\pi}{3}) & -\sin(\theta-2\frac{\pi}{3}) & \frac{1}{\sqrt{2}}\\ \cos(\theta+2\frac{\pi}{3}) & -\sin(\theta+2\frac{\pi}{3}) & \frac{1}{\sqrt{2}}\end{bmatrix} \quad \text{(A.17)}$$

b.Application de la transformation de Park au modèle de la MAS :

b.1.Equations électriques :

$$V_{sd}=R_s i_{sd}+\frac{d}{dt}\phi_{sd}-\omega_s\phi_{sq} \quad \text{(A.18)}$$

$$V_{sq}=R_s i_{sq}+\frac{d}{dt}\phi_{sq}-\omega_s\phi_{sd} \quad \text{(A.19)}$$

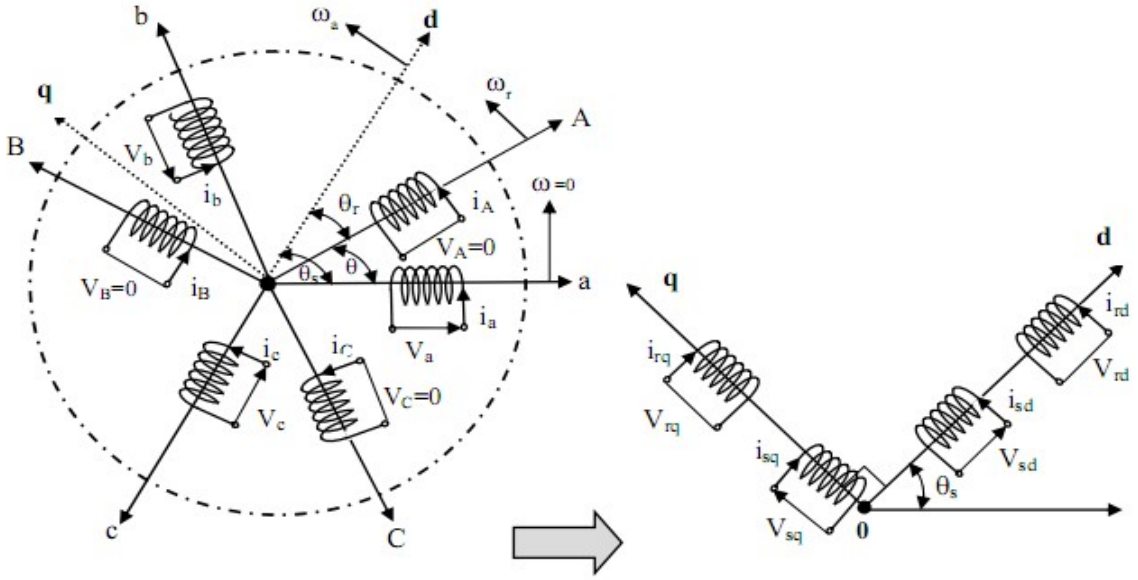
$$0=R_r i_{rd}+\frac{d}{dt}\phi_{rd}-(\omega_s-\omega_r)\phi_{rq} \quad \text{(A.20)}$$

$$0 = R_r i_{rq} + \frac{d}{dt}\phi_{rq} - (\omega_s - \omega_r)\phi_{rd} \tag{A.21}$$

b.2.Equations magnétiques :

$$\begin{aligned} \phi_{sd} &= L_s i_{sd} + M i_{rd} \\ \phi_{sq} &= L_s i_{sq} + M i_{rq} \\ \phi_{rd} &= L_r i_{rd} + M i_{sd} \\ \phi_{rq} &= L_s i_{sd} + M i_{sq} \end{aligned} \tag{A.22}$$

Avec : $L_s = l_s + M$, $L_r = l_r + M$, $M = \frac{3}{2} M_0$



a) Modèle triphasé réel. b) Modèle biphasé équivalent.

Figure A.2. Représentation de la machine asynchrone triphasée et sa machine biphasée.

b.3.Equation mécanique :

$$C_{em} = p\frac{M}{L_r}(\phi_{rq_s} i_{sq} - \phi_{rq}\ i_{sd}) \tag{A.23}$$

A.3. ALIMENTATION DE LA MACHINE ASYNCHRONE :

Pour une machine asynchrone alimentée en tension, si on considère le courant statorique i_s et le flux rotorique ϕ_r comme variables d'état, et la pulsation ω_r et les tensions V_{sd} , V_{sq} **,** comme grandeur de commande et le couple C_r comme une perturbation, on aura le schéma bloc suivant, (Figure.A.3).

