MOHAMED BARKAOUI

ÉTUDE ET DÉVELOPPEMENT D'UNE COMMANDE NEURONIQUE ADAPTATIVE

Mémoire
présenté
à la faculté des études supérieures
de l'Université Laval
pour l'obtention
du grade de Maître ès Sciences (M. Sc.)

Département d'Informatique
FACULTÉ DES SCIENCES ET DE GÉNIE
UNIVERSITÉ LAVAL

Mars 1998

Je dédie ce mémoire à mes parents, à mes soeurs et à mes frères. Je vous aime très fort et je pense à vous.

REMERCIEMENTS

Ce projet de recherche n'aurait pu être réalisé sans la collaboration de plusieurs personnes, soit par leurs conseils ou leurs encouragements. J'aimerais remercier mon directeur de recherche, le professeur J. M. Beaulieu du département d'informatique, pour la grande disponibilité et les conseils avisés qu'il m'a accordés. Je voudrais aussi remercier, mon codirecteur de recherche, le professeur M. Guillot du département de génie mécanique, pour m'avoir guidé dans la réalisation de ce travail, pour son support, son aide, ses bons conseils et sa patience.

Je remercie M. Pierre Marchand, Professeur au département d'informatique, d'avoir aimablement accepté de se pencher sur l'évaluation de ce mémoire en tant qu'examinateur.

Un remerciement au Dr. R. Azouzi, qui par la grande disponibilité et les conseils avisés qu'il m'a accordés, a été une aide des plus précieuses dans la réussite de ce projet. J'aimerais remercier l'ensemble des techniciens de l'atelier de mécanique du pavillon Pouliot de l'Université Laval, qui m'ont apporté un soutien technique de qualité. Un gros merci en particulier à M. Y. Jean ingénieur électrique du département de génie mécanique, pour son aide et ses conseils.

Je n'oublierais pas de remercier, tout particulièrement, le Gouvernement Tunisien ainsi que l'Agence Canadienne de Développement International (ACDI) pour leur support financier.

RÉSUMÉ

L'objectif de ce projet de recherche est d'améliorer l'adaptation en continu d'un réseau perceptron multicouche et d'utiliser l'approche neuronique pour le contrôle de la température dans le laboratoire de métrologie industrielle (LMI) du département de génie mécanique de l'Université Laval.

L'issue majeur de ce mémoire est le développement d'un algorithme d'apprentissage pour un réseau perceptron multicouche. Il en résulte une organisation de l'enregistrement des connaissances dans le réseau similaire à celle présente dans le réseau neuronique biologique. Ceci permet de réduire le risque de dégradation des informations préenregistrées dans le réseau lorsque de nouvelles informations lui sont fournies.

La seconde phase du projet était l'implantation d'une approche de neurocommande adaptative afin de contrôler la température dans le LMI. Les résultats obtenus ont permis de montrer que, non seulement l'approche de neurocommande adaptative était supérieur au système de contrôle à base de PID (Proportional, integral and derivative), mais aussi, qu'elle possède d'excellentes capacités d'adaptation qui sont renforcées par de remarquables propriétés de stabilité.

Québec, Mars 1998

TABLE DES MATIÈRES

Table des matières vi

Liste des figures ix

Liste des tableaux x

1. INTRODUCTION GÉNÉRALE 1
1.1. PROBLÉMATIQUE ET OBJECTIFS 3
1.1.1. Problématique 4
1.1.2. Objectifs 4

2. REVUE DE LA LITTERATURE 6
2.1. INTRODUCTION 6
2.2. RÉSEAUX NEURONIQUES 7
2.2.1. Description des réseaux neuroniques 7
2.2.2.Fonctionnement des réseaux 12
2.2.3. Classification et caractéristiques des réseaux 13
2.2.4. Le réseau perceptron multicouche 15
2.2.4.1. Étude des méthodes d'apprentissage 16
2.2.4.1.1. La méthode standard de rétropropagation 18
2.2.4.1.2. La méthode de descente du gradient 19
2.2.4.1.2.1. L'algorithme de recherche du gain optimal 20
2.2.4.1.3. La méthode quasi-Newton 21
2.2.1.4.4. Comparaison et discussion 22
2.2.5. Le réseau de Kohonen 23

2.3. AVANTAGES ET LIMITES DES RÉSEAUX NEURONIQUES 26
2.3.1. Avantages des réseaux neuroniques 27
2.3.2. Limites des réseaux neuroniques 28
2.4. CONCLUSION 28

3. COMMANDE DE PROCÉDÉ 29
3.1. INTRODUCTION 29
3.2. REVUE DE LA NEUROCOMMANDE 29
3.2.1. Approches de neurocommande 29
3.2.1.1. Neurocommande directe 29
3.2.1.2. Neurocommande indirecte 32
3.2.2. Les techniques d'adaptation 33
3.3. RÉGULATEUR NEURONAL 38
3.3.1. Structure de commande avec le modèle neuronal inverse 38
3.4. CONCLUSION 41

4. RÉSEAU PERCEPTRON CLUSTERISÉ 42
4.1. INTRODUCTION 42
4.2. ÉTUDE DE L'ORGANISATION, DE L'APPRENTISSAGE ET DE LA MÉMORISATION DANS LE CERVEAU BIOLOGIQUE 43
4.2.1. L'organisation du cerveau 43
4.2.2. Les mécanismes physiologiques de l'apprentissage et de la mémorisation dans le cerveau 51
4.3. RÉSEAU PERCEPTRON CLUSTERISÉ 52
4.3.1. Algorithme d'apprentissage du réseau perceptron clusterisé 54
4.3.2. Entraînement du réseau perceptron clusterisé 59
4.3.2.1. Entraînement hors-ligne du perceptron clusterisé 59
4.3.2.2. Entraînement continu du réseau perceptron clusterisé 59
4.3.3. Étude des performances du réseau perceptron clusterisé et comparaison avec le réseau standard 60

4.3.3.1. Tests d'entraînement a priori 60
4.3.3.2. Étude de la clusterisation 63
4.3.3.3. Tests d'entraînement continu 64
4.4. CONCLUSION 66

5. COMMANDE DE TEMPÉRATURE 67
5.1. INTRODUCTION 67
5.2. LE LABORATOIRE LMI 69
5.3. APPAREILLAGE 71
5.3.1. Les senseurs 75
5.3.2. Les actionneurs 76
5.3.3. L'organe de commande 76
5.3.4. L'étalonnage 77
5.4. RÉSULTATS EXPÉRIMENTAUX 78
5.4.1. Entraînement du réseau neuronique 78
5.4.2. Comparaison avec le régulateur à base de PID 80
5.5. CONCLUSION 83

6. CONCLUSION GÉNÉRALE 84

RÉFÉRENCES 87

ANNEXE 91

LISTE DES FIGURES

2.1. Composantes d'un réseau de neurones. 9
2.2. Fonctions d'activation typiques 11
2.3. Les types de réseaux neuroniques les plus importants 13
2.4. Perceptron multicouche typique. 16
2.5. Le réseau de Kohonen 26
3.1. (a) Schéma de neurocommande directe, (b) neurocommande supervisée et (c) neurocommande par modèle inverse 31
3.2. (a) schéma explicite pour la régulation du procédé et (b) modèle neuronique du procédé. 32
3.3. Un système typique de neurocommande directe adaptative 37
3.4. Structure de commande ayant comme régulateur le modèle neuronique inverse 39
3.5. Structure de commande prédictive avec adaptation des modèles neuronaux. 40
4.1. Représentations corticales des différentes parties du corps (d'après [27]) 45
4.2. Structure du cortex cérébrale (d'après [27]) 46
4.3. Aires cytoarchitecturales du cerveau humain (d'après [27]). 48
4.4. Nouvelle architecture du réseau perceptron 53
4.5. (a) Le coefficient de clusterisation spatiale, (b) le coefficient de clusterisation par les données, (c) le coefficient de clusterisation pour l'entraînement du réseau perceptron clusterisé 58
4.6. Représentation graphique de la fonction E(x). 60
5.1. Laboratoire de métrologie industrielle du département génie mécanique de l'Université Laval 70
5.2. Dispositif expérimental 72
5.3. Liens entre LabVIEW, NI-DAQ et la carte d'acquisition 73
5.4. (a) Le système de neurocommande pour le contrôle de la température, (b) schéma de neurocommande adaptative. 74
5.5. Pourcentage d'utilisation de la puissance des câbles chauffants en fonction du voltage d'entrée. 76

LISTE DES TABLEAUX

4.1. Données d'entraînement a priori 61
4.2. Données d'entraînement en continu 62
4.3. Résultats des entraînements a priori et en continu 62
4.4. Résultat de l'étude de la clusterisation au sein des réseaux 63
4.5. Résultats de SE_C en fonction de SE_0 66
5.1. Les conditions environnementales du LMI (S.O. # 700035) 71
5.2. Les valeurs des coefficients a_i de l'équation (5-2) pour les quatre senseurs 78
5.3. Données d'entraînement a priori du réseau perceptron clusterisé (les températures sont en °C) 79
5.4. Les valeurs des paramètres d'entraînement à priori et en continu du réseau perceptron clusterisé 80
5.5. Les résultats des essais de perturbation avec le système de commande à base de PID (les températures sont en °C) 81
5.6. Les résultats des essais de perturbation avec le régulateur neuronal (les températures sont en °C) 82

CHAPITRE I

INTRODUCTION GÉNÉRALE

Au cours des cinq dernières décennies, une nouvelle science aux applications variées a commencer à se développer, soit l'intelligence artificielle (I.A.). Particulièrement dans le domaine de la robotique, son implication consiste à donner à ces machines la capacité d'assumer des fonctions cognitives et de s'auto-administrer en présence de certaines situations. C'est pour cela que de façon indispensable, elle s'attelle à comprendre dans un premier temps la nature de l'intelligence humaine, puis ensuite à la mimiquer par l'élaboration de programmes d'ordinateur dont pourrait découler l'intelligence.

Les activités du corps humain se divisent en deux groupes principaux: celles qui relèvent du mental, de l'esprit, dites cognitives et celles qui relèvent du physique, dites motrices. L'ensemble des fonctions mentales qui entrent en ligne de compte dans l'accomplissement des activités cognitives est appelé intelligence. Elle se rapporte à l'acquisition de la connaissance, à la mémorisation, au langage, à la conception ou discernement des solutions aux problèmes, à la capacité de s'adapter à des nouvelles situations, à la représentation du monde à travers des symboles, etc. Comme cela paraît évident, lorsqu'on parle d'intelligence, la principale partie du corps qui est à l'honneur n'est autre que le cerveau. C'est lui qui se préoccupe de la gestion de toutes les activités du corps humain. Il reçoit les informations en provenance de nos sens à travers les nerfs, les analyse, les mémorise au besoin, fait des raisonnements et des programmes, prend

les décisions qu'il juge opportunes avant d'ordonner aux autres parties du corps d'exécuter tels mouvements en vue de l'atteinte d'un objectif. Tout ce qui se passe dans le corps humain se fait sous son autorité.

L'intelligence artificielle en ingénierie de la production est orienté vers le développement de machines dotées de sens, intelligentes, autonomes, et dont on attend une capacité d'agir et de reproduire à volonté certains comportements de l'humain, par auto-contrôle ou sous instruction de celui-ci. L'objectif principal visé est d'accroître la flexibilité des moyens de production industriels afin d'améliorer la qualité des produits et des services. En d'autre termes, on veut créer des êtres artificiels intelligents à l'image de l'homme qui peuvent apprendre et mémoriser des événements, analyser des situations, prendre des décisions, résoudre des problèmes divers, etc., en tenant compte de certaines données de leur environnement. Ce rêve qui ne date pas d'aujourd'hui a connu ses débuts de matérialisation avec la naissance des robots. Ces derniers, bien que sur la voie de concrétiser un tel rêve, sont encore déficients à quelques niveaux. Même s'ils arrivent à accomplir certaines tâches (ceci dans des limites bien connues), le problème d'intelligence et d'autonomie est encore loin d'être résolu.

Pour la reproduction des fonctions du cerveau, plusieurs approches ont été développées. Celles des systèmes experts s'appuie sur des mécanismes de raisonnement et de déduction de l'intelligence humaine (sans tenir compte d'une quelconque physiologie biologique) pour concevoir des logiciels qui semblent donner de l'esprit aux machines. Ces systèmes regroupent toutes les informations relatives à l'utilisation des machines dans un domaine particulier, les organisent de telle façon qu'elles permettent à ces machines d'analyser des situations, de solutionner des problèmes et d'agir mieux que ne le ferait un expert du domaine. A titre d'exemple, mentionnons la prise de décision, les choix de solutions, la commande intelligente et l'optimisation de certains procédés complexes, le diagnostic de pannes en maintenance industrielle, la gestion financière et le diagnostic de maladies en médecine. Les systèmes experts ont fait leurs preuves mais montrent des faiblesses à certains niveaux.

Outre les systèmes experts, une nouvelle technique de l'intelligence artificielle est en train de gagner du terrain. C'est la technique dite des réseaux neuroniques. Il s'agit de modèles qui imitent exactement le cerveau à travers des logiciels supposés générer de l'intelligence comme chez l'humain. Étant donné que l'influx nerveux est un signal électrique qui se propage dans un réseau de neurones chez l'être humain, l'idée est de créer un réseau de neurones qui sont interconnectés comme les neurones biologiques. Ces neurones artificiels s'avèrent performants pour la duplication de ce qui se passe dans le cerveau. On peut alors reproduire avec ces logiciels de type réseaux neuroniques toute une foule d'activités du cerveau. Cette approche n'est pas loin de son heure de gloire, car les premiers résultats sont encourageants et on remarque que les embryons d'une vraie intelligence sont en train de germer peu à peu.

1.1. PROBLÉMATIQUE ET OBJECTIFS

Compte tenu de leur implication dans des domaines très variés, les réseaux neuroniques ont connu un développement considérable ces dernières années, tant au niveau algorithmique qu'au niveau des applications. Leur flexibilité leur a valu un grand attrait pour les chercheurs de divers domaines telles que l'ingénierie, les mathématiques, la chimie, la gestion industrielle, la neurobiologie, la neuropsychologie, etc.

En ingénierie, les applications des réseaux neuroniques sont très variées. Que ce soit en génie mécanique, en génie civil, en génie électrique, en génie chimique et autres, on peut citer plusieurs exemples d'applications des réseaux neuroniques. Par exemple, en fabrication mécanique, des chercheurs comme le professeur Michel Guillot et le Dr. Riadh Azouzi ont développé un système de commande adaptative à partir du perceptron multicouche et du réseau de Kohonen décrits au chapitre II pour optimiser les défauts géométriques et améliorer la qualité d'usinage [15]. En vision numérique, pour la reconnaissance des formes et des objets, les réseaux neuroniques permettent aujourd'hui à certaines machines de recueillir des informations pertinentes sur leur environnement, ou même d'améliorer la qualité des images. En sciences et technologie des aliments, là encore leur application porte sur le contrôle de la qualité des aliments en terme de dosage des composantes.

1.1.1. Problématique

Du point de vue algorithmique, plusieurs développements ont suivi ceux des pionniers comme Warren McCulloch et Walter Pitts, Frank Rosenblatt, Bernard Widrow, Hopfield, et Kohonen, mais la tendance actuelle est de s'imprégner davantage des connaissances neurobiologiques pour modéliser l'apprentissage et la mémorisation au niveau du cerveau humain.

Sans doute, le réseau le plus important et le plus populaire est le perceptron multicouche introduit en 1958 par Rosenblatt [4]. Le réseau perceptron multicouche est un outil prometteur surtout quand il s'agit de résoudre des problèmes non-linéaires. Toutefois, le problème de dégradation des informations au niveau de la mémoire du réseau, lorsque ce dernier est adapté en continu, constitue un sérieux danger pour la stabilité et la performance des systèmes de neurocommande (les systèmes utilisant un contrôleur neuronal). Ce problème est un handicap majeur pour le développement de stratégies efficaces de neurocommande adaptative. L'étude de la neurocommande fera l'objet du chapitre III

1.1.2. Objectifs

Pour surmonter les difficultés citées ci-dessus, nos réflexions nous ont amenés à nous poser la question de savoir comment le cerveau humain procède. Ce qui nous diffère des autres êtres est notre capacité d'apprentissage des choses, de les mémoriser, de faire de l'analyse et du raisonnement, et de prendre des décisions avant d'agir. Suite à l'étude anatomique et physiologique du cerveau, nous expliciterons le processus selon lequel l'apprentissage et la mémorisation se produisent chez l'humain. Cela nous conduira à développer un nouvel algorithme d'apprentissage, en se basant sur une nouvelle architecture du réseau, qui permet d'améliorer les capacités d'apprentissage en continu du réseau perceptron multicouche. Cette partie fera l'objet du chapitre IV.

Une deuxième étape dans notre mémoire consistera à étudier et développer un régulateur neuronal à base d'un réseau possédant des bonnes capacités d'adaptation. Le principale objectif de cette étude est d'effectuer la commande de contrôle de la température du laboratoire de métrologie industrielle (LMI) du département de génie mécanique de l'Université Laval et d'évaluer les performances d'un tel régulateur. Cette étude fera l'objet du chapitre V.

Dans le chapitre suivant, nous introduirons les différents concepts des réseaux neuroniques. Nous parlerons du perceptron multicouche que nous aurons à utiliser pour développer notre algorithme, et du réseau de Kohonen.

CHAPITRE II

REVUE DE LA LITTÉRATURE

2.1. INTRODUCTION

Depuis très longtemps, l'homme croit que l'étude des réseaux biologiques et le design de réseaux neuroniques artificiels pourraient apporter de nouvelles façons d'aborder les problèmes courants et ainsi conduire à des améliorations algorithmiques et méthodologiques. En effet, les travaux de recherche concernant les réseaux neuroniques artificiels ont une longue histoire. Le développement de modèles de réseaux neuroniques détaillés a débuté il y a plus de quatre décennies avec les travaux de McCulloch et Pitts [1], Hehh [2], Rosenblatt [3], etc. Plusieurs modèles différents ont surgi. Aujourd'hui, le terme *réseaux neuroniques* regroupe un certain nombre de types de réseaux différents. Le cerveau biologique excelle dans des tâches de commande des opérations, de reconnaissance de forme, de raisonnement intuitif, etc. Le cerveau biologique constitue donc une preuve vivante de la possibilité du développement d'un contrôleur généralisé qui prendrait avantage de la capacité d'apprentissage des réseaux neuroniques.