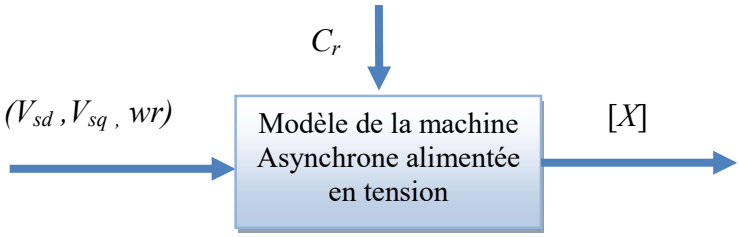


Figure. A.3. Schéma bloc de la machine asynchrone alimentée en tension.

A. 3.1.REPRESENTATION D'ETAT DU MODELE DE LA MACHINE ASYNCHRONE :

Le modèle de la machine asynchrone [60], est un modèle d'état de la forme:

$$\begin{cases} [\dot{X}] = [A].[X] + [B].[U] \\ [Y] = [C].[X] \end{cases} \quad \text{(A.24)}$$

Le vecteur de sortie $[X]$, peut avoir une des formes des différentes expressions :

$$[X] = [i_{sd} i_{sq} i_{rd} i_{rq}]^T \quad \text{(A.25)}$$

Ou bien : $[X] = [\phi_{sd} \phi_{sq} \phi_{rd} \phi_{rq}]^T$ (A.26)

Ou bien : $[X] = [\phi_{sd} \phi_{sq} i_{rd} i_{rq}]^T$ (A.27)

Ou bien : $[X] = [i_{sd} i_{sq} i_{rd} i_{rq} w_r]^T$ (A.28)

Notre choix est porté sur le vecteur : $[X] = [i_{sd} i_{sq} i_{rd} i_{rq} w_r]^T$

Le vecteur d'entrée $[U]$ est donné par :

$$[U] = [V_{sd}\ V_{sq}]^T \quad \text{(A.29)}$$

Le vecteur de sortie est donné par :

$$[Y] = [i_{sd} i_{sq} w_r]^T \quad \text{(A.30)}$$

Avec :

$$A(X)=\begin{pmatrix}\xi\left(-L_r.R_s.i_{sd}+M^2.P.w_r.i_{sq}+M.R_r.i_{rd}+P.L_r.M.w_r.i_{rq}\right)\\ \xi\left(-M^2.P.w_r.i_{sd}-L_r.R_s.i_{sq}-P.L_r.M.w_r.i_{rd}+M.R_r.i_{rq}\right)\\ \xi\left(M.L_s.w_r.i_{sd}+M.R_s.i_{sq}+L_s.L_rP.w_r.i_{rd}+L_r.R_r.i_{rq}\right)\\ \xi\left(P.i_s.w_r.i_{sd}+M.R_s.i_{sq}+L_s.L_rP.w_r.i_{rd}-L_r.R_r.i_{rq}\right)\\ \frac{1}{J}\left(P.M.i_{sq}.i_{rd}-P.M.i_{sd}i_{rq}-fv.w_r-C_r\right)\end{pmatrix} \quad (A.31)$$

$$[B]=\begin{bmatrix}\xi L_r & 0\\ 0 & \xi L_r\\ -\xi M & 0\\ 0 & -\xi M\\ 0 & 0\end{bmatrix},\ [C]=\begin{pmatrix}1&0&0&0&0\\0&1&0&0&0\\0&0&0&0&1\end{pmatrix} \quad (A.32)$$

Avec : $\xi = 1/(L_r.L_s - M^2)$

Les paramètres de la machine asynchrone utilisée en simulation :

GRANDEURS NOMINALES :

Puissances nominales : $P = 4\ kw$.

Nombre de paires de pôles $p = 2$.

Tension efficace nominale : $V = 230V\ /380V$.

Fréquence nominale : $f = 50\ Hz$.

Courant efficace nominal : $In = 15A\ /12A$.

Vitesse de rotation nominale : $\Omega\ = 1500\ tr/mn$.

PARAMETRES ELECTRIQUES :

Résistance statorique $Rs\ =\ 1.2\ (\Omega)$.

Résistance rotorique $Rr\ =\ 1.8\ (\Omega)$.

Inductance statorique $Ls\ =\ 0.1554\ (mH)$.

Inductance rotorique $Lr\ =\ 0.1568\ (mH)$.

Inductance mutuelle $M\ =\ 0.15\ (mH)$.

PARAMETRES MECANIQUES :

Inertie $J\ =\ 0.07\ (kg.m)$.

Frottement visqueux $fv\ =\ 0.0001\ (N.m.s/rad)$.

Références bibliographiques

[1] J. RICHALEL, «Modélisation et identification des processus, Technique d'ingénieur », 1983.

[2] L. Ljuing, «System Identification: Theory for the User», Englewood Cliffs, NJ, 1987.

[3] M. Kunt, M. Bellanger, «Techniques modernes de traitement numérique des signaux », collection électricité, traitement de l'information, Volume1, 1991.

[4] D.Landau, «Identification et commande des systèmes », Hermes, Paris, 1998.

[5] D. Urbani, «Méthodes statistiques de sélection d'architectures neuronales : application à la conception de modèles de processus dynamiques », Master's thesis, Université Paris 6, 1995.

[6] T. Cibas, F.F. Soulié, P. Gallinari and S. Raudys, «Variable selection with neural networks», NEUROCOMPUTING, 12 :223–248, 1996.

[7]G. Dreyfus, «Réseaux de Neurones - Méthodologie et applications », Eyrolles, 2002.

[8] K.Narendra, P.Gallman , «An iterative method for the identification of nonlinear systems using a Hammerstein model», IEEE Transactions on Automatic Control, 11(3):546-550, 1966.

[9] R.Haber, H.Unbehauen, «Structure identification of nonlinear dynamic systems Asurvey on input/output approaches», Automatica, 26(4):651-677, 1990.

[10] J.Ragot, D.Mielcarek, «Recursive identification of multivariaable interconnected systems», International Journal of Science, 23(6):987-1000, 1992.

[11] R. Pearson, M.Pottman, «Gray-box identification of block-oriented nonlinear models», Journal of Process control, 10(4): 301-315, 2000.

[12] D.Filev, «Fuzzy modeling of complex systems», International Journal of Approximate Reasoning, 5(3):281-290. 1991.

[13] Y. HARKOUSS, «Application de réseaux de neurones à la modélisation de composants et de dispositifs micro-ondes non linéaires», Thèse de doctorat de l'Université de Limoges, décembre 1998.

[14] M.Stone, «An asymptotic equivalence of choice of model by cross-validation and akaike's criterion». J. R. Stat. Soc, 38:44–47, 1977.

[15] L. Breiman, «Heuristics of instability and stabilization in model selection», Annals of

Statistics, 24 :2350–2383, 1996.

[16] B. Efron , R.J.Tibshirani, «Improvements on cross-validation »: The .632+ bootstrap method. J. of the American Statistical Association, 92 :548–560, 1997.

[17] T.A.Johansen, «Robust identification of takagi-sugeno-kang fuzzy models using regularization», In IEEE Conf, Fuzzy Systems, New Orleans, 1996.

[18] T.A.Johansen, «On tikhonov regularization, bias and variance in nonlinear system identification», Automatica, 33 :441–446, 1997.

[19] M. Parizeau, «Réseaux de neurones», LAVAL, 2004.

[20] B.Widrow, al. , «Adaptive Noise canceling: Principles and Applications», Proc. IEEE, 63, N° 12, Décembre 1975.

[21] R.Ben Abdennour, P.Borne, «Identification et commande numérique des procédés industriels», Avril 2001.

[22] D.Hebb, «The organization of behavior», Wiley, New York, 1949.

[23] F.Rosenblatt, «The perceptron : A perceiving and recognizing Automaton», Technical report, Cornell Aeronautical Lab, 1957.

[24] M.L. Minsky, S.A. Papert, «PERCEPTRONS», Cambridge, MIT Press, 1969.

[25] J. J. Hopfield, «Neural networks and physical systems with emergent collective computational abilities», Proc. Nat. Acad. Sci., 2554–2558, 1982.

[26] J. Moody, C.J.Darken, «Fast learning in networks of locally-tuned processing units, Neural Computation», 281–294, 1989.

[27] M.T. Evangelia, «Supervised and unsupervised pattern recoganition feature extraction and computational intelligence», CRC Press LLC, 2000.

[28] P.J.W.Melsa, «Neural networks: A conceptual overview», Technical Report, Mishawaka, Aug 1989.

[29] E.Davalo, P.Naim, «Des réseaux de neurones», 2eme édition, Eyrolles 1990.

[30] P.K. Simpson, «Artificial neural systems», Pregramon press, Elmsford, New York, 1989.

[31] B.Widrow, M. A. Lehr, «30 Years of adaptive neural Networks: Perceptron, Madaline and Backpropagation», Proceedings of the IEEE, vol 78, no 9, pp 1415-1441, Sep. 1990.

[32] D. Tank, J. Hopfield, «Les réseaux de neurones formels», Pour la science, no 123, Fev 1988.