D'un point de vue systémique, un réseau neuronique est un modèle connexionniste constitué de simples fonctions neuroniques non-linéaires et dont le fonctionnement est inspiré des réseaux neuroniques biologiques. Ces réseaux peuvent modéliser pratiquement toute relation continue. Ils apprennent le comportement physique et dynamique d'un procédé à partir de

l'expérience et en utilisant des algorithmes d'apprentissage et d'adaptation. L'expérience est représentée par des exemples d'entrées/sorties qui sont obtenus à partir de courts essais expérimentaux. En fait, l'avantage principal des réseaux neuroniques réside en leur capacité d'extraire les informations pertinentes à partir d'un nombre relativement limité de données expérimentales. De cette façon, ils compensent pour l'incapacité de comprendre complètement et de décrire adéquatement les mécanismes et les lois qui régissent le procédé. De plus, les réseaux neuroniques ont la capacité de construire des modèles qui représentent très bien la réalité et permettent de réduire considérablement le temps nécessaire pour converger à un point d'opération raisonnablement optimal: une approche qui rassemble à la fois la simplicité relative des approches agrégées et l'avantage théorique des approches de modélisation désagrégées, sans la complexité de ces dernières.

Une revue de la littérature nous a permis de constater que l'utilisation des modèles à apprentissage comme les réseaux neuroniques artificiels, constitue une approche de modélisation robuste et adaptée.

Par conséquent, ce mémoire portera particulièrement sur la technique de modélisation par les réseaux neuroniques. Nous comparerons les méthodes qui ont été proposées pour l'entraînement des réseaux neuroniques en considérant la vitesse d'entraînement et de la qualité des modèles résultants. Ensuite, pour des fins de vérification, la technique de modélisation par réseaux neuroniques sera comparée à la technique de modélisation par régression multiple.

À la section suivante, nous introduisons les réseaux neuroniques.

2.2. RÉSEAUX NEURONIQUES

2.2.1. Description des réseaux neuroniques

Il existe dans la littérature, de nombreux modèles neuroniques présentant des différences notables. Cependant, nous essayerons dans un premier temps de décrire les principales composantes communes à chacun d'eux.

La Figure 2.1a présente les différentes composantes d'un réseau neuronique sous sa forme générale. À gauche, est schématisé un réseau dans sa ressemblance biologique et à droite, son simulateur électronique. Les cercles indiquent les neurones ou processeurs, présentant chacun un niveau d'activation $a_j(t)$ variable dans le temps. Les lignes reliant les points d'entrée et de sortie aux neurones ou les neurones entre eux sont des connexions et permettent la transmission des signaux.

Dans l'approche neuronique, chacun des neurones est modélisé par un signal d'entrée I_j et un signal de sortie O_j (Figure 2.1b). Les signaux de sortie des neurones précédents (O_i) sont pondérés par des coefficients synaptiques w_{ij} ou poids qui déterminent l'influence d'un signal O_i sur la fonction d'entrée neuronique I_j de la forme:

$$I_j = net_i = \sum_j w_{i,j} O_i \qquad (2\text{-}1)$$

L'influence de chacun des neurones i précédents est donc additive. Les coefficients synaptiques $w_{i,j}$ peuvent varier dans le temps selon l'apprentissage effectué sur le réseau.

Le niveau d'activation $a_j(t)$ du neurone est établi à partir d'une fonction F_j variant selon le niveau d'activation précédant $a_j(t-1)$ et la valeur d'entrée I_j.

$$a_j(t) = F_j(a_j(t-1), I_j) \qquad (2\text{-}2)$$

La valeur de sortie du neurone j sera affectée par une troisième fonction dont la forme générale sera:

$$O_j(t) = f_j(a_j(t)) \qquad (2\text{-}3)$$

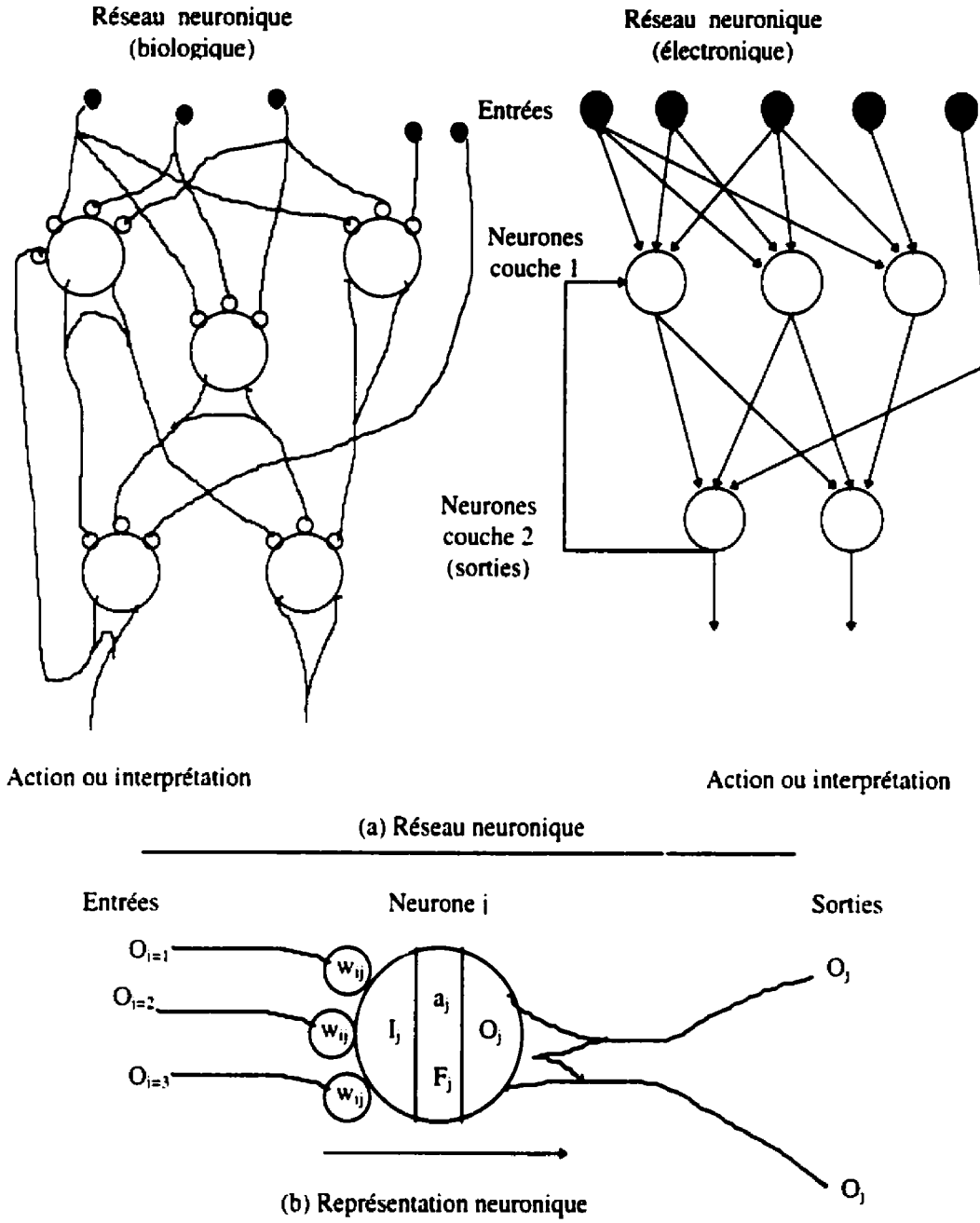


Figure 2.1. Composantes d'un réseau de neurones

Chacun des modèles neuroniques sera donc caractérisé par au moins sept éléments principaux:

1- La représentation des processeurs (neurones). Dans certains modèles de réseaux neuroniques, les neurones peuvent représenter des objets conceptuels, dans d'autres, ils ne serviront que d'unité de traitement et d'emmagasinage des connaissances.

2- Un réseau de connectivités. Comme il est indiqué à la Figure 2.1, le réseau de connectivités relie les neurones entre eux. Dans certains modèles neuroniques, les connexions sont toujours dirigées vers le bas (feedforward), dans d'autres, elles effectueront certaines retroactions (feedbacks), ou encore, seront distribuées de manière à présenter l'information entre concepts. Chacune des connexions sera affectée par un coefficient synaptique , i.e., un poids définissant l'importance de la connexion. Ces coefficients prendront habituellement des valeurs entre 0 et 1. Dans certains cas, on leur attribuera des valeurs positives lorsque leur effet est excitatif, négatives pour un effet inhibitif et nulles lorsqu'il n'y a aucun effet. L'ensemble des coefficients synaptiques d'un réseau de connectivités sera décrit dans la matrice W.

3- Des règles de propagation. Une règle est habituellement décrite par l'équation (2-1) et détermine de quelle façon les connectivités seront associées à l'entrée Ij d'un neurone.

4- L'état d'activation. En général, l'état d'activation prendra des valeurs variant entre 0 et 1, inactif à 0 et actif à 1. Dans le cas des réseaux 'binaires' ou 'booléens', seules les valeurs entières 0 et 1 seront considérées. Pour les réseaux continus, toutes les valeurs réelles comprises entre 0 et 1 seront acceptables. Dans d'autres modèles, l'état d'activation sera délimité par [-1, +1], [-1, 0, +1] ou autres.

5- La règle d'activation F_j. Elle est décrite dans sa forme générale par l'équation (2-2). Le plus souvent, elle sera indépendante de $a_j(t-1)$ et apparaîtra soit sous la forme d'un sigmoïde, d'une fonction linéarisée du sigmoïde, ou d'une fonction binaire (Figure 2.2). La fonction sigmoïdale est définie par:

$$a_j(t) = 1/(1+\exp(-I_j)) \qquad (2\text{-}4)$$

$$a_j(t) = (2/(1+\exp(-I_j))) - 1 \qquad (2\text{-}4')$$

6- La fonction de sortie f. Décrite dans sa forme générale par l'équation (2-3), elle prendra assez souvent cependant, la forme d'une identité:

$$O_j(t) = a_j(t) \qquad (2\text{-}5)$$

ou encore une fonction déportée d'un écart de sortie θ.

7- Une règle d'apprentissage. Dans un réseau neuronique, la connaissance est emmagasinée par le biais des valeurs de connectivités. L'apprentissage ou la modification des connaissances pourra donc se faire de trois façons:

- En modifiant la valeur des coefficients synaptiques
- En ajoutant de nouvelles connexions
- En enlevant certaines connexions

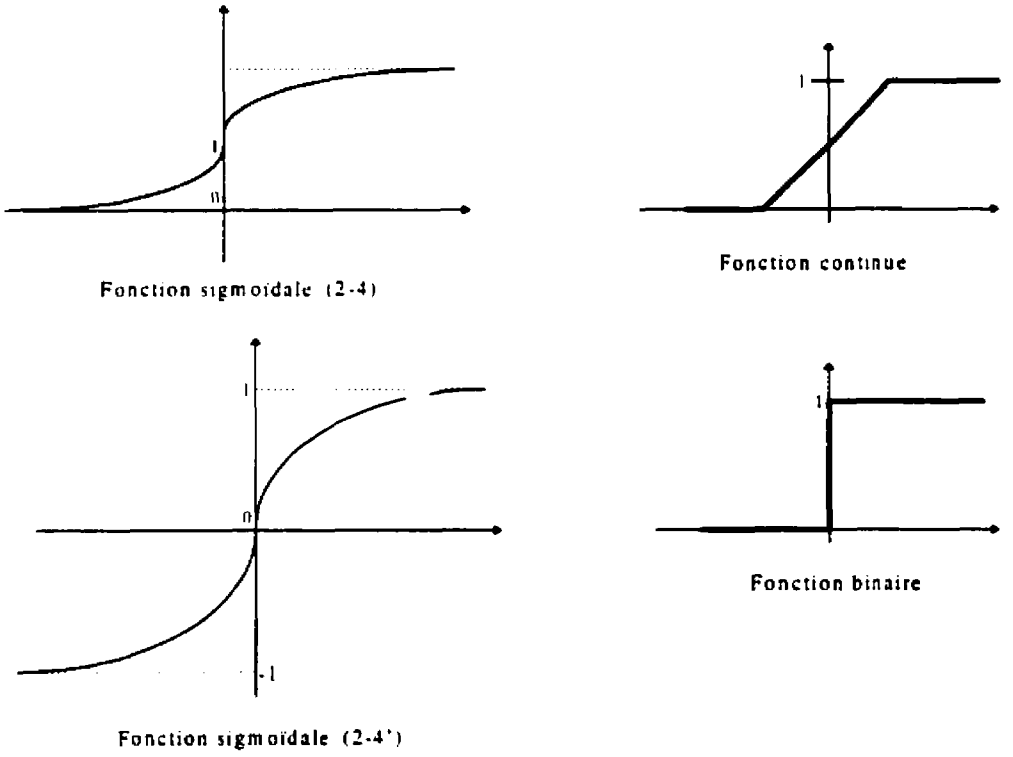


Figure 2.2. Fonctions d'activation typiques

La plupart des règles d'apprentissage existantes ne permettront que d'ajuster les coefficients synaptiques. Cependant, lorsque ces coefficients sont nuls, ceci revient à enlever une connexion. Nous verrons ces règles d'apprentissage plus loin.

2.2.2.Fonctionnement des réseaux

Les réseaux neuroniques artificiels peuvent être implantés de deux façons, soit (1) par logiciel (e.g. nnet, logiciels de la série PDP, etc.) tournant sur un ordinateur conventionnel, ou (2), par plusieurs processeurs électroniques (hardware) travaillant en parallèle dans un système ordinique où chacun des processeurs représente et exécute les fonctions neuronales.

En pratique, l'approche logiciel est la plus couramment utilisée. Les ports d'entrées et de sorties du réseau neuronique peuvent être reliés à l'extérieur au moyen de systèmes d'acquisition de données et de commandes, ou encore, être utilisés à l'intérieur d'un programme principal lui fournissant les valeurs d'entrées et collectant les valeurs de sorties.

Comme pour l'être humain, les réseaux neuroniques nécessitent en général, un certain apprentissage avant de pouvoir appliquer leurs connaissances à différentes tâches. Donc, nous aurons deux étapes spécifiques:

1) L'entraînement du modèle neuronique pour l'apprentissage des coefficients synaptiques et d'autres types de coefficients, et dans certains cas, pour l'addition ou l'enlèvement de connectivités.
2) L'utilisation des réseaux pour calculer les valeurs de sorties correspondant à certaines valeurs d'entrées.

Lors de l'utilisation des réseaux, nous avons à estimer successivement les fonctions d'entrée, d'activation et de sortie pour chaque neurone du réseau, et ce, à partir des valeurs d'entrées, jusqu'aux sorties du réseau. Une autre différence importante entre l'approche logiciel

et celle par matériel, apparaît ici. Alors que les calculs sur chaque neurone se font successivement dans un logiciel, l'approche par matériel, pourra effectuer l'ensemble des calculs simultanément.

Lors de l'apprentissage, nous propagerons l'erreur ε (l'erreur entre les valeurs de sortie attendues et celles estimées par le modèle) à partir des sorties et en remontant vers les entrées.

$$\varepsilon = O_{attendue} - O_{estimée} \qquad (2\text{-}6)$$

Les valeurs attendues sont celles que l'on veut que le modèle apprenne, tandis que les valeurs estimées proviendront du réseau même. Ces dernières seront évaluées comme auparavant, soit en calculant successivement les valeurs des fonctions d'entrée, d'activation et de sortie pour chaque neurone du réseau, et ce, à partir des valeurs d'entrées, jusqu'aux sorties du réseau. La rétropropagation des erreurs se fera successivement pour chaque donnée (entrée vs sortie attendue) jusqu'à ce que l'erreur ε soit acceptable pour l'ensemble des données à apprendre.

2.2.3. Classification et caractéristiques des réseaux

Comme nous l'avions mentionné précédemment, les réseaux se différencient notamment par leurs entrées binaires ou continues ainsi que par leur mode d'apprentissage supervisé ou non-supervisé. La Figure 2.3 présente une classification des principaux réseaux neuroniques.

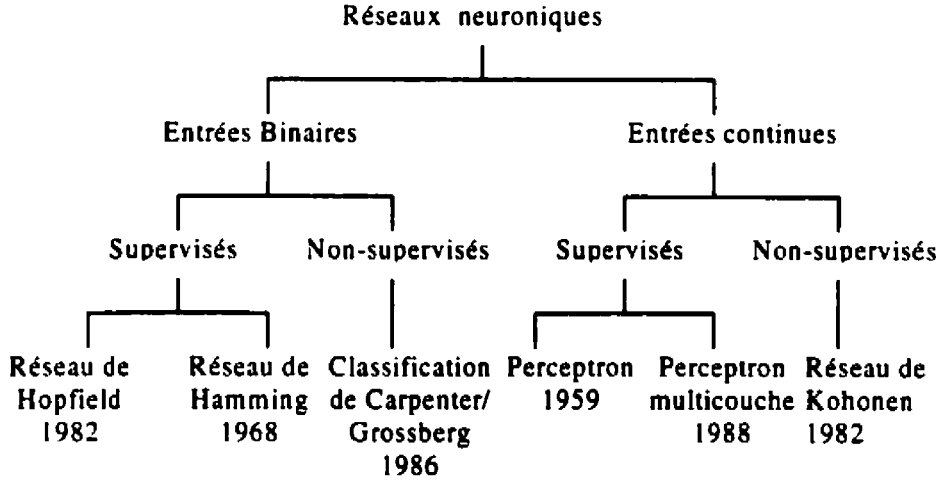


Figure 2.3. Les types de réseaux neuroniques les plus importants

L'apprentissage supervisé permet de configurer les coefficients synaptiques d'un réseau d'une manière telle que l'on puisse éventuellement obtenir les niveaux d'activation désirés aux sorties. Avec les réseaux à apprentissage non-supervisé, nous ne savons pas à l'avance quels seront les niveaux d'activation vers lesquels le réseau se configurera.

La plupart des réseaux de la Figure 2.3 sauf les réseaux perceptrons sont des mémoires associatives, c'est-à-dire des réseaux dont les neurones peuvent alternativement être employés comme entrées ou sorties du réseau. Les mémoires associatives sont habituellement caractérisées par une convergence nécessaire pendant l'utilisation et par un apprentissage sans itération pendant lequel on définit les coefficients synaptiques.

Sans doute, le réseau le plus important et le plus populaire est le perceptron multicouche. Ce réseau apprend par supervision, c'est-à-dire, par présentation d'exemplaires d'entrées/sorties. Il a été introduit en 1958 par Rosenblatt [4]. Ce dernier a montré que le perceptron est capable d'apprendre quelques fonctions logiques en modifiant les poids des connexions synaptiques. En 1969, Minsky et Paper [5] ont montré que le perceptron possède des limitations sévères en ce qui concerne ses capacités de calculs, à cause de la discontinuité de la fonction d'activation de ce modèle neuronique. En 1986, Rumelhart [6] a introduit une fonction continue pour l'activation des neurones qui non seulement n'affecte pas l'utilisation du perceptron pour des applications discrètes, mais aussi, lui confère des capacités illimitées pour la modélisation continue. En effet, Hornik et Stinchombe [7] montrent en utilisant le théorème de Weierstrass, qu'un réseau perceptron multicouche bien dimensionné peut approximer n'importe quelle fonction continue f $\in (R^n, R^m)$.

Aujourd'hui, le perceptron multicouche est employé pour la solution d'une grande diversité de problèmes. A titre d'exemple, citons les applications suivantes: la reconnaissance de formes, le traitement adaptatif des signaux, la modélisation dynamique, la surveillance des procédés, et dans les systèmes experts et les systèmes à logique floue. D'autres applications spécifiques incluent la commande du bras d'un robot, le diagnostic, et la conversion symbolique/numérique. Les réseaux perceptrons multicouche semblent être prometteurs surtout

quand il s'agit de résoudre des problèmes non-linéaires et complexes. Finalement, les calculs parallèles qui y sont effectués et la possibilité de le réaliser sur un circuit intégré rendent ce réseau encore plus prometteur pour son exploitation dans d'autres domaines.

Tout au long de cette étude, notre intérêt portera particulièrement sur le perceptron multicouche. Nous comparerons les méthodes qui ont été proposées pour l'entraînement de ce dernier au niveau de la vitesse d'entraînement et de la qualité des modèles résultants. Toutefois, une étude du réseau de Kohonen semble aussi nécessaire pour la compréhension de l'algorithme d'apprentissage proposé dans le présent rapport.

2.2.4. Le réseau perceptron multicouche

Un réseau perceptron multicouche est constitué de simples unités de calcul qui sont non-linéaires et interreliées entre elles. Ces unités qui opèrent parallèlement les unes aux autres sont arrangées d'une façon similaire au réseau neuronique biologique (voir la Figure 2.4). En général, les signaux se propagent progressivement à travers le réseau seulement entre les couches adjacentes. Ces signaux sont modifiés par les poids des connexions entre les neurones (aussi appelés coefficients synaptiques). Le signal d'activation (le signal d'entrée) $I_{j,c}$ arrivant au neurone j de la couche c est donné par:

$$I_{j,c} = \sum_{i=1}^{n_{c-1}} w_{i,j,c} O_{i,c-1} + \theta_{j,c} \qquad (2\text{-}7)$$

où,

$O_{i,c-1}$: est la sortie de la $i^{ième}$ neurone sur la couche précédente (couche c-1),

$\theta_{j,c}$: est l'écart associé au neurone j de la couche c

n_{c-1} : est le nombre de neurones sur la couche c-1, et,

$w_{i,j,c}$: est le poids de la connexion reliant le neurone i sur la couche c-1 au neurone j sur la couche c.

La sortie de chaque neurone est une fonction croissante et monotone selon l'entrée $I_{j,c}$. Dans la majorité des cas, la fonction sigmoïdale, Γ (Γ: $R^1 \rightarrow R^1$), est utilisée. Celle-ci simule bien l'activité qui se passe au niveau d'un neurone biologique typique. Elle s'exprime comme suit:

$$O_{j,c} = \Gamma(I_{j,c}) = 1/(1+\exp(-I_{j,c})) \qquad (2\text{-}8)$$

Dans un réseau, l'information est emmagasinée dans les coefficients synaptiques des connexions. Le réseau fonctionne comme suit: les valeurs d'entrée de l'exemplaire sont présentées à la première couche du réseau. Celle-ci ne comporte pas de neurone ayant une fonction d'activation sigmoïdale. En fait, les signaux d'entrée sont transmis intégralement vers la couche suivante. Cependant, ils sont pondérés par les poids des connexions reliant la couche d'entrée à la couche suivante.

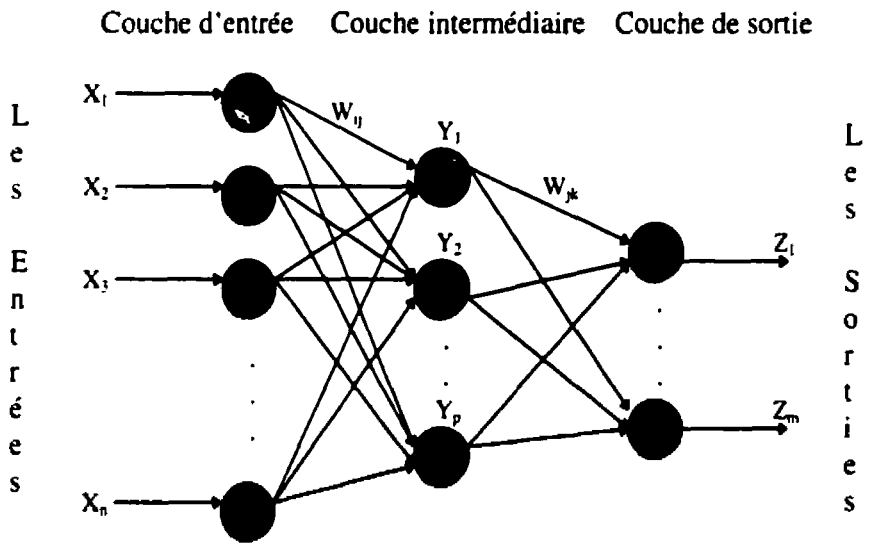


Figure 2.4. Perceptron multicouche typique

2.2.4.1. Étude des méthodes d'apprentissage

Le problème de l'apprentissage ou de l'entraînement d'un réseau neuronique se ramène en réalité à celui de l'estimation des paramètres du réseau et constitue un aspect important de

l'utilisation des réseaux neuroniques. L'ensemble des paramètres qu'il faut estimer inclut tous les poids et les écarts qui se trouvent dans le réseau. Généralement, le nombre de ces paramètres est assez grand.

Les réseaux perceptrons multicouches sont entraînés par supervision, c'est-à-dire par présentation d'exemplaires formés chacun d'une paire d'Entrée/Sortie. Les entrées de chaque exemplaire sont présentées à la couche d'entrée du réseau, et les sorties sont comparées à celles obtenues au niveau de la couche de sortie du réseau. L'objective de la majorité des algorithmes d'apprentissage, consiste alors à ajuster les valeurs des poids et des écarts de telle sorte que SE, la somme des erreurs au carré observées au niveau de toutes les sorties du réseau et avec tous les exemplaires d'entraînement, soit minimisée. L'erreur SE est exprimée comme suit:

$$SE = \sum_{e=1}^{ee} SE_e \qquad (2\text{-}9)$$

où,

$$SE_e = \sum_{i=1}^{n_{cc}} (O_{i,cc} - D_{i,e})^2 \qquad (2\text{-}10)$$

ee : est le nombre d'exemplaires d'entraînement,

n_{cc} : indique le nombre de neurones sur la couche de sortie (couche cc),

$O_{i,cc}$: est la sortie du $i^{ième}$ neurone de la couche de sortie, et

$D_{i,e}$: est la sortie correspondante à la $i^{ième}$ sortie du $e^{ième}$ exemplaire d'entraînement.

Nous présentons maintenant la méthode qui est couramment utilisée pour l'entraînement des réseaux neuroniques, soit la méthode de rétropropagation, et nous la comparons à des méthodes d'optimisation non-linéaire qui peuvent aussi être utilisées pour entraîner un réseau perceptron multicouche. Notre intérêt porte, particulièrement, sur la méthode de la descente du gradient et sur la méthode quasi-Newton.

2.2.4.1.1. La méthode standard de rétropropagation

Généralement, la règle du delta généralisé [6], qui est très souvent appelée méthode de rétropropagation (BP), est automatiquement associée à l'entraînement du perceptron multicouche. Cette règle a été introduite en 1986 par Rumelhart et McClelland [8, 9]. Elle rétropropage les erreurs obtenues au niveau de la couche de sortie à travers le réseau. C'est une méthode basée sur le gradient. A chaque itération, tous les exemplaires d'entraînement sont présentés au réseau à tour de rôle. A chaque présentation d'un exemplaire, l'erreur SE_e est calculée. Ensuite, elle est utilisée pour modifier la valeur de chaque paramètre dans le réseau, comme suit:

$$w_{i,j,c}(k+1) = w_{i,j,c}(k) - \eta\, \partial SE_e/\partial w_{i,j,c} + \alpha[w_{i,j,c}(k) - w_{i,j,c}(k-1)] \qquad (2\text{-}11)$$

$$\theta_{j,c}(k+1) = \theta_{j,c}(k) - \eta\, \partial SE_e/\partial \theta_{j,c} + \alpha[\theta_{j,c}(k) - \theta_{j,c}(k-1)] \qquad (2\text{-}12)$$

Où k indique l'itération en cours. η et α sont les coefficients de gain et de momentum. Leurs valeurs doivent être spécifiées par l'usager. Le gain détermine la grandeur de la variation durant la descente du gradient, alors que le momentum a pour rôle de forcer la variation du poids dans la même direction que celle utilisée lors de la dernière itération (k-1). En plus d'accélérer la convergence de l'algorithme, le momentum doit aussi permettre d'éviter les minimums locaux [9, 10]. Ces deux derniers coefficients sont ajustés empiriquement et prennent des valeurs comprises entre 0 et 1. Les dérivées partielles de SE_e sont obtenues comme suit:

$$\partial SE_e/\partial w_{i,j,c} = -\delta O_{i,c-1} \qquad (2\text{-}13)$$

$$\partial SE_e/\partial \theta_{j,c} = -\delta_{j,c} \qquad (2\text{-}14)$$

où
$$\begin{cases} \delta_{j,c} = (d_{j,e} - O_{j,c}) O_{j,c} (1 - O_{j,c}) & si\ c = cc \\ \delta_{j,c} = O_{j,c} (1 - O_{j,c}) \left[\sum_{i=1}^{\Lambda_{c+1}} \delta_{i,c+1} w_{j,i,c+1} \right] & si\ c < cc \end{cases}$$
si on utilise l'équation (2-4)

$$\begin{cases} \delta_{j,c} = 1/2(d_{j,c} - O_{j,c})(1 - O_{j,c}^2) & si\ c = cc \\ \delta_{j,c} = 1/2(1 - O_{j,c}^2)\left[\sum_{i=1}^{n_{c+1}} \delta_{i,c+1} w_{j,i,c+1}\right] & si\ c < cc \end{cases}$$ si on utilise l'équation (2-4')

Cet algorithme est principalement un algorithme itératif car le processus de rétropropagation est répété jusqu'à ce que le réseau ait atteint un état stable. Il est aussi un algorithme d'apprentissage par essai et erreur, car il n'assure pas une adaptation pertinente pour chaque paramètre à chaque itération. En effet, il considère un gain constant lors de la variation des paramètres selon la direction du gradient. De plus, la direction du gradient est dictée par un seul exemplaire et ne tient pas compte du fait qu'un autre exemplaire pourrait influencer le réseau dans une direction complètement opposée. Cela pourrait même entraîner une augmentation de la fonction objective et du nombre d'itérations.

2.2.4.1.2. La méthode de descente du gradient

L'algorithme standard d'optimisation à l'aide de la technique de la descente du gradient (DG) est une procédure d'optimisation non-linéaire du premier ordre [11, 12, 13]. Contrairement à la méthode de rétropropagation, l'algorithme GD est utilisé dans le contexte de l'entraînement des réseaux neuroniques, pour minimiser l'erreur globale SE donnée par l'équation (2-9).

A chaque itération, les paramètres du réseau sont ajustés comme suit:

$$v(k+1) = v(k) - \eta\, g_{SE,v}(k) \qquad (2\text{-}15)$$

où,

$$v = \begin{bmatrix} v_1 \\ \vdots \\ v_{n_p} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \vdots \\ w_{i,j,c} \\ \vdots \end{bmatrix} \\ \begin{bmatrix} \vdots \\ \theta_{j,c} \\ \vdots \end{bmatrix} \end{bmatrix} \quad \text{et} \quad g_{SE,v} = \begin{bmatrix} \partial SE / \partial v_1 \\ \vdots \\ \partial SE / \partial v_{n_p} \end{bmatrix} \qquad (2\text{-}16)$$

n_p : est le nombre de paramètres dans le réseau. Il correspond au nombre total de connexions et de neurones dans le réseau (nombre total de poids et d'écarts).

Les dérivées partielles de SE par rapport aux paramètres du réseau sont calculées à l'aide de l'équation suivante:

$$\partial SE/\partial v_i = \sum_{e=1}^{n_e} \partial SE_e/\partial v_i \qquad (2\text{-}17)$$

Contrairement à la méthode BP, la méthode GD utilise un coefficient de gain η variable. A chaque itération, η est déterminé à l'aide d'un algorithme de recherche directionnelle. Cet algorithme détermine la valeur de η pour la quelle l'erreur SE est minimisé dans la direction négative du gradient $g_{SE,v}$ (vecteur gradient de SE par rapport aux paramètres v du réseau (poids et écarts)). Puisque la recherche se fait dans une direction qui a été déterminée en considérant tous les exemplaires d'entraînement, il y a donc amélioration de la fonction objective SE d'une itération à l'autre. Les itérations sont répétées tant et aussi longtemps que la norme du vecteur gradient $g_{SE,v}$ est non nulle ou que la variation de SE demeure significative.

2.2.4.1.2.1. L'algorithme de recherche du gain optimal

L'algorithme de recherche du gain optimal est fondamentalement une méthode de minimisation monovariable qui minimise SE', une transformation de la fonction multivariable SE, en une fonction monovariable qui dépend de ce même gain [11, 12, 14]. Pour une valeur donnée du gain d'adaptation, SE' donne la valeur de SE dans la direction négative du gradient passant par un point P dans l'espace des paramètres du réseau. Les coordonnées du point P correspondent aux valeurs des paramètres du réseau lors de l'itération d'apprentissage et sont données par le vecteur $\vec{v}$. La fonction SE' est donnée par:

$$\begin{aligned} &SE': R1 \rightarrow R1 \\ &\quad \eta \rightarrow SE'(\eta) = SE(\vec{v} - \eta \vec{g}_{SE,v}) \end{aligned} \qquad (2\text{-}18)$$

2.2.4.1.3. La méthode quasi-Newton

La fonction objective de l'équation (2-9) peut aussi être minimisée à l'aide de la méthode quasi-Newton. Celle-ci est une procédure de minimisation non-linéaire du deuxième ordre [11, 12]. En effet, la direction de recherche du gain optimal est obtenue en pré-multipliant le gradient par une matrice $H_{SE,v}$ qui est une approximation de l'inverse de la matrice hessienne (G) soit, $-H_{SE,v}\vec{g}_{SE,v}$,où;

$$H_{SE,v} \approx G_{SE,v}^{-1} = \begin{pmatrix} \partial SE^2/\partial v_1^2 & \cdots & \partial SE^2/\partial v_1 \partial v_{n_p} \\ \vdots & \ddots & \vdots \\ \partial SE^2/\partial v_{n_p}\partial v_1 & \cdots & \partial SE^2/\partial v_{n_p}^2 \end{pmatrix}^{-1} \qquad (2\text{-}19)$$

Pour garantir que la direction $-H_{SE,v}\vec{g}_{SE,v}$ soit toujours une direction de descente sans avoir besoin d'évaluer exactement les dérivées secondes, la matrice $H_{SE,v}$ doit être définie positive.

Cependant, $H_{SE,v}$ est invariablement une matrice très dense et qui pourrait nécessiter une grande mémoire si le nombre de paramètres dans le réseau est très grand. Puisque la capacité de mémoire des ordinateurs modernes devient de plus en plus importante, cela ne constitue pas alors un problème surtout pour un nombre modéré de variables dans le réseau.

A chaque itération, les paramètres du réseau sont ajustés comme suit:

$$\vec{v}(k+1) = \vec{v}(k) - \eta H_{SE,v}(k)\vec{g}_{SE,v}(k) \qquad (2\text{-}20)$$

Le gain d'apprentissage selon chaque direction est déterminé en utilisant le même algorithme de recherche du gain optimal utilisé par la méthode GD.

2.2.1.4.4. Comparaison et discussion

Une comparaison entre les trois méthodes d'apprentissage présentées ci-dessus a été fait par Riadh Azouzi [15]. La vitesse et la qualité de la convergence des trois méthodes sont comparées en se basant sur l'entraînement de réseaux neuroniques pour modéliser la déviation dimensionnelle et le fini de surface en fonction de: la vitesse d'avance, la vitesse de coupe, la profondeur de coupe, la force de coupe, la vibration, les émissions acoustiques et les déflexions de l'outil.

La comparaison des trois méthodes a permis de faire les remarques suivantes:

- Avec la méthode de rétropropagation standard et la méthode de descente du gradient, l'apprentissage s'effectue très lentement.
- Plus le réseau est grand, plus la courbe de SE obtenue avec la méthode de rétropropagation standard, présente des oscillations. Ceci peut s'expliquer par le fait que l'adaptation des paramètres ne s'effectue pas selon une direction unique à chaque itération.
- De son côté, la méthode de descente du gradient ne présente pas d'oscillation mais atteint son niveau minimum très rapidement et l'améliore très pauvrement lors des itérations subséquentes.
- Manifestement, la méthode d'entraînement de quasi-Newton fournit les meilleurs résultats; non seulement la courbe de l'erreur SE est monotone et décroissante, mais aussi une valeur très petite de SE est atteinte après quelques dizaines d'itérations seulement.

Enfin, il est important de noter que même si les méthodes du premier ordre (rétropropagation et descente du gradient) nécessitent un très grand nombre d'itérations, elles sont économiques en espace mémoire. La convergence avec la méthode de quasi-Newton est extrêmement rapide et amène à une précision infinitésimale, cependant elle nécessite beaucoup d'espace mémoire (plus de variables à stocker et plus de temps de calcul pour chaque itération).