[33] D.R.Hush, B. G. Horne, «Progress in supervised Neural Networks», IEEE Signal P

rocessing Magazine, vol. 10, no. 1, pp 8-39, Jan1993.

[34] K.Kara, «Application des réseaux de neurones à l'identification des systèmes non linéaires», Thèse de Magister, Université de Constantine 1995.

[35] B.Kosko, «Neural Networks and Fuzzy systems», Prentice-Hall, 1992.

[36] T.Kohonen, «Self organization and associative memory», 2eme edition, Springer Verlag Berlin Heidelberg, 1988.

[37] A.Blum, «Neural Networks in C^{++}», Wiley, New York , 1992.

[38] B.Kosko, «Unsupervised Learning in Noise», IEEE Trans, Neural Networks, vol. 1, no.1, pp 44-57, Mars 1990.

[39] R.S.Sutton, A.G.Bareo, R.J.Williams, «Reinforcement Learning is direct adaptive optimal control», IEEE Control Systems Magazine, vol. 12, no. 3, pp 49-22, Apr1992.

[40] H.R.Berenji, P.Khedkar, «Learning and Tuning Fuzzy Logic controllers Though Reinforcements », IEEE Trans. Neural Networks, vol, 3, no. 5, pp 724-740, Sep1992.

[41] R.P.Lippman, «An introduction to computing with neural networks», IEEE ASSP Magazine, vol.4, no. 2, pp4-22, Apr 1987.

[42] K.Knight, «connectionist udeas and algorithms», Communications of the ACM, vol. 33, no. 11, pp 59-74, Nov1990.

[43] E.Davalo, P.Naim , «Des Réseaux de Neurones», Edition Eyrolles, 1993.

[44] K.S.Narendra, K. Parthasaraty, «Gradient methods for the optimization of systems con taining neural networks», IEEE Trans, Neural Networks, vol. 2, no. 2n pp252-262. Mars1991.

[45] H.A.Malki, A.Moghaddamjoo, «Uising the Karhunen-Loeve transformation in the backoropagation training algorithme», IEEE Trans. Neural Networks, vol.2, no.1, Jan1991.

[46] T.Yamada, T.Yabuta, «Neural network controller using autotuning method for nonlinear functions», IEEE Trans. Neural Networks vol.3, no.4, pp 595-601, Jul 1992.

[47] M.R.Azimi-Sadjadi, S.Sheedvash, F. O. Trujillo, «Recursive dynamic nod creation in multilayer neural networks», IEEE Trans Neural Networks, vol. 4, no.2, pp242-256, Mars1993.

[48] J.A.Leonard, M.A.Kramer, L.H.Ungar, «Using radial basis functions to approximate a function and its error bounds», IEE Trans. Neural Networks, vol. 3, no. 4, pp 624-627, Jul 1992.

[49] S.Chen, C.N.Cowan, P.M.Grant, «Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks», IEEE Trans. Neural Networks, vol. 2, no. 2, pp302-309,

Mars1991.

[50] K.S.Narendra, «Neural networks for identification and control», Center for Systems Science,Yale University, Dec1998.

[51] S.Chen, S.A.Billings, «Neural Networks for nonlinear system modeling and Identification», *Int. J. Contro*l, vol. 56, no. 2, pp 319-346, Aug1992.

[52] K.S.Narendra, K. Parthasaraty, «Identification and control of dynamical systems using neural networks», IEEE Trans. Neural Networks, vol. 1, no. 1, pp 4-27, Mars1990.

[53] K.J.Hunt, D.Sbarbro, R.Zbikowski, P.J.Gawthrop, «Neural Networks for control systems-A survey», Automatica, vol. 28, no. 6, pp 1083-1112, Dec 1992.

[54] K.Kara, M.L.HADJILI, K.BENMAHAMMED, «Modélisation neuronale des systèmes non linéaires en présence des perturbations».

[55] J.Sjöberg, al, «Nonlinear black-box modelling in system identification: a unified overview» , *Automatica*, 31(12), 16911724, 1995.

[56] M.Norgaard, O.RaVn, N.K.Poulsen, L.K.Hansen, «Neural Networks for Modeling and Control of Dynamic Systems», Spriger-Verlag, London 2000.

[57] J.Jairo, E.Oviedo, «Fuzzy Modeling and Control», Thèse de Doctorat, Katholieke Universiteit Leuven, Belgique, april 2001.

[58] X.Yao, «Evolvin Artificial Neural Networks», *Procedings of the IEEE*, 87(9), 1423-1447, 1999.