Il y a donc un compromis à faire entre les facteurs de temps de calcul, d'espace mémoire requis et de précision recherchée. Les deux premiers facteurs ont peu d'influence quand il s'agit de réseaux de petites tailles ou même de tailles modérées. En plus avec la nouvelle technologie, les problèmes d'espace mémoire et vitesse d'exécution se manifestent de moins en moins. Toutefois, les récents progrès dans les techniques numériques pour la minimisation des fonctions non-linéaires incluant un grand nombre de paramètres rendent la méthode de quasi-Newton peu prometteuse pour l'entraînement des réseaux neuroniques. Dans ce sens, notre intérêt dans le présent mémoire portera essentiellement sur la méthode de rétropropagation pour élaborer un algorithme d'apprentissage adaptatif.

2.2.5. Le réseau de Kohonen

Le réseau de Kohonen a été introduit par Teuvo Kohonen en 1981 [16] comme étant un excellent modèle de l'organisation topographique dans le cerveau biologique. Ce réseau est essentiellement une fonction de réduction de dimension: Il permet de cartographier un espace de dimension k sur une carte de dimension 2 $\left(R^k \rightarrow R^2\right)$. Cette carte est discrétisée en un nombre fini de positions. Chaque position est occupée par un neurone qui est relié à l'espace d'entrées (les entrées du réseau) par des vecteurs de dimension k. Les composantes de ces vecteurs sont les paramètres du réseau (les poids) dont les valeurs sont obtenues itérativement à l'aide d'un algorithme d'apprentissage non supervisé.

Après qu'un certain nombre d'itérations ait été faites et que l'algorithme d'entraînement ait convergé, la carte topographique ainsi formée représente une approximation de l'espace d'entrées [10]. Cette approximation se matérialise par les vecteurs de poids qui sont emmagasinés dans le réseau. Nous désignons ces vecteurs par le terme "prototypes". Ces derniers constituent une représentation fidèle, mais compacte, des caractéristiques importantes des exemplaires d'entrées utilisés pour la formation de la carte. De plus, cette carte est topographiquement ordonnancée, c'est-à-dire que l'emplacement de chaque neurone dans la carte correspond spécifiquement à des traits saillants qui caractérisent les exemplaires d'entraînement.

Une autre propriété importante de la carte topographique formée dans le réseau de Kohonen consiste en sa capacité à refléter les variations statistiques de la distribution des entrées [17]. En effet, les régions de l'espace d'entrées, à partir desquelles la majorité des exemplaires d'entraînement sont échantillonnés, seront représentées par d'importants domaines sur la carte topographique générée par le réseau. En conséquence, la résolution de la représentation de telles régions sera plus grande que celles des régions où seulement un petit nombre d'exemplaires est choisi.

La Figure 2.5 montre une représentation typique du réseau de Kohonen. Ce réseau possède k entrées et une carte topographique de dimension Nrs x Ncs (Nrs rangées et Ncs colonnes). Chaque neurone se trouvant à la position (i, j) de la carte topographique $(1 \le i \le Nrs, 1 \le j \le Ncs)$, est relié aux différentes entrées du réseau à l'aide d'un vecteur de paramètres $\vec{u}_{i,j}$:

$$\vec{u}_{i,j} = \left[u_{i,j,1}, u_{i,j,2}, \cdots u_{i,j,p}, \cdots u_{i,j,k}\right]^T \qquad (2\text{-}21)$$

À noter que chaque composante $u_{i,j,k}$ du vecteur $\vec{u}_{i,j}$ est associée à la connexion reliant le neurone (i, j) à l'entrée k. À sa sortie, un neurone (i, j) fournit une mesure de la distance euclidienne $d_{i,j}$ entre le vecteur des paramètres qui lui est associé et le vecteur d'entrée qui lui arrive $\vec{x} = \left[x_0, x_1, \cdots, x_i, \cdots, x_k\right]^T$. La distance $d_{i,j}$ est donnée par:

$$d_{i,j} = \|\vec{x} - \vec{u}\| = \sqrt{\sum_{p=0}^{k}\left(x_p - u_{i,j,p}\right)^2} \qquad (2\text{-}22)$$

Les valeurs des paramètres $u_{i,j,k}$ seront fixées une fois la formation de la carte topographique terminée. Pour cela, il faut entraîner le réseau. D'abord, les paramètres $u_{i,j,k}$ sont initialisés à de petites valeurs choisies aléatoirement. Puis, les étapes suivantes sont répétées à chaque itération jusqu'à la convergence:

1) Sélectionner aléatoirement un exemplaire d'entrées parmi l'ensemble des exemplaires d'entrées disponibles pour l'entraînement du réseau et le présenter à l'entrée du réseau.
2) En utilisant l'équation (2-22), déterminer la distance euclidienne qui sépare cet exemplaire d'entrées de chaque position de la carte topographique du réseau (ou de chaque prototype emmagasiné dans le réseau).
3) Trouver la position (i_{min}, j_{min}) du neurone par rapport à laquelle la distance $d_{i,j}$ est minimale. C'est-à-dire qu'il faut chercher $d_{i_{min},j_{min}}$ tel que:

$$d_{i_{min},j_{min}} = \min_{i,j}\left\{d_{i,j} : \left(1 \le i \le Nrs, 1 \le j \le Ncs\right)\right\} \qquad (2\text{-}23)$$

4) Adapter les valeurs des paramètres du neurone "gagnant" (neurone se trouvant à la position (i_{min}, j_{min})) et des neurones avoisinants. Généralement, la règle suivante est utilisée:

$$u_{i,j,k}(t+1) = \begin{cases} u_{i,j,k}(t) + \eta(t)\left[x_k(t) - u_{i,j,k}(t)\right] & si\ (i,j) \in \Lambda_{(i_{min},j_{min})}(t) \\ u_{i,j,k}(t) & si\ (i,j) \notin \Lambda_{(i_{min},j_{min})}(t) \end{cases} \qquad (2\text{-}24)$$

où t indique l'itération en cours, $\eta(t)$ est le gain d'adaptation et $\Lambda_{(i_{min},j_{min})}(t)$ est une fonction de voisinage centrée autour du neurone gagnant.

5) Répéter les étapes 1, 2, 3 et 4 jusqu'à la convergence, c'est-à-dire jusqu'à ce que les variations des paramètres d'une itération à l'autre deviennent négligeables.

À l'équation (2-24), le gain d'adaptation et le rayon de voisinage sont variés dynamiquement durant l'apprentissage. Tel que montré à la Figure 2.5, leurs valeurs dépendent de t qui représente le nombre d'itérations d'apprentissage accompli.

Le rayon $\Lambda_{(i_{min},j_{min})}(t)$ est généralement choisi de manière à inclure tous les neurones se trouvant dans une région de forme carrée autour du neurone gagnant. Par exemple, un rayon de 1 fait en sorte que la région couverte par $\Lambda_{(i_{min},j_{min})}(t)$ n'inclut que le neurone gagnant lui-même et ses huit voisins les plus proches (voir Figure 2.5). Au début de la formation de la carte, le rayon

doit être assez grand de façon à couvrir tous les neurones du réseau, puis il est réduit graduellement à toutes les t_q itérations (généralement $k_p = 0$).

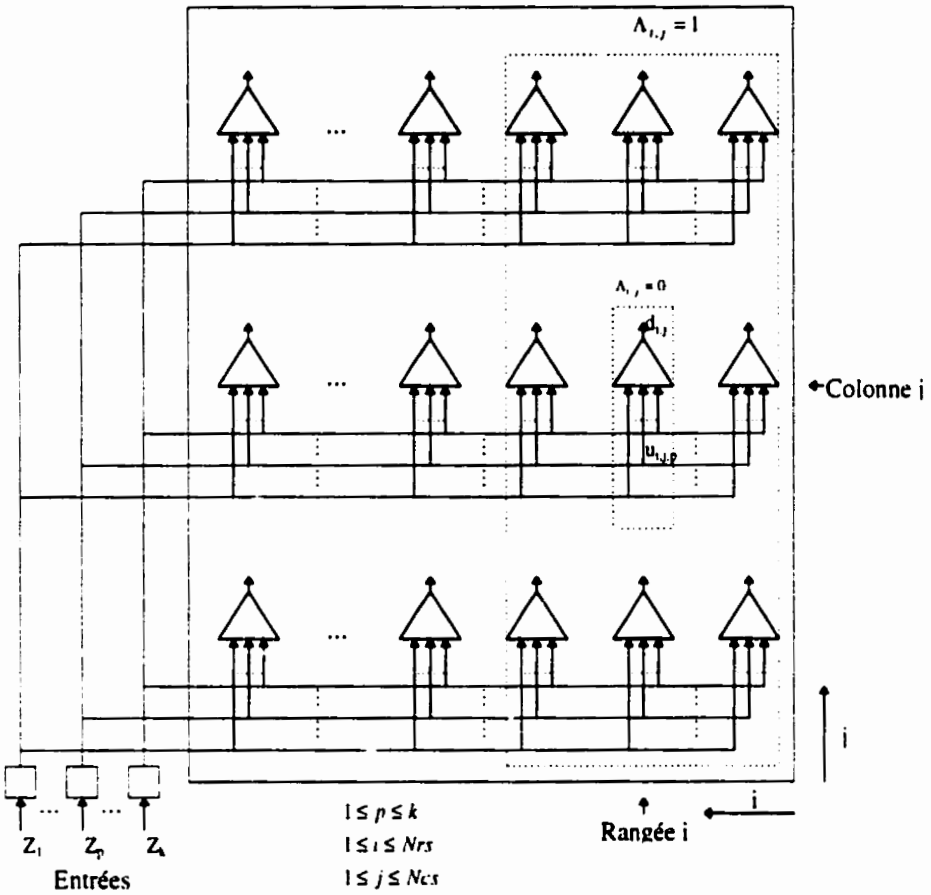


Figure 2.5. Le réseau de Kohonen

2.3. AVANTAGES ET LIMITES DES RÉSEAUX NEURONIQUES

Les réseaux neuroniques peuvent être utilisés dans n'importe quel problème qui dispose d'un ensemble de données d'entraînement. Il est aussi prouvé qu'avec les réseaux neuroniques on peut modéliser toute fonction continue incluant celles modélisées par des méthodes de statistique, et le plus souvent avec plus de performance et de précision [18, 19]. En plus, l'avantage majeur de n'importe quel réseau neuronique est son approche non-paramétrique. Ils ne demandent aucune connaissance antérieure de la distribution des modèles.

Récemment, il y a eu plusieurs tentatives pour modéliser une grande variété de problèmes dans différents domaines en utilisant l'approche neuronique. A titre d'exemple, citons les applications suivantes: la reconnaissance de formes, le traitement adaptatif des signaux, la modélisation dynamique, la surveillance des procédés. Les réseaux neuroniques sont également utilisés dans les systèmes experts et les systèmes à logique floue, et dans d'autres applications spécifiques comme le contrôle de température [20, 21, 22]. Dans la majorité de ces applications, les réseaux neuroniques ont été jugés comme étant un outil prometteur de modélisation.

Toutefois, la nature de type "boîte noir" de cette technologie et l'absence d'interprétation des poids des connections, peuvent justifier sa non-utilisation.

2.3.1. Avantages des réseaux neuroniques

Les avantages des réseaux neuroniques sont facilement justifiés sur la base de considérations pratiques:

- la capacité de modéliser des problèmes complexes,
- une plus grande liberté et flexibilité que les approches conventionnelles,
- la capacité d'extraire l'information nécessaire à partir d'un nombre fini d'exemplaires,
- la capacité d'apprendre avec des données qui contiennent du bruit,
- la possibilité de rassembler à la fois la simplicité relative des approches agrégées et l'avantage théorique des approches désagrégées sans la complexité de ces dernières,
- la capacité de modéliser des problèmes qui présentent une grande variété de données multidimensionnelles,
- les réseau neuroniques vont au delà de la simple régression en permettant d'obtenir automatiquement la forme de la fonction recherchée tout en ajustant cette forme aux données,

- la facilité de l'implantation de la méthode d'apprentissage en continu pour les réseaux neuroniques justifie à elle seule l'utilisation de ces réseaux pour la commande des procédés (adaptation des modèles), et
- la structure parallèle des réseaux permet de réduire énormément le temps de calcul. Ils sont donc très pratiques pour l'utilisation en temps réel.

2.3.2. Limites des réseaux neuroniques

Les limites du réseau neuronique sont:

- l'absence d'interprétation de valeurs des poids (les connections), et de l'impact relatif de chaque variable d'entrée sur les variables de sorties, et
- la capacité de prédiction n'est pas toujours améliorée avec plus d'apprentissage; les résultats pour les exemplaires avec lesquels il a été entraîné sont précis, mais la prédiction avec les données qui n'appartiennent pas à l'ensemble d'entraînement est faible si on a un "sur-apprentissage" [23]. Le réseau peut avoir une erreur moyenne très petite avec les données d'apprentissage, mais il peut ne pas justifier cette performance avec les données d'un test indépendant. Dans certain cas, les réseaux neuroniques arrivent à apprendre les réponses pour des données contenant du bruit, mais ils ne peuvent pas les généraliser pour un ensemble de données plus large.

2.4. CONCLUSION

Dans ce chapitre, nous avons effectué une brève revue des concepts de base des réseaux neuroniques, en particulier le réseau perceptron multicouche et le réseau de Kohonen, ainsi que leurs algorithmes d'apprentissage. Au chapitre suivant, nous allons étudier quelques algorithmes de neurocommande et les techniques utilisées pour l'adaptation en continu.

CHAPITRE III

COMMANDE DE PROCÉDÉ

3.1. INTRODUCTION

Durant les dernières années, la neurocommande a évoluée vers des techniques puissantes, généralisées et extrêmement prometteuses pour la commande des procédés très complexes. Dans les sections qui suivent, nous revoyons les algorithmes de neurocommande et les techniques utilisées pour la régulation et l'adaptation en continu.

3.2. REVUE DE LA NEUROCOMMANDE

3.2.1. Approches de neurocommande

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

3.2.1.1. Neurocommande directe

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

La Figure 3.1a montre un schéma de commande supervisée. Dans ce schéma, un réseau neuronique comme celui montré à la Figure 3.1b représente les transformations qui existent entre les signaux des senseurs et les actions de commande. Les exemplaires utilisés pour l'entraînement de ce réseau peuvent être obtenus simplement en remplaçant le contrôleur neuronique par un opérateur humain. Dans ce cas, l'opérateur doit déterminer minutieusement les actions de commande non pas en se basant sur ses propres sens et son expérience mais plutôt en se basant uniquement sur les informations fournies par les senseurs et les estimations de performance dérivées à partir de ces senseurs. En entraînant a priori le réseau à l'aide de ces exemplaires, la dynamique du procédé et le temps de réponse de l'opérateur humain seront communiqués au neurocontrôleur. La sortie du procédé y_c (la consigne) est implicitement définie dans la pensée de l'opérateur comme étant un objectif. Néanmoins, cet objectif est présent dans les exemplaires et dans les connaissances acquises par le réseau neuronique.

Une des propriétés les plus importantes des réseaux neuroniques, est leur aptitude à apprendre presque n'importe quelle relation à partir d'un ensemble d'exemplaires d'Entrées/Sorties (E/S). Cette propriété est vraie aussi pour l'apprentissage de l'inverse de ces relations. En effet, si l'on considère un ensemble d'exemplaires représentant le comportement physique et dynamique d'un procédé, alors le réseau neuronique peut apprendre l'inverse de cette relation simplement en commutant les entrées et les sorties dans les exemplaires avant de commencer l'opération d'apprentissage. Un modèle inverse de la dynamique du procédé sera ainsi obtenu.

En utilisant cette dernière propriété, le schéma de commande de la Figure 3.1 pourrait devenir prédictif si l'on représente explicitement la consigne comme entrée supplémentaire au réseau neuronique (voir la Figure 3.1 c). En effet, en étudiant le procédé, il serait possible d'obtenir la sortie y du procédé au temps t+1 en fonction des commandes u et des sorties y aux temps t-i où i=0 à n. La valeur de n dépend spécialement de l'ordre de la dynamique du procédé. Après avoir appris l'inverse de cette dernière relation, le neurocontrôleur peut être intégré dans différents schémas de commande directe comme celui de la Figure 3.1a. Contrairement à un

système de neurocommande supervisé, le neurocontrôleur inverse apprend explicitement la commande à fournir pour obtenir une certaine sortie désirée du procédé.

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

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

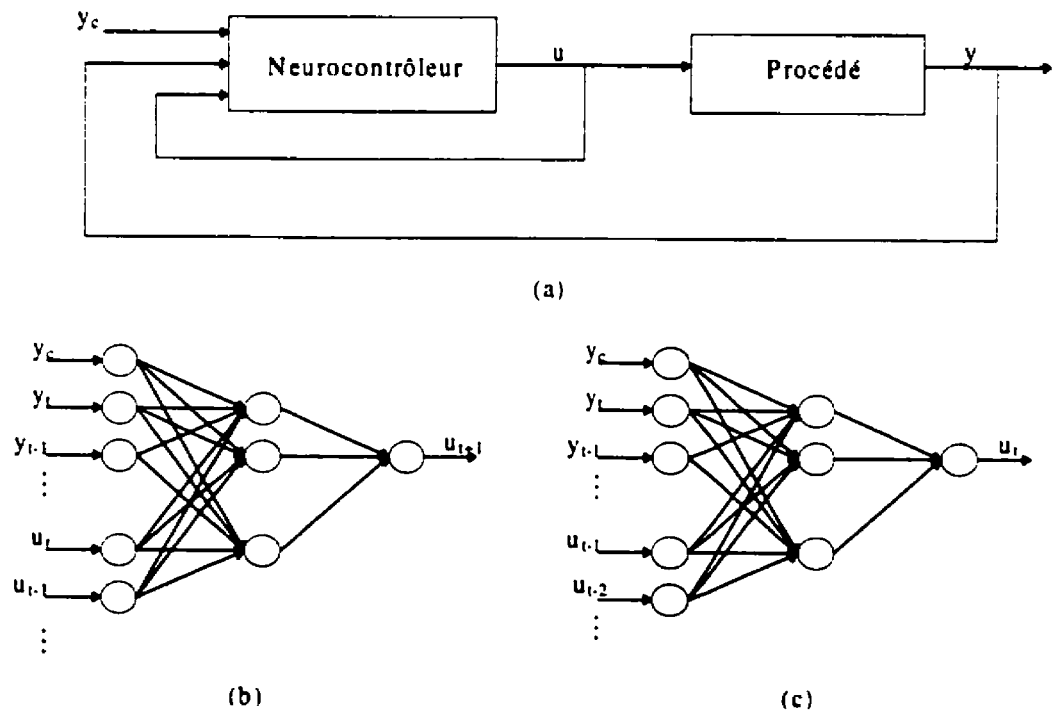


Figure 3.1. (a) Schéma de neurocommande directe, (b) neurocommande supervisée et (c) neurocommande par modèle inverse

3.2.1.2. Neurocommande indirecte

Dans une approche de neurocommande indirecte, le réseau neuronique est utilisé typiquement pour paramétrer ou modéliser le procédé. L'objectif de la commande est exprimé explicitement tel que montré à la Figure 3.2. Dans ce schéma, le modèle neuronique fournit une estimation de la prochaine sortie du procédé y. Le régulateur compare la sortie estimée à la consigne et propose une meilleure action de commande en se basant sur une certaine stratégie. Il existe plusieurs types de régulateurs. À titre d'exemple, nous citons le travail de A. Drouin [24] qui a utilisé les régulateurs PID, L/A forme de base, L/A forme PI et L/A forme PID pour effectuer la commande du système liquide-niveau. Le modèle neuronique du procédé est entraîné a priori en utilisant un certain nombre d'exemplaires d'E/S prédéterminés, alors que l'adaptation en continu se fait en rétropropageant l'erreur qui correspond à la différence entre la sortie du procédé et la sortie prédite par le modèle neuronique (voir Figure 3.2).

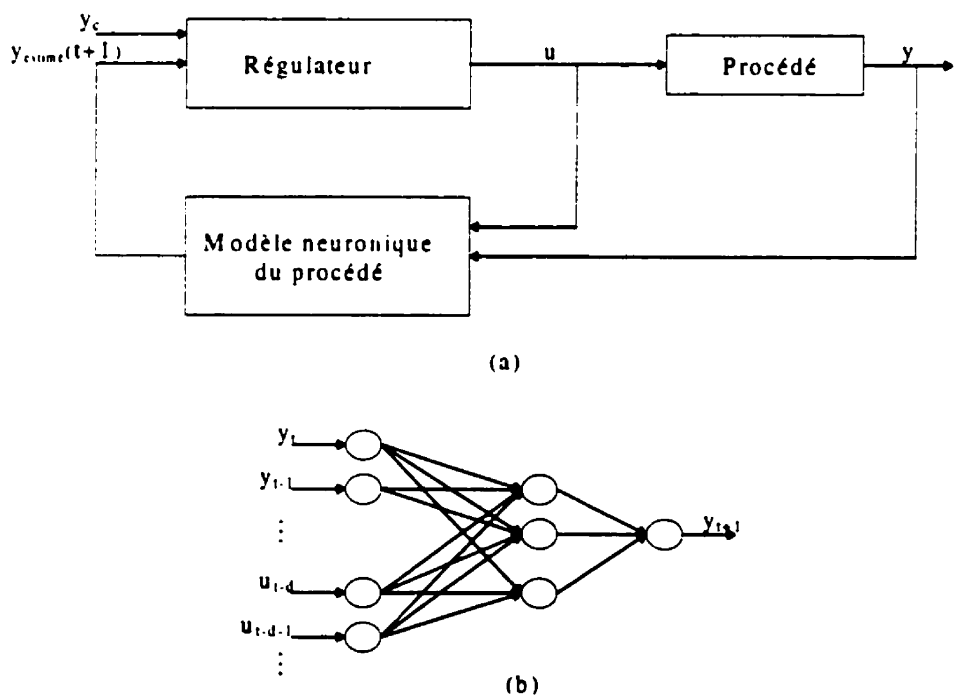


Figure 3.2. (a) schéma explicite pour la régulation du procédé et (b) modèle neuronique du procédé

3.2.2. Les techniques d'adaptation

Dans leurs revues des méthodes d'adaptation des modèles neuroniques, Thibault et Grandjean [25] distinguent deux méthodes principales auxquelles les chercheurs ont eu recours: (i) la méthode d'adaptation en lot, et (ii) la méthode d'adaptation récursive.

La première méthode évoque les techniques d'identification en lot où, à chaque itération de commande, un nombre représentatif des derniers exemplaires d'E/S du procédé est utilisé pour entraîner le modèle neuronique, le forçant à suivre la plus récente dynamique du procédé (A. Drouin [24]).

La seconde méthode, la méthode récursive, adapte les coefficients synaptiques du réseau en réponse au dernier exemplaire seulement. Une seule itération d'adaptation est effectuée à chaque itération de commande. L'erreur à la sortie du modèle est rétropropagée sous une forme ou une autre pour corriger les poids du réseau neuronique.

Dans la majorité des cas, l'algorithme standard d'entraînement par la méthode de rétropropagation est associé à la méthode d'adaptation récursive, alors que la méthode d'adaptation en lot permet à d'autre algorithmes d'apprentissage d'être utilisés. À titre d'exemple, citons l'algorithme de quasi-Newton et l'algorithme de descente du gradient.

Les deux stratégies d'adaptation présentées ci-dessus, ne sont pas efficaces pour l'identification en continu du procédé sur tout le domaine de variation des paramètres de commande, car elles présentent deux problèmes importants:

(i) Le réseau s'adapte très lentement aux nouveaux exemplaires. Ce problème est plus évident avec la méthode d'adaptation récursive où une seule itération d'entraînement est effectuée sur le nouvel exemplaire. Cette méthode est souvent associée à l'algorithme standard de rétropropagation qui converge très lentement. Avec la méthode d'adaptation en lot, la vitesse

de convergence est très dépendante du nombre d'exemplaires utilisés pour adapter les paramètres du réseau et du nombre d'itérations d'adaptation effectuées à chaque fois.

(ii) L'entraînement du réseau avec les nouvelles données cause d'importantes dégradations aux anciennes connaissances qui y sont déjà emmagasinées. Après un certain nombre de cycles échantillonnage commande/adaptation, le réseau va présenter correctement le procédé uniquement à proximité du plus récent point d'opération, oubliant partiellement le comportement du procédé au niveau des autres régions de l'espace d'entrées. Ce deuxième problème est beaucoup plus critique que le premier. Non seulement cela réduit les capacités du système neurocommande à fournir un point d'opération optimal, mais aussi il affecte sa stabilité.

En particulier, le problème de dégradation de l'information mémorisée est fortement lié à la nature du traitement de l'information dans les réseaux neuroniques artificiels. Effectivement, ces réseaux traitent les informations d'une manière totalement distribuée, et négligent toute forme d'organisation spatiale ou de classification des informations. La modification de n'importe quel paramètre dans le réseau par la méthode d'adaptation en lot ou bien par la méthode récursive, entraîne automatiquement la modification du réseau en entier et affecte ainsi la mémorisation d'un grand nombre d'exemplaires préalablement stockés dans le réseau.

Pour mettre en évidence les principales causes de ce comportement, il faut étudier les méthodes employées pour l'adaptation des paramètres du réseau neuronique. La méthode d'adaptation en lot et la méthode récursive se réduisent toutes les deux à une implantation partielle de l'un ou l'autre des algorithmes utilisés pour l'entraînement a priori des réseaux neuroniques. Dans ces derniers algorithmes, les paramètres du réseau sont adaptés à l'aide de quelques relations dont la forme généralisée est donnée par l'équation suivante:

$$\vec{v}(k+1) = \vec{v}(k) - \eta(t) f_1\left(\vec{g}_{SE,v}(k)\right) + \alpha(\vec{v}(k) - \vec{v}(k-1)) \qquad (3\text{-}1)$$

où,

$\vec{v}(k)$:est le vecteur contenant les paramètres du réseau (poids et écarts),

$\vec{g}_{SE,v}(k)$: est le vecteur gradient de SE par rapport aux composantes du vecteur $\vec{v}(k)$,

f_1 : est une fonction du gradient $\vec{g}_{SE,v}(k)$,

k : est l'itération d'adaptation en cours,

α : est le momentum , et

η : est le gain.

L'algorithme standard de rétropropagation utilise un vecteur gradient calculé en fonction d'un seul exemplaire à la fois. Dans ce dernier algorithme, le gain d'adaptation η demeure constant à chaque itération d'adaptation.

En réalité, le vecteur gradient dans l'équation (3-1) constitue la direction selon laquelle les paramètres du réseau varient. Ainsi, si cette direction est obtenue seulement en fonction des plus récents exemplaires, alors les paramètres du réseau varient en fonction de ces exemplaires seulement, et tout autre exemplaire est ignoré (cela inclut les exemplaires appris auparavant).

Cette situation devient encore plus critique si, après un certain nombre d'itérations de commande, les variables d'entrées ne changent pas. Dans ces conditions, les plus récents exemplaires seront tous très similaires et le réseau se trouve à être adapté en fonction du même exemplaire (ou un exemplaire qui varie très peu pendant plusieurs itérations de commande successives).

En essayant d'atténuer la gravité de ces problèmes, certains chercheurs ont établi des précautions à prendre lors de l'implantation de l'une ou l'autre des deux méthodes d'adaptation présentées. Les plus importantes sont les suivantes:

1- Avec la technique d'adaptation en lot, l'oubli des événements antérieurs est légèrement atténué par la variation du nombre d'itération d'entraînement et du nombre des derniers exemplaires que l'on présente au réseau.

2- Lorsqu'une technique récursive est employée, les gains d'adaptation sont limités à de faibles valeurs et ce, dans le but de réduire les variations des paramètres d'apprentissage.

3- Adapter les paramètres du réseau en utilisant l'erreur actuelle et employer le temps de calcul restant pour l'apprentissage des exemplaires antérieurs.

4- Durant l'adaptation continue, fixer des limites minimales et maximales pour les valeurs des paramètres du réseau afin de prévenir les changements importants.

5- N'adapter les paramètres du réseau que lorsqu'il y a suffisamment d'informations dans les exemplaires d'E/S, c'est-à-dire que les plus récents exemplaires doivent être différents de façon assez significative.

6- Varier périodiquement les entrées du procédé pour y injecter une excitation et éviter que les plus récents exemplaires deviennent peu différents.

En réalité, ces précautions n'améliorent pas la vitesse de convergence de l'algorithme d'adaptation et ne réduisent pas suffisamment la dégradation des informations stockées a priori dans le réseau pour éviter l'instabilité du neurocontrôleur. L'instabilité du neurocontrôleur se manifeste par son incapacité de maintenir le point d'opération proche du point d'opération optimal, et ce durant un certain nombre d'itérations de commande successives.

En plus des problèmes d'oubli et de lenteur de l'adaptation, il y a les problèmes qui sont spécifiques pour certains algorithmes de neurocommande. Ainsi, avec les techniques d'adaptation spécialisées qui sont utilisées dans les systèmes de neurocommande par modèle inverse, le signal d'erreur qui sera employé pour la correction des paramètres du réseau ne peut pas être obtenu directement en comparant la sortie du modèle neuronique avec la sortie du procédé. Pour remédier à ce problème, les chercheurs proposent d'obtenir ce signal à partir d'un modèle direct du procédé. Par exemple, le schéma de commande de la Figure 3.1 peut être transformé en un schéma adaptatif tel que illustré dans la Figure 3.3. La variation de la sortie y_c - y(t) est entrée au

modèle direct. Celui-ci détermine l'erreur u_c - u(t-1) qui sera finalement rétropropagée à travers le réseau. Pour obtenir le modèle direct, certains chercheurs ont utilisé une fonction de transfert de forme connue et dont les paramètres sont déterminés expérimentalement, d'autres se sont servis d'un réseau neuronique entraîné a priori. Dans un cas comme dans l'autre, le modèle direct demeure dans son état original. Il en résulte que le neurocontrôleur ne pourra apprendre que ce que le modèle direct pourrait lui montrer. Cela constitue un handicap majeur des techniques spécialisées d'adaptation car les connaissances emmagasinées dans le modèle direct peuvent être très limitées et peuvent contenir des erreurs.

Cette brève revue démontre qu'il y a un manque important de stratégies efficaces pour l'adaptation en continu des réseaux neuroniques. Les stratégies existantes engendrent, d'une façon ou d'une autre, des problèmes de stabilité dans la commande. Comme nous le verrons dans le prochain chapitre, l'amélioration des capacités d'adaptation du perceptron multicouche par la modification de sa structure et de son algorithme d'apprentissage est une solution très prometteuse et constitue en fait la contribution majeur de ce mémoire.

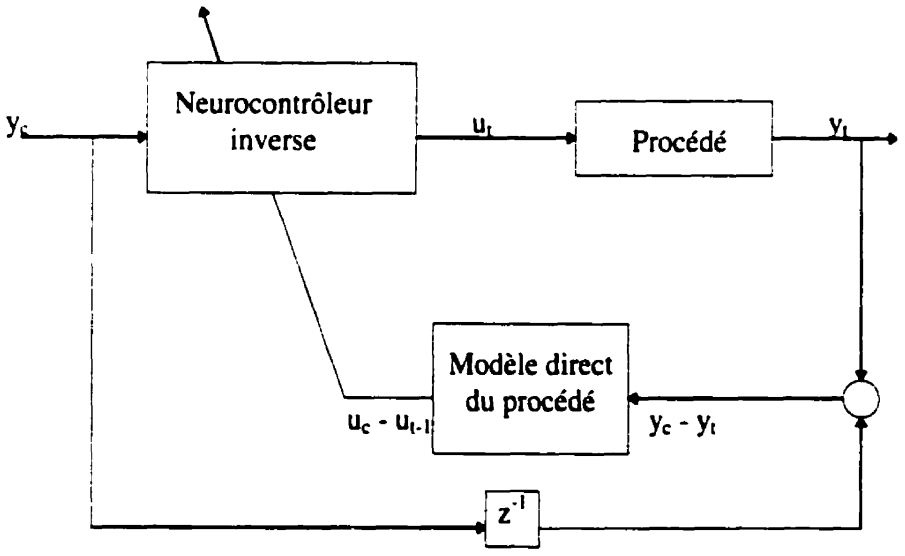


Figure 3.3. Un système typique de neurocommande directe adaptative

3.3. RÉGULATEUR NEURONAL

La commande neuronique pour la régulation est une appellation qui englobe une vaste gamme de systèmes servant à maintenir des paramètres tels que température, pression et niveau à une valeur prescrite dans toutes les conditions de service. Pour posséder un bon système de commande neuronique, les éléments qui composent le système sont d'une très grande importance. Les éléments composant une boucle de rétroaction sont: le capteur, l'actionneur, le régulateur et l'organe de commande.

La philosophie derrière l'utilisation des réseaux neuronaux pour la modélisation ou la commande d'un procédé est très différente de celle des techniques de modélisation conventionnelles. Quoique le réseau neuronal puisse servir de modèle, c'est surtout son utilisation comme régulateur qui est très intéressante. Même si le régulateur neuronal possède plus de paramètres d'ajustement qu'un régulateur classique, il n'y a pas d'ajustements tels quels à effectuer. Un réseau neuronal devient régulateur après un apprentissage de données expérimentales provenant du système à réguler. Une fois l'apprentissage terminé, c'est-à-dire l'ajustement des poids et des écarts, le régulateur neuronal est en mesure de déterminer quelle commande le procédé doit recevoir pour atteindre la consigne désirée. Une fois que le réseau de neurones artificiels a appris la dynamique directe ou inverse d'un procédé, il peut par la suite s'adapter à l'évolution de ce dernier par l'apprentissage dynamique ou en ligne. Une application des réseaux neuronaux pour commander un système liquide-niveau (un système de contrôle du niveau d'eau dans un réservoir) se trouve dans [24].

3.3.1. Structure de commande avec le modèle neuronal inverse

Le modèle neuronal inverse, aussi nommé régulateur neuronal, car il peut apprendre à déterminer une commande dans le but d'obtenir la sortie désirée, peut être utilisé dans différentes structures de commande. Une architecture de commande simple utilisant le modèle neuronal inverse est présenté à la Figure 3.4. Elle consiste en un modèle neuronal inverse du procédé utilisé comme régulateur d'anticipation dans une boucle de rétroaction fermée.

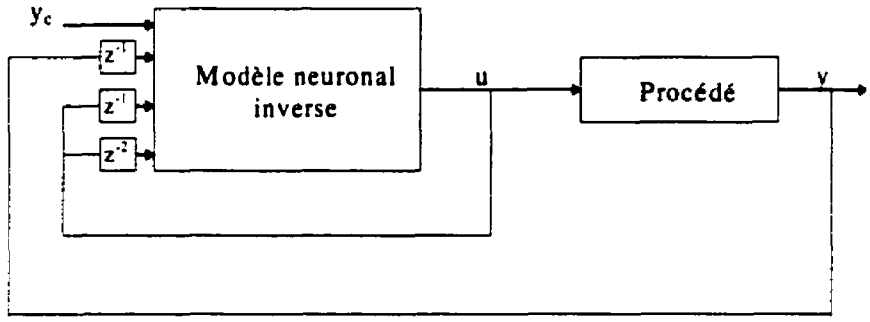


Figure 3.4. Structure de commande ayant comme régulateur le modèle neuronique inverse

Pour l'apprentissage en continu du modèle neuronal inverse, la détermination de la commande que le modèle neuronal inverse aurait dû envoyer à la place de celle obtenue, n'est pas une tâche aussi évidente que dans le cas du modèle neuronal direct. La plupart des algorithmes utilisés pour l'apprentissage du modèle neuronal inverse vont, à partir de l'erreur entre le point de consigne et la sortie du procédé (y_c - y), déterminer l'erreur de commande (u_c - u), c'est-à-dire la différence entre la commande idéale (u_c) et la commande obtenue par le modèle inverse (u). L'erreur de commande Δu s'obtient par la relation suivante [24]:

$$\Delta u = \Delta y/k \qquad (3\text{-}2)$$

où,

k : est le gain du modèle neuronal direct.

Comme l'obtention du gain ou du Jacobien ($\partial y/\partial u$) du procédé ou du modèle neuronal direct n'est pas chose simple, Nguyen et Widrow [26] ont proposé un algorithme utilisant deux réseaux neuronaux, l'un étant le modèle neuronal inverse et le second le modèle neuronal direct. Ce dernier est utilisé pour rétropropager l'erreur entre la consigne et la sortie du procédé dans le but d'estimer l'erreur associée au régulateur neuronal.

L'algorithme d'apprentissage en ligne du modèle inverse adapté dans [24] est schématisé à la Figure 3.5. Comme le modèle neuronal direct est la pierre angulaire de cet algorithme d'apprentissage, la première étape est d'effectuer son apprentissage ou son adaptation, à partir de la valeur prédite y_m (t) à la période d'échantillonnage précédente et de la nouvelle lecture du capteur y(t). De cette façon nous nous assurons que le modèle neuronal direct est continuellement adapté aux perturbations et à l'évolution du système, avant de l'utiliser pour l'apprentissage du modèle neuronal inverse. La deuxième étape est d'envoyer la commande idéale ou améliorée u_{am}. Cette commande est obtenue à partir de la consigne, de la valeur de prédiction et de la commande envoyée. Dans [24], A. Drouin utilise un régulateur de type L/A forme de base pour calculer u_{am}. Avec cette commande idéale et la commande déterminée par le modèle neuronal inverse, nous obtenons une erreur de commande Δu, qui est utilisée pour l'apprentissage du régulateur neuronal.

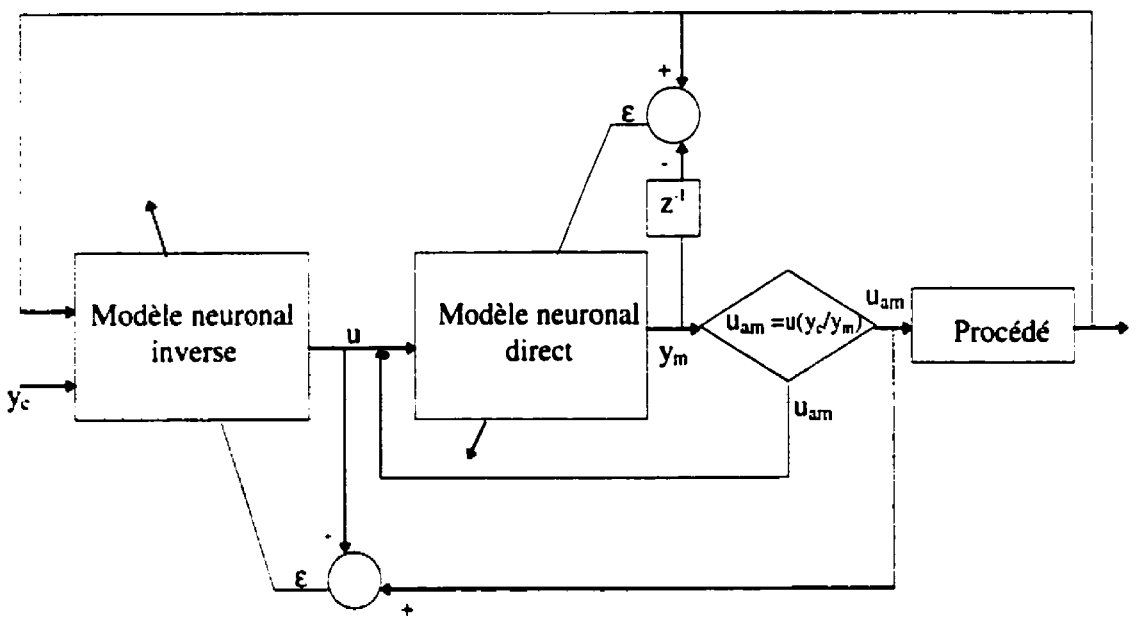


Figure 3.5. Structure de commande prédictive avec adaptation des modèles neuronaux

3.4. CONCLUSION

L'application des réseaux neuronaux, quoique récente, est vouée à un avenir très prometteur, soit pour modéliser et/ou pour commander des systèmes complexes. Toutefois, nous pensons qu'il est nécessaire d'approfondir la recherche pour améliorer les capacités d'adaptation des réseaux neuroniques. L'amélioration des capacités de ces derniers et en particulier du perceptron multicouche doit passer par l'amélioration de la structure de ce dernier et de son algorithme d'apprentissage. Les réseaux neuroniques artificiels sont tous inspirés des réseaux neuroniques biologiques, mais leurs correspondances avec ces derniers demeurent assez limitées.

Dans le prochain chapitre, nous revoyons les connaissances accumulées sur le réseau neuronique biologique. Nous résumons en particuliers les plus importantes et récentes informations. À partir de cette revue, nous proposons des améliorations au réseau perceptron multicouche dans le but d'améliorer ses capacités d'apprentissage et d'adaptation.

CHAPITRE IV

RÉSEAU PERCEPTRON CLUSTERISÉ

4.1. INTRODUCTION

Dans le chapitre précédant, nous avons montré que le problème de dégradation de la mémoire du réseau, lorsque ce dernier est adapté en continu, constitue un sérieux danger pour la stabilité et la performance du système neurocommande.

Comme nous l'avons mentionné au chapitre précédant, l'amélioration des capacités du réseau neuronique artificiel et en particulier, du perceptron multicouche doit passer par l'amélioration de la structure de ce dernier et de son algorithme d'apprentissage. Les réseaux neuroniques artificiels sont tous inspirés des réseaux neuroniques biologiques, mais on est encore loin d'obtenir des résultats comparables à ces derniers.

Sans doute, le très grand réseau neuronique se trouvant dans le cerveau biologique possède d'excellentes capacités d'apprentissage et d'adaptation que l'homme n'est jamais parvenu à réaliser dans les systèmes qu'il met en oeuvre (le cerveau humain comprend 100 milliard de neurones à un facteur de 10 près). Dans la section suivante, nous revoyons les connaissances accumulées sur le réseau neuronique biologique. Nous résumons en particuliers les plus importantes et récentes informations. À partir de cette revue nous proposons des

améliorations au réseau perceptron multicouche dans le but d'améliorer ses capacités d'apprentissage et d'adaptation.

4.2. ÉTUDE DE L'ORGANISATION, DE L'APPRENTISSAGE ET DE LA MÉMORISATION DANS LE CERVEAU BIOLOGIQUE

La compréhension du fonctionnement du cerveau est rendue difficile par l'infinie complexité du réseau de cellules et de fibres nerveuses qui le composent. Cependant, les neurologues (les biologistes spécialistes du système nerveux) ont accumulé sur le cerveau beaucoup d'informations d'ordre anatomique, physiologique, biochimique, etc. Nous avons effectué une revue de ces connaissances pour déterminer à quels niveaux de la structure cellulaire, de l'organisation neuronique et des mécanismes physiologiques et chimiques se manifestent les phénomènes d'apprentissage et de mémoire.

4.2.1. L'organisation du cerveau

L'une des structures qui revêt d'un grand intérêt pour nous est le cortex cérébral. Comportant plus de 75 milliards de neurones, le cortex cérébral est l'enveloppe qui couvre toutes les autre composantes du cerveau. A sa surface, il est pourvu de plusieurs circonvolutions (les gyrus) entre lesquels se trouvent des sillons (les sulcus). Le cortex cérébral est divisé en allocortex (10%) et en néocortex (90%) . L'allocortex comprend la formation hippocampale, le gyrus parahippocampale, le cortex olfactoire ou cortex d'uncus, le gyrus cingulaire, et le gyrus fasciolaire. Le néocortex est composé des lobes frontal, pariétal, occipital, temporal, et central. Le néocortex a été divisé en plusieurs parcelles appelées aires corticales selon leur concentration plus ou moins importante en chacun des types de neurones et leur rôle joué dans les fonctions cognitives.

Vu son rôle essentiel dans le traitement des informations provenant de partout, il est important de connaître et de comprendre les fonctions de chacune des aires corticales, leur structure neuronale, leur relation avec des aires voisines, telles que d'autres structures du cerveau,

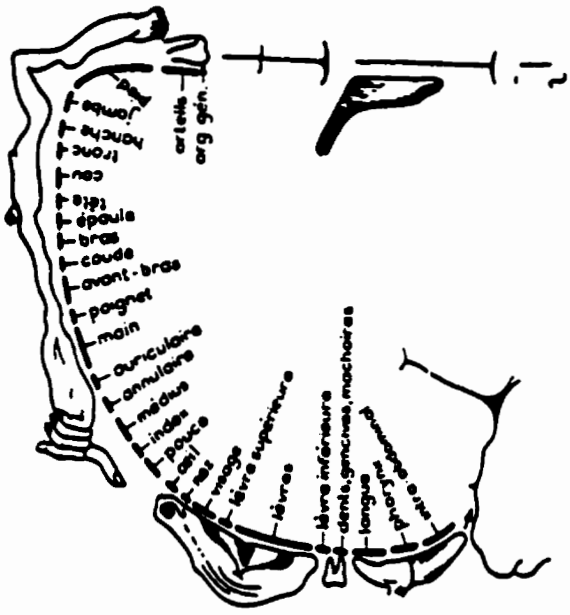
les muscles et les organes sensoriels (Figure 4.1). Pour des fins d'application à l'intelligence artificielle, il est nécessaire d'effectuer des études plus approfondies.

Les neurones du néocortex sont de trois types:

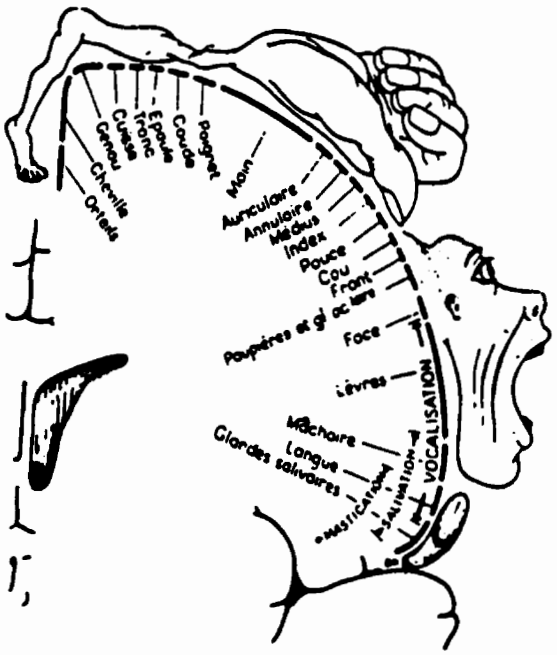
- les neurones pyramidaux,
- les neurones stellaires ou granulaires, et
- les neurones fusiformes.

Le néocortex est conventionnellement constitué par la superposition de différentes couches de neurones dont le nombre de couches permet de différencie l'allocortex (3 couches), le mésocortex (4 à 5 couches) et l'isocortex (6 couches). Ces couches sont disposées parallèlement à la surface corticale. En effet, les unités fonctionnelles de base du néocortex sont des colonnes verticales traversant la substance grise et perpendiculaires à la surface pliale. Ces colonnes ont un diamètre d'environ 1 mm et sont placées côte à côte tout en remplissant des fonctions bien précises dans l'organisation des fibres afférentes et efférentes. Leurs neurones sont agencés verticalement et forment des chaînes auxquelles contribuent les afférents, les interneurones et les efférents.

En effet, chacune des colonnes est concernée par une sous-modalité sensorielle. Elle est le terminus de fibres afférentes en provenance du thalamus ou d'autres aires corticales, et la source de fibres efférents terminant dans d'autres colonnes des aires corticales du même hémisphère (fibres d'association), dans la même aire corticale de l'hémisphère contralatérale (fibres commissurales passant à travers le corpus callosum et la commissure antérieure), et dans les noyaux subcorticaux du cervelet, du tronc cérébral et de la corde spinale (fibres de projections). En général, ce sont les neurones des couches I à IV de chaque colonne qui reçoivent et traitent les informations afférentes d'une seule sous-modalité sensorielle, alors les neurones des couches V et VI effectuent les fonctions de projection et d'association et génèrent essentiellement l'influx nerveux efférent (voir Figure 4.2). Le nombre de neurones par colonne est constant indépendamment de l'aire à laquelle cette colonne appartient.



(a) Projections somesthésiques dans la pariétale ascendante (homonculus sensitif).



(b) Organisation somato-topique dans la frontale ascendante (homonculus moteur).

Figure 4.1. Représentations corticales des différentes parties du corps (d'après [27])

	Couches	Cellules	Fonctions
I	moléculaire	horizontales Golgi II	associations intracorticales
II	granulaire externe	pyramidales petites	associations intra- et extracorticales
III	pyramidale externe	pyramidales moyennes granulaires Martinotti	associations intra- et extracorticales et commissurales
IV	granulaire interne	étoilées granulaires	associations intra- et extracorticales
V	pyramidale interne	pyramidales (Betz) granulaires (Martinotti)	projections efférentes, intra- et extracorticales et commissurales
VI	fusiforme	fusiformes granulaires Martinotti étoilées	associations intra- et extracorticales et commissurales; projections efférentes

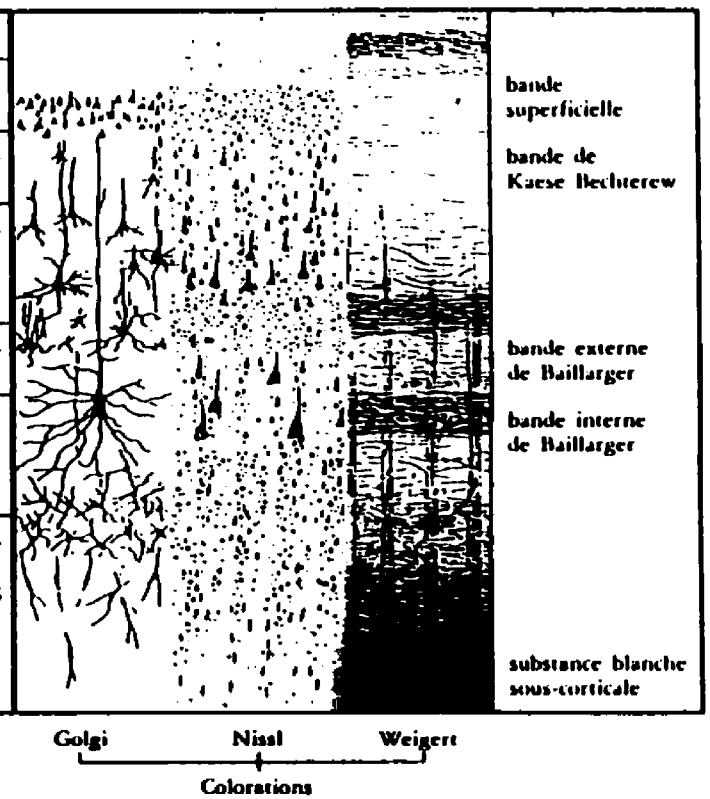


Figure 4.2. Structure du cortex cérébral (d'après [27])

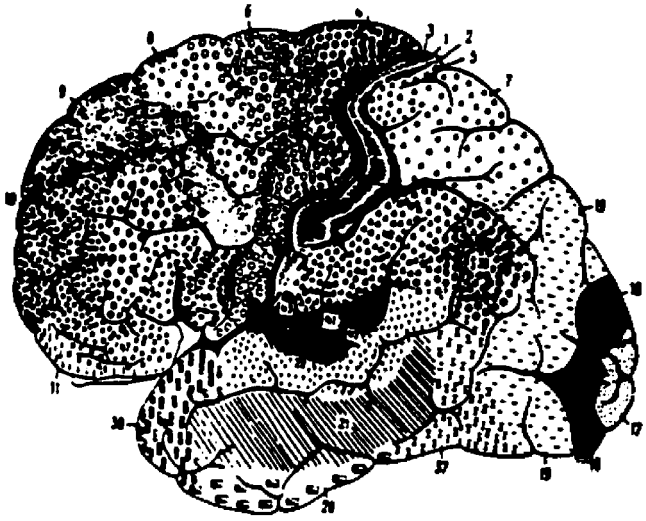
Sur le plan physiologique, le cortex cérébral est le siège du traitement des informations sensorielles et motrices, mais aussi de celui de la pensée conceptuelle, de la créativité, de la planification et de tout ce qui caractérise l'être humain. Il est en continuelle interaction avec l'hippocampe, l'amygdale et les ganglions de la base pour nous permettre d'apprécier et d'interagir avec notre environnement.

Les fonctions du néocortex ont été mises en évidence grâce à des lésions ou des simulations qui ont permis de le subdivise en plusieurs aires (voir Figure 4.3). Ainsi, on a trois groupes d'aires corticales dont:

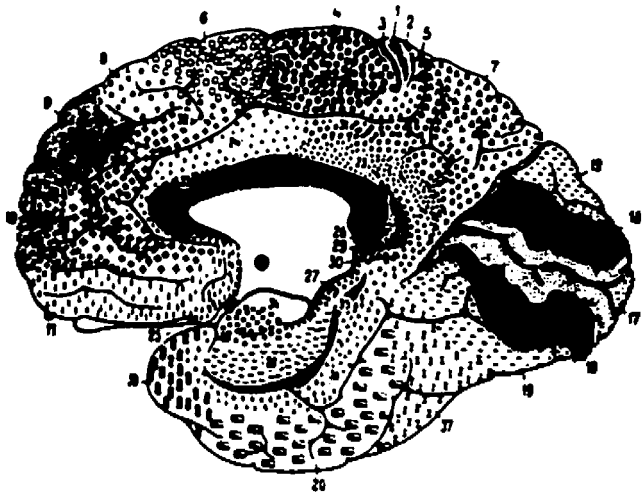
(1) les aires sensitives incluant les aires sensorielles primaires, les aires sensorielles secondaires et les aires d'associations,
(2) les aires motrices incluant l'aire motrice primaire, les aires prémotrices et les aires motrices supplémentaires, et
(3) les aires frontales.

Les aires sensitives appartiennent exclusivement aux lobes pariétaux, occipitaux et temporaux. Leur rôle est de recevoir, de décoder, d'interpréter, et de mémoriser les informations afférentes provenant des récepteurs sensorielles (senseurs visuel, auditif, gustatif, olfactif et somesthétique).

Au niveau du lobe pariétal, c'est dans l'aire pariétal primaire que les influx se distribuent selon une représentation somatotopique. Les aires pariétales 5 et 7, participent à la programmation d'activités motrices, en particulier celles qui sont en rapport avec la projection du bras vers une cible visuelle et avec toute manipulation d'objet. Ces aires associatives pariétales élaborent un système de référence spatial utile au guidage des mouvements du corps en combinant les informations visuelles et somatosensitives. Leur participation à la coordination oculo-manuelle illustre le rôle que joue le cortex pariétal entre les cortex occipital et frontal.



(a) Vue latérale gauche du cerveau humain.



(b) Vue médiane droite du cerveau humain.

Figure 4.3. Aires cytoarchitecturales du cerveau humain (d'après [27])

Le lobe occipital, composé des aires 17, 18 et 19, est impliqué principalement dans le traitement des informations visuelles. L'aire visuelle primaire 17 reçoit les signaux visuels provenant du corps génicullé latéral et de l'hémichamp visuel opposé. En ce qui concerne les aires 18 et 19, elles constituent les aires visuelles d'associations et, en traitant les informations reçues de l'aire 17, elles permettent de reconnaître et d'identifier l'objet perçu.

Le lobe temporal est impliqué dans l'affinement des informations auditives et visuelles qui s'associent pour la perception globale. Il contribue au stockage des informations importantes réutilisables à volonté. En association avec le système limbique, il prend en charge notre mémoire. D'autres informations sensorielles semblent y avoir accès également.

Les aires motrices appartiennent en grande partie au lobe frontal et sont adjacentes à l'aire somesthésique. Ces aires sont responsables du contrôle de nos mouvements. Quant aux aires associatives, elles se constituent d'aires autres que les aires sensitives et motrices. Elles occupent la partie la plus importante des aires corticales. Elles ne participent pas directement aux processus sensitifs et moteurs mais plutôt aux fonctions les plus complexes de la pensée, de l'intelligence, de la mémoire et du langage.

Le cortex associatif préfrontal comprend tout le lobe frontal sans les aires motrices et prémotrices. Il reçoit de nombreuses fibres afférentes en provenance du thalamus et de toutes les autres aires corticales. Il est impliqué dans la mémoire à court terme, le contrôle des émotions et de l'anxiété, l'éveil général ou l'attention, la pensée et le raisonnement. Il joue un rôle dans la planification des mouvements et l'initiation du comportement volontaire.

En dehors des aires sensitives auditives (41, 42, et 22), le lobe temporal possède une aire associative supérieure en rapport avec l'audition et une autre inférieure en rapport la vision. Cette dernière partie est indispensable à la classification visuelles des formes et favorise l'utilisation des données visuelles dans l'apprentissage et la mémoire. Dans l'hémisphère gauche, la partie postéro-supérieure forme l'aire de Wernicke associée à la compréhension du langage alors que l'aire correspondante de l'hémisphère droit favorise la compréhension des aspects affectifs du

langage (intonation et gestes produits par l'interlocuteur). Toute lésion de la partie gauche fait éprouver une grande difficulté à mémoriser une liste de noms alors que lorsqu'il s'agit de la partie correspondante droite, elle fait éprouver la difficulté à se souvenir des visages et à reconnaître une personne qui vient d'être vue sur une photo mélangée à d'autres.

Le cortex associatif pariétal intervient dans l'orientation spatiale en fournissant un cadre de référence aux mouvements oculaires et aux mouvements visuellement guidés dans l'espace extrapersonnel. Sa partie postérieure a des connexions neuroniques importantes avec les aires corticales sensorielles, le lobe frontal, la partie postérieure du lobe temporal, le système limbique, les ganglions de la base, le tronc cérébral et le cervelet. Les informations sensorielles relatives à l'organisation spatiale et les informations motrices convergent vers ces aires. La combinaison des données spatiales somesthésiques et visuelles fournit la base de l'orientation spatiale et de l'attention aux stimuli externes et à leur localisation. Le cortex associatif intervient également dans la planification du mouvement et dans la mémorisation des programmes moteurs. Pour comprendre le rôle des autres aires corticales non étudiées ici, il suffit de se référer à la cytocartographie commentée du cortex cérébral.

Cette sous-section nous indique que tous nos comportement et nos attitudes ont une place au niveau du cortex cérébral. Pour modéliser une aire corticale, il suffit de considérer son organisation spatiale en terme de disposition des neurones en couches et leurs interconnections. Ceci nous conduira à une clusterisation dans une même aire corticale de fonction bien connue, les informations ne sont pas traitées par tous les neurones mais plutôt chaque information ou chaque classe d'informations serait traitée par des clusters de l'aire corticale concernée.

Cependant, en neurobiologie, rien n'indique encore selon quelle séquence l'information circule entre les clusters à l'intérieur d'une aire corticale et à l'extérieur avec d'autres aires ou macrostructures. Seules des lésions permettent de définir le rôle de chaque aire mais à l'échelle microscopique, il n'est pas encore possible de décrire le mode de fonctionnement collectif des neurones.

4.2.2. Les mécanismes physiologiques de l'apprentissage et de la mémorisation dans le cerveau

Les biologistes définissent l'apprentissage par la capacité de modifier le comportement en fonction de l'expérience, alors que la mémoire est vue comme étant la capacité de conserver cette modification pendant une certaine période [28]. La forme d'apprentissage et de mémorisation la plus répondue est l'habituation.

L'habituation désigne une diminution de l'intensité de la réponse comportementale. Elle est d'une simplicité remarquable et implique à la fois une mémoire à court terme et une mémoire à long terme. L'habituation à court terme entraîne une réduction transitoire de l'efficacité synaptique en modifiant l'intensité des connexions entre les cellules sensorielles et leurs principales cellules cibles (les interneurones et les neurones moteurs).

De son côté, l'habituation à long terme affecte l'efficacité synaptique de façon plus permanente. Elle engendre une modification plus profonde et durable, et un blocage fonctionnel de la plupart des jonctions auparavant efficaces. Par ailleurs, les chercheurs ont montré qu'un apprentissage de courte durée suffit pour engendrer une modification profonde de la transmission synaptique dans les synapses dont l'importance est capitale pour l'apprentissage. Ainsi, l'habituation à court terme et l'habituation à long terme peuvent s'exercer sur un site commun et sont deux aspects d'un même mécanisme cellulaire [28].

Avec ces idées, nous terminons notre revue et étude du réseau neuronique biologique. Nous constatons que la compréhension des mécanismes de l'apprentissage et de la mémorisation dans le cerveau est encore à ses débuts, ou comme disent les biologistes et les neurologues " à l'état embryonnaire".

Toutefois, cette étude démontre qu'il existe dans le cerveau une organisation stupéfiante du traitement de l'information qui pourrait être à l'origine des phénomènes d'apprentissage et de mémoire. Il s'en suit que pour améliorer ces caractéristiques dans les réseaux neuroniques

artificiels, il faut utiliser des réseaux plus grands qui incorporent une certaine capacité d'auto-organisation pour le traitement du flux d'information à l'intérieur même des réseaux. Cela peut s'effectuer en employant des mécanismes de classification et de contrôle des informations dans le réseau.

4.3. RÉSEAU PERCEPTRON CLUSTERISÉ

L'idée principale des modifications que nous proposons d'apporter au perceptron multicouche standard consiste à ajouter à ce dernier une capacité d'auto-organisation et des mécanismes de contrôle similaires à ceux qu'on retrouve dans le réseau biologique. Pour atteindre cet objectif, nous apportons des modifications à l'algorithme d'apprentissage du réseau perceptron multicouche standard ainsi qu'à son architecture (voir Figure 4.4). Une autre solution a été proposé par R. Azouzi [15], et consiste à ajouter un réseau secondaire à auto-organisation.

Dans la nouvelle architecture du perceptron multicouche, les neurones ne sont plus alignés, mais ils sont dispersés aléatoirement dans un plan. Chaque neurone i a une position (abs_i, ord_i) et est relié aux différents neurones de la couche précédante (ou aux entrées, s'il s'agit de la première couche) à l'aide d'un vecteur "prototype" de paramètres $\vec{v}_i$:

$$\vec{v}_i = \left[v_{i,1}, v_{i,2}, \cdots v_{i,p}, \cdots v_{i,k}\right]^T \qquad (4\text{-}1)$$

À noter que chaque composante $v_{i,p}$ du vecteur $\vec{v}_i$ est associée à la connexion reliant le neurone i au neurone p (ou à l'entrée p). Dans l'équation (4-1), k est le nombre de neurones sur la couche précédente (ou le nombre d'entrées).

À sa sortie, un neurone i fournit une mesure de la distance euclidienne D_i entre le vecteur prototype qui lui est associé et le vecteur $\vec{o} = \left[o_1, o_2, \cdots o_p, \cdots o_k\right]^T$ (ou $\vec{x} = \left[x_1, x_2, \cdots x_p, \cdots x_k\right]^T$ si i appartient à la première couche), où o_p est la sortie du neurone

p de la couche précédante. L'idée a été inspirée de l'algorithme d'apprentissage du réseau de Kohonen (voir chapitre II). La distance D_i est donnée par:

$$D_i = \|\bar{o} - \bar{v}\| = \sqrt{\sum_{p=1}^{k} \left(o_p - v_{i,p}\right)^2} \qquad (4\text{-}2)$$

On définit le neurone gagnant comme celui qui a la distance D_i minimale parmi les neurones de la même couche. Les coordonnées de chaque neurone sont fixées au début, alors que les valeurs des paramètres $v_{i,p}$ sont initialisés aléatoirement. Ensuite, ils seront ajustés au fur et à mesure que le réseau s'entraîne. Les étapes de l'algorithme d'apprentissage du réseau perceptron clusterisé sont présentées dans la prochaine section.

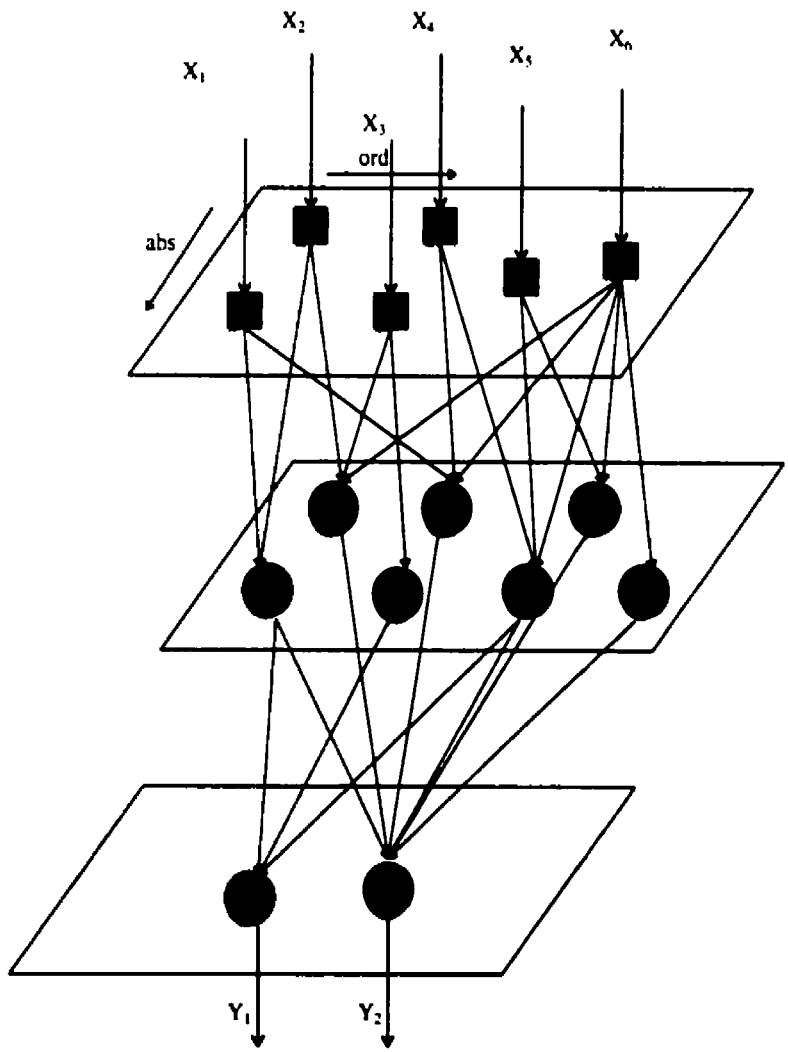


Figure 4.4. Nouvelle architecture du réseau perceptron multicouche

Chaque neurone j est caractérisé par:

- Un vecteur de poids $\vec{w}:[w_{ij}]$
- Un vecteur prototype $\vec{v}:[v_{kj}]$ $\quad |\vec{w}| = |\vec{v}|$ (4-3)
- Un écart θ_j
- Des coordonnées: abs_j et ord_j

4.3.1. Algorithme d'apprentissage du réseau perceptron clusterisé

Soient k_1 et k_2 les nombres de neurones sur la couche courante et sur la couche précédante respectivement. Pour tous i et j, tel que $1 \le j \le k_1$ et $1 \le i \le k_2$, les étapes d'apprentissage du réseau sont:

1) Initialiser les poids, les vecteurs prototypes et les écarts
2) Présenter les données X_i à l'entrée du réseau et mettre à jour les vecteurs prototypes

- Sur la première couche cachée adjacente aux entrées X_i

$$I_j = \varphi_j \sum_i w_{ij} x_i C_{ij} + \theta_j \qquad (4\text{-}4)$$

$$o_j = 1/(1+e^{-I_j}) \qquad (4\text{-}5)$$

$$v_{ij}(n+1) = v_{ij}(n) + \eta_1 [x_i - v_{ij}(n)] G_j \qquad (4\text{-}6)$$

- Sur les couches intermédiaires

$$I_j = \varphi_j \sum_i w_{ij} o_i C_{ij} + \theta_j \qquad (4\text{-}7)$$

$$o_j = 1/(1+e^{-I_j}) \qquad (4\text{-}8)$$

$$v_{ij}(n+1) = v_{ij}(n) + \eta_1 [o_i - v_{ij}(n)] G_j \qquad (4\text{-}9)$$

- Sur la couche de sortie

$$I_j = \varphi_j \sum_i w_{ij} o_i C_{ij} + \theta_j \qquad (4\text{-}10)$$

$$y_j = 1/(1+e^{-I_j}) \qquad (4\text{-}11)$$

$$v_{ij}(n+1) = v_{ij}(n) + \eta_1 [o_i - v_{ij}(n)] G_j \qquad (4\text{-}12)$$

3) Évaluer les valeurs nécessaires de correction dans le réseau à partir de l'erreur de sortie.

- Sur la couche de sortie

$$\delta_j = y_j(1-y_j)(y_{dés_j} - y_j) \qquad (4\text{-}13)$$

- Sur les couches intermédiaires

$$\delta_j = o_j(1-o_j)\sum_k \delta_k \varphi_k \left[w_{kj} C_{kj} + \left[C_2^{\cdot}(o_j - v_{kj}) / \log \varphi_k \right] \sum_i w_{ki} o_i C_{ki} \right] \qquad (4\text{-}14)$$

4) Ajuster les coefficients synaptiques

$$w_{ij}(n+1) = w_{ij}(n) + \eta \delta_j o_i + \alpha [w_{ij}(n) - w_{ij}(n-1)] \qquad (4\text{-}15)$$

5) Ajuster les écarts

$$\theta_j(n+1) = \theta_j(n) + \eta \delta_j + \alpha [\theta_j(n) - \theta_j(n-1)] \qquad (4\text{-}16)$$

6) Répéter les étapes 2 à 5 pour toutes les données du fichier d'apprentissage
7) Répéter les étapes 2 à 6 pour autant de passes qu'il faut jusqu'à ce que l'erreur maximale au niveau de chaque sortie soit inférieure à la valeur de la tolérance.

Si on utilise l'équation (2.4') du chapitre II, les équations de l'étape 3 de l'algorithme d'apprentissage seront remplacées par:

- Sur la couche de sortie

$$\delta_j = 1/2(1-y_j^2)(y_{desj} - y_j) \qquad (4\text{-}17)$$

- Sur les couches intermédiaires

$$\delta_j = 1/2(1-o_j^2)\sum_k \delta_k \varphi_k \left[w_{kj} C_{kj} + \left[C_2^2 (o_j - v_{kj}) / \log \varphi_k \right] \sum_i w_{ki} o_i C_{ki} \right] \qquad (4\text{-}18)$$

Plus de détails sur les calculs des dérivées et du δ_j se trouvent dans l'annexe.

D'un point de vue neurobiologique, il est reconnu que l'interaction entre les neurones se fait comme suit: lorsqu'un neurone particulier répond à un signal d'entrée, il tend à exciter les neurones voisins de telle manière que le voisin le plus éloigné reçoit le moins d'excitation. Pour s'approcher le plus possible du fonctionnement du réseau biologique, nous avons établi les coefficients C_{ij}, G_j et φ_j qui sont respectivement, le coefficient de clusterisation spatiale relatif au neurone i et au neurone j, le coefficient d'interaction entre le neurone gagnant et le neurone j, et le coefficient de clusterisation par les données du neurone j. Ces coefficients sont données par:

$$C_{ij} = e^{-C_1 d_{ij}} \qquad (4\text{-}19)$$

C_1: est une constante qui dépend des données à apprendre

d_{ij}: est la distance physique euclidienne entre le neurone i (abs_i, ord_i) et le neurone j (abs_j, ord_j).

Elle est donnée par:

$$d_{ij} = \sqrt{(abs_i - abs_j)^2 + (ord_i - ord_j)^2} \qquad (4\text{-}20)$$

$$\varphi_j = e^{-C_2 D_j^2} \qquad (4\text{-}21)$$

C_2: est une constante qui dépend des données à apprendre

D_j: est la distance euclidienne entre le vecteur prototype du neurone j et les sorties des neurones de la couche précédante

$$G_j = e^{-C_3 d_{ig}^2} \qquad (4\text{-}22)$$

C_3: est une constante qui dépend des données à apprendre

g : est l'indice du neurone gagnant, c'est à dire le neurone qui a la distance D_i minimale parmi les neurones de sa couche.

Une illustration de l'équation (4-21) est présentée à la Figure 4.5.

Avec de la pratique et un bon jugement, il est possible d'obtenir des valeurs optimales pour les constantes C_1, C_2 et C_3. Pour obtenir une bonne convergence, il est recommandé généralement d'utiliser une petite valeur de C_2 (au voisinage de 0.0001) et des valeurs de C_1 et C_3 dans l'intervalle [1, 3]. Notons que lorsque $C_1 = 0$ et $C_2 = 0$, le fonctionnement du réseau se réduit à celui d'un perceptron multicouche standard.

Les coefficients de clusterisation C_{ij} et φ_j permettent de diviser le réseau perceptron en plusieurs clusters. Chaque cluster est spécifié par la hiérarchie de neurones qui commence à partir d'un ou plusieurs neurones de la première couche et tous les neurones supérieurs qui lui sont connectés directement ou indirectement en remontant à travers le réseau. À remarquer que plus le nombre de couches est élevé, plus les clusters se chevauchent. Cela signifie qu'il n'y a pas de frontières réelles entre les clusters. Ceci permettra de préserver la nature distribuée du traitement de l'information. Celle-ci est une propriété fondamentale des réseau neuroniques.

Le coefficient de clusterisation spatiale C_{ij} et le coefficient de clusterisation par les données φ_j une fois calculés, fixent la part de la contribution de chaque signal d'entrée au neurone j dans le signal d'activation de ce dernier. Ainsi, plus un coefficient C_{ij} ou φ_j est élevé, plus la connexion au neurone associé est excitatrice. De la même façon, plus le coefficient est faible, plus la connexion est inhibitrice. Les coefficients de clusterisation imposent dans une certaine mesure le comportement de chaque neurone vis-à-vis les données d'entrées. Implicitement, la sensibilité du réseau aux valeurs initiales des poids est réduite. Notons que lorsque $C_{ij} = 1$ et $\varphi_j = 1$, le fonctionnement du réseau se réduit à celui d'un perceptron multicouche standard.

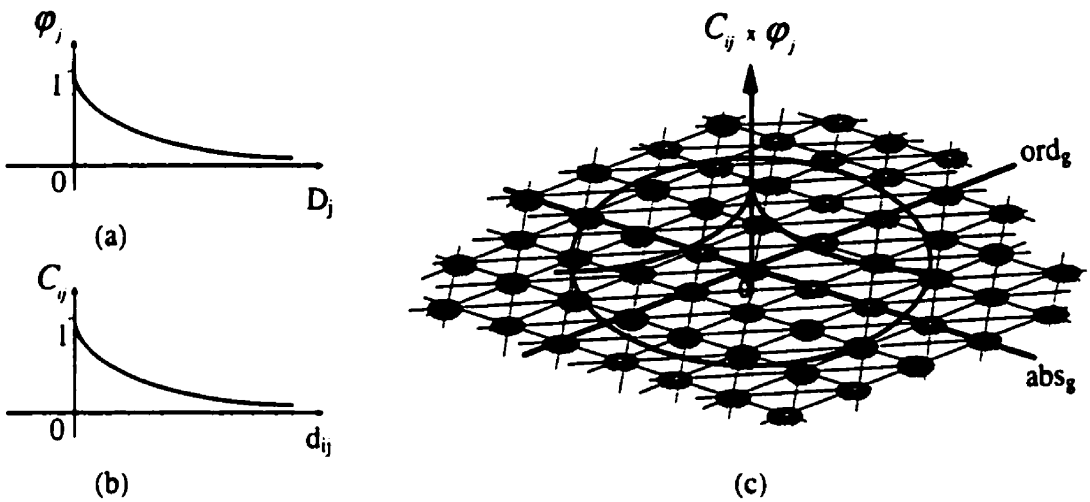


Figure 4.5. (a) Le coefficient de clusterisation spatiale, (b) le coefficient de clusterisation par les données, (c) le coefficient de clusterisation pour l'entraînement du réseau perceptron clusterisé

Pour la suite du texte, un réseau perceptron clusterisé désigne un réseau perceptron qui a l'architecture de la Figure 4.4 et qui est entraîné par l'algorithme présenté ci-dessus. Un réseau perceptron standard est un réseau perceptron multicouche entraîné avec l'algorithme de rétropropagation présenté au chapitre II.

4.3.2. Entraînement du réseau perceptron clusterisé

4.3.2.1. Entraînement hors-ligne du perceptron clusterisé

Même si l'algorithme d'apprentissage ainsi que l'architecture du perceptron multicouche ont été modifiés, le principe de présenter des données au réseau pour l'apprentissage reste le même. En effet, chaque exemplaire présenté au réseau doit être composé d'un vecteur d'entrées $\vec{x}$ et d'un vecteur de sorties ou de réponses désirées $\vec{y}$. À ce stade, le réseau perceptron clusterisé possède, pratiquement, les mêmes avantages et limites du réseau perceptron multicouche standard pour de la vitesse de convergence (nombre d'itérations). On parle d'une clusterisation spatiale lorsque $\varphi_j = 1$, $\forall$ j et que C_{ij} est variable. Dans le cas inverse, c'est-à-dire lorsque $C_{ij} = 1$, $\forall$ j et que φ_j est variable, il s'agit d'une clusterisation par les données. Si $\varphi_j \neq 1$, et $C_{ij} \neq 1$ $\forall$ j, on parle alors d'une clusterisation mixte.

4.3.2.2. Entraînement continu du réseau perceptron clusterisé

Le rôle le plus important qui sera joué par l'algorithme d'apprentissage proposé dans le présent mémoire, sera de contribuer à améliorer la capacité de mémorisation du réseau perceptron multicouche en adaptation continue. L'entraînement en continu du réseau clusterisé peut être effectué en utilisant le même algorithme présenté ci-dessus. Cependant, une modification des paramètres des coefficients de clusterisation serait souhaitable. En effet, lorsqu'un réseau perceptron clusterisé doit apprendre de façon permanente, il est préférable d'adapter légèrement les poids seulement pour les neurones se trouvant dans la zone vers laquelle pointe la nouvelle donnée. En conséquence, pour localiser les changements et réduire l'ampleur de l'interaction entre les neurones voisins, il suffit d'augmenter les valeurs de C_2 et C_3 respectivement, dans les équations (4-21) et (4-22). Ceci contrôle le rayon de voisinage d'un neurone. L'adaptation en continu du réseau perceptron clusterisé entraîne en réalité un léger changement dans la répartition des clusters (le nombre de clusters peut augmenter ou le nombre de neurones par cluster peut être modifié). D'après notre expérience, même si on garde les mêmes valeurs pour les coefficients de clusterisation (C_2 et C_3) pendant l'apprentissage en continu, on obtient aussi des bons résultats.

4.3.3. Étude des performances du réseau perceptron clusterisé et comparaison avec le réseau perceptron standard

L'évaluation du réseau perceptron clusterisé est basée sur l'apprentissage a priori et en continu de la fonction E(x) qui retourne la valeur entière de x. Le choix de cette dernière est justifié par sa nature clusterisée (voir Figure 4.6). En effet, ceci nous permet de mieux visualiser le phénomène de clusterisation au sein du réseau.

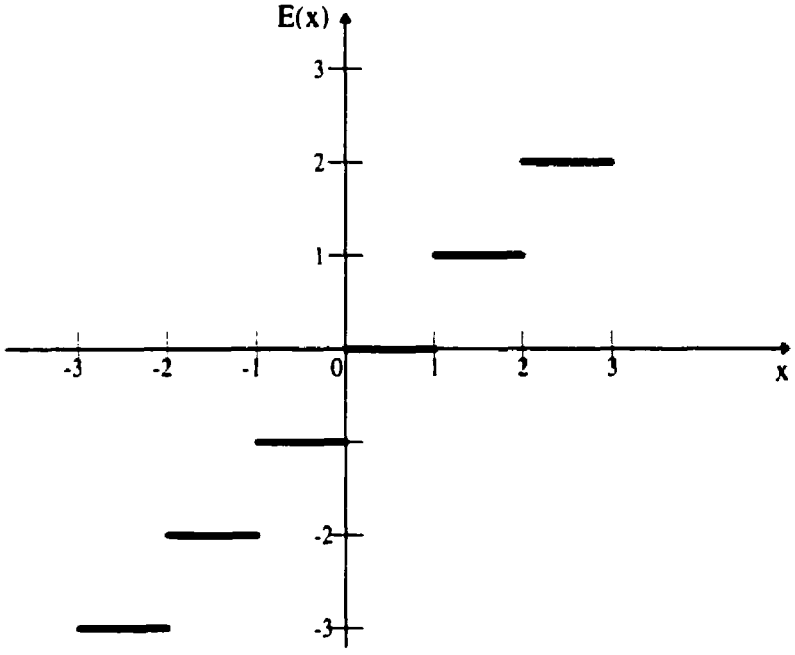


Figure 4.6. Représentation graphique de la fonction E(x)

Dans les tests suivants, la performance du réseau perceptron clusterisé est comparée à celle du perceptron multicouche standard.

4.3.3.1. Tests d'entraînement a priori

La précision et la vitesse d'apprentissage du réseau perceptron clusterisé de dimensions 2 x 9 x 5 x 1, sont comparées à celles du réseau perceptron standard de mêmes dimensions. Ces

réseaux sont entraînés en utilisant les exemplaires d'Entrées/Sorties du Tableau 4.1 et les paramètres d'entraînement ainsi que les valeurs des coefficients de clusterisation indiqués dans le Tableau 4.3. On calcul la partie entière de X_2. Pour les fins de comparaison, nous avons ajouté une autre entrée (X_1) qui prend, la valeur 1 si l'entrée X_2 est négative et 2 lorsque X_2 est positive.

La notation 2 x 9 x 5 x 1 veut dire que nous avons 2 entrées, une première couche cachée à 9 neurones, une deuxième couche cachée à 5 neurones, et une seule sortie.

Le Tableau 4.3 montre les sommes des erreurs au carré obtenus au niveau des sorties des réseaux en utilisant l'ensemble des données d'entraînement (SE_0).

Tableau 4.1. Données d'entraînement a priori

Entrée X_1	Entrée X_2	Sortie Y ($E(X_2)$)
1	-1	-1
1	-0.75	-1
1	-0.66	-1
1	-0.25	-1
2	0	0
2	0.25	0
2	0.5	0
2	0.75	0
2	1	1
2	1.25	1
2	1.5	1
2	1.75	1

En général, avec la clusterisation spatiale le réseau apprend moins rapidement que les deux autres types de clusterisation et le réseau perceptron standard. Cependant, pour la clusterisation mixte et celle par les données, leurs vitesses de convergence par rapport à celle du perceptron standard dépendent en partie des valeurs initiales des poids. Dans notre cas, les valeurs des poids sont initialisées d'une façon aléatoire, ce qui ne nous permet pas de juger lequel des réseaux est meilleur en terme de nombre d'itérations. Toutefois, les vitesses de convergence des deux types de clusterisation sont en général très proches de celle du perceptron standard.

Tableau 4.2. Données d'entraînement en continu

Entrée X_1	Entrée X_2	Sortie Y ($E(X_2)$)
1	-0.33	-1
1	-0.5	-1
2	0.33	0
2	0.9	0
2	1.33	1
2	1.66	1
1	-0.05	-1
1	-0.8	-1
2	0.15	0
2	0.85	0
2	1.15	1
2	1.4	1

Tableau 4.3. Résultats des entraînements a priori et en continu

Type du réseau	Nb. Itérations (10^4)	SE_0 (10^{-6})	SEc
Perceptron standard	33.6	5.78	7.63
Clusterisation spatiale $C_1 = 3$, $C_2 = 0$, $C_3 = 0$ et $\eta_1 = 0.9$	170	7.71	6.40
Clusterisation par les données $C_1 = 0$, $C_2 = 0.001$, $C_3 = 3$ et $\eta_1 = 0.9$	16.4	6.68	2.40
Clusterisation mixte $C_1 = 1$, $C_2 = 0.0005$, $C_3 = 1$ et $\eta_1 = 0.9$	43.3	5.25	0.70
SE_0: somme des erreurs au carré des données d'entraînement a priori. SE_C: somme des erreurs au carré des données d'entraînement a priori et en continu. Tous les réseaux ont été entraînés avec les paramètres suivants: $\eta = 0.9$, $\alpha = 0.9$ et tolérance = 0.00101.			

D'après notre expérience, l'utilisation de la fonction d'activation de l'équation (2-4') permet aux réseaux de converger plus vite que dans le cas de l'équation (2-4). La différence par rapport à cette dernière, est qu'il faut choisir une petite valeur du gain (au voisinage de 0.1), et une grande valeur de momentum (au voisinage de 0.9). Dans le cas de l'exemple ci-dessus, la fonction d'activation utilisée est celle de l'équation (2-4).

4.3.3.2. Étude de la clusterisation

Pour étudier la clusterisation au sein des différents réseaux, nous avons vérifié, dans un premier temps, l'effet de l'élimination d'un neurone donné du réseau sur le résultat en sortie. Dans notre exemple, nous avons quinze neurones (9 +5 + 1) numérotés de zéro à quatorze. Le Tableau 4.4 présente les résultats obtenus. Chaque numéro dans le Tableau 4.4 veut dire que l'enlèvement du neurone associé génère une dégradation au niveau de la sortie, du réseau en question, de plus que 20 % pour toutes les données appartenant à l'intervalle indiqué en haut de la même colonne. Par exemple, si nous éliminons le neurone 9 (premier neurone de la deuxième couche cachée), dans le cas du réseau avec clusterisation spatiale, tous les résultats associés aux données appartenant aux intervalles [0, 1[et [1, 2[de X_2 seront incorrectes.

Tableau 4.4. Résultat de l'étude de la clusterisation au sein des réseaux

Type du réseau	N° des neurones [-1, 0[	N° des neurones [0, 1[	N° des neurones [1, 2[
Perceptron standard	--	--	10
Clusterisation spatiale C1=3, C2=0, C3=0	0 et 10	9 et 10	1 et 9
Clusterisation par les données C1=0, C2=0.001, C3=3	--	6 et 12	10
Clusterisation mixte C1=1, C2=0.0005, C3=1	9 et 10	0 et 10	1

Un cluster regroupe tous les neurones qui affectent directement et d'une façon non négligeable les résultats d'un même ensemble de données. Dans notre exemple, nous avons trois clusters, un pour chaque intervalle. En général, quel que soit le type de clusterisation, nous avons au moins un neurone par cluster. Ce qui n'est pas le cas pour le réseau perceptron standard.

Ces résultats nous montrent bien qu'il s'agit d'une organisation au sein d'un réseau perceptron multicouche entraîné par l'algorithme d'apprentissage présenté à la sous-section 4.2.1.

4.3.3.3. Tests d'entraînement continu

Les tests effectués ci-dessus démontrent la capacité du réseau perceptron clusterisé à modéliser une fonction lorsque toutes les données sont disponibles a priori. Cependant, un système de neurocommande adaptative nécessite non seulement un bon modèle, mais aussi un modèle capable d'apprendre progressivement de nouveaux exemplaires avec le minimum de perte d'informations sur les anciens exemplaires.

Puisqu'avec la méthode d'adaptation récursive une seule itération d'entraînement est effectuée à chaque itération de commande, un important nombre d'itérations d'entraînement peuvent être effectuées ce qui affecte globalement la mémoire du réseau et détruit le modèle au niveau des autres régions.

Le réseau perceptron clusterisé est organisé de façon à démontrer une grande résistance vis-à-vis le problème de dégradation des informations préenregistrées. En conséquence, les tests d'entraînement en continu vont quantifier cette résistance, en utilisant dix itérations et ce pour chaque nouvel exemplaire d'E/S présenté au réseau. L'entraînement en ligne commencera avec les modèles du Tableau 4.3 (modèles entraînés a priori).

Les douze exemplaires du Tableau 4.2 sont présentés séquentiellement aux réseaux. À chaque étape, dix itérations d'entraînement sont effectuées sur le nouvel exemplaire. Le Tableau 4.3 présente l'erreur globale (SE_C), soit l'erreur obtenue avec tous les exemplaires des tableaux 4.1 et 4.2. Tous les réseaux ont gardé les mêmes valeurs des paramètres avec lesquels ils ont été entraînés a priori.

A partir du Tableau 4.3, il est clair que les meilleurs résultats sont obtenus avec les réseaux perceptrons clusterisés. Ces derniers améliorent graduellement leurs performances avec l'apprentissage séquentiel des nouvelles données (en particulier celui avec la custerisation mixte). Ces performances sont légèrement dégradées pour les données apprises à priori. Cependant, avec le réseau perceptron standard, au fur et à mesure que ce dernier est entraîné avec de nouveaux

exemplaires, ses performances avec les anciennes deviennent de plus en plus mauvaises et cela est facilement observable. C'est le problème général d'interférence ou "d'oubli catastrophique" associé aux réseaux perceptrons standards entraînés avec les algorithmes conventionnels. Il est clair que ce problème est non désiré au sein des systèmes de neurocommande adaptative.

La plupart des tests effectués avec des réseaux perceptrons standards et clusterisés, montrent que les valeurs initiales des poids influencent en partie les résultats lors de l'apprentissage en continu. Avoir une petite valeur de SE_0 lors de l'apprentissage a priori, n'est pas une condition suffisante pour garantir une petite valeur de la somme des erreurs au carré après l'apprentissage en continu (SE_C). Le Tableau 4.5 présente les résultats obtenus pour des réseaux de taille 2 x 17 x 9 x 1 entraînés, a priori avec le cosinus des angles 0, 45, 90, 105, 120, 180, 270 et 315, et en continu avec le sinus des mêmes angles (sauf 105 et 120). D'après les résultats obtenus, on remarque bien que les valeurs de SE_C des réseaux clusterisés sont les plus petites. Ceci peut être expliqué par le fait que les réseaux clusterisés sont moins sensibles aux valeurs initiales des poids. En plus, les réseaux perceptrons clusterisés trouvent leurs performances dans le cas où les données utilisées dans l'apprentissage en continu sont très différentes de celles de l'entraînement a priori.

Un autre résultat important, est que plus la taille du réseau est grande plus les performances des réseaux perceptrons clusterisés sont bonnes. Par exemple, avec les mêmes paramètres utilisés dans la clusterisation mixte, si nous utilisons un réseau de taille 2 x 17 x 9 x 1 la somme des erreurs au carré SE_C se réduit à 0.61 (la valeur de SE_0 est $6.03\ 10^{-6}$). Ce dernier résultat montre que plus la taille du réseau est grande plus l'effet de la clusterisation est prouvé.

En conclusion, les résultats obtenus démontrent que les réseaux perceptrons clusterisés possèdent la même capacité de modélisation que celle des perceptrons standards. Cependant, leurs capacités de mémorisation en adaptation continue sont nettement supérieures. Ces capacités rendent le réseau perceptron clusterisé très prometteur pour la commande adaptative des procédés en général.

Tableau 4.5. Résultats de SE_C en fonction de SE_0

Perceptron standard		Perceptron clusterisé $C_1 = 3, C_2 = 0, C_3 = 0$ et $\eta_1 = 0.9$	
SE_0 (10^{-5})	SE_C	SE_0 (10^{-5})	SE_C
1.69	9.32	1.79	2.02
0.88	22.70	0.85	9.11
Les deux réseaux ont été entraînés avec les paramètres suivants: $\eta = 0.9$, $\alpha = 0.9$ et tolérance = 0.00101.			

4.4. CONCLUSION

Dans ce chapitre, nous avons présenté un résumé sur les plus importantes et récentes informations concernant le réseau neuronique biologique. À partir