[59] G. Dreyfus, « Réseaux de Neurones - Méthodologie et applications », Eyrolles, 2002.

[60] H. YANTOUR, J. SAADI, A. KHOUMSI, 6ème Conférence Francophone de Modélisation et Simulation MOSIM06, « Modélisation et Simulation d'une Commande directe du couple appliquée à la Machine Asynchrone » Rabat-Maroc, Avril2006.

[61]: BENNOUI HASSINA, «Apport de la logique floue et des réseaux de neurones pour la commande avec minimisation des pertes de la machine asynchrone ». Université de Batna.

[62] : THESE BENAISSA MALIKA, « Minimisation des pulsations du couple dans une commande directe du couple « DTC » d'une machine asynchrone» .Université de Batna.

[63] : DISSA ABDENNOUR, « Contrôle direct du couple du moteur à induction sans capteur de vitesse associée à un observateur non linéaire », Université de Batna.

[64] : F.NACERI, « Commande non linéaire adaptative des machines électriques associées à des convertisseurs statiques » , projet « J0201320070006 ».

Résumé:

L'identification des systèmes non linéaires par réseaux de neurones a fait l'objet de nombreux travaux de recherche depuis une trentaine d'années à cause de la capacité d'apprentissage, d'approximation et de généralisation que possèdent ces réseaux .En effet, cette nouvelle approche fournit une solution efficace à travers laquelle de larges classes des systèmes non linéaires peuvent être modélisés sans une description mathématique précise.

Ce travail consiste en l'exploitation des réseaux de neurone artificiels (RNA) à l'identification des systèmes non linéaires. En effet, les RNA eux-mêmes sont des structures à modèles Boite-noire non linéaires. Le but visé par ce travail c'est de voir l'effet du choix de l'architecture d'un RNA sur la procédure d'identification voire la validation du modèle estimé. Plusieurs modèles non linéaires existent dans la littérature tels que : NAR, NARMA, NARX, NARMAX, etc…

Dans ce travail, nous avons appliqué la technique d'identification par réseaux de neurones sur une machine asynchrone (la machine asynchrone est un système dynamique non linéaire et multi-variables). Pour cela, en utilisant deux architectures différentes des réseaux de neurones (perceptron multicouche (MLP) et fonctions de base radiale (RBF)) basent sur la structure NARX.

Mots clés : identification, systèmes non linéaires, réseaux de neurones, MLP, RBF, NARX ,machine asynchrone.

ملخص:

تحديد أنظمة غير الخطية باستخدام الشبكات العصبية موضوع الكثير من الأبحاث على مدى السنوات الثلاثين الماضية ويرجع ذلك إلى قدرة التعلم، وتقريب والتعميم لدى هذه الشبكات. هذا النهج الحديث يوفر حل فعال،من خلاله مجالات كثيرة من أنظمة غير الخطية يمكن أن تحدد بدون وصف رياضي دقيق.

ويشمل هذا العمل استخدام الشبكات العصبية الاصطناعية لتحديد النظم غير الخطية. في الواقع، الشبكات العصبية أنفسهم الهياكل في مربع نماذج السوداء غير الخطية. والهدف من هذا العمل هو أن نرى تأثير على اختيار الهندسة على إجراءات تحديد الهوية أو التحقق من صحة النموذج المقدر. وتوجد عدة نماذج غير الخطية.

في هذا العمل، طبقنا هذه التقنية لتحديد الشبكات العصبية على جهاز غير متزامن (المحرك التعريفي هو نظام ديناميكي غير الخطية والمتغيرات المتعددة). لهذا الغرض، وذلك يستند إلى بنية

Abstract:

Identification of nonlinear systems using neural networks has been the subject of much research over the last thirty years ago to the ability of learning, approximation and generalization have these networks. Indeed, this new approach provides an efficient solution through which large classes of nonlinear systems can be modeled without a precise mathematical description.

This work consists of the exploitation of the artificial networks of neuron (RNA) to the identification of the nonlinear systems. Indeed, the RNA themselves are structures with nonlinear models Limp-black. The aim set by this work is to see the effect of the choice of the architecture of a RNA on the procedure of identification even the validation of the estimated model. Several nonlinear models exist in the literature such as: NAR, NARMA, NARX, NARMAX, etc…

In this work, we applied the technique of identification neural networks on a asynchronous machine (asynchronous machine is a non-linear and multi-variable dynamic system). To do this, using two different architectures of neural networks (multilayer perceptron (MLP) and radial basis function (RBF)) base on structure NARX.

Keywords: identification, nonlinear systems, neural networks, MLP, RBF, NARX, asynchronous machine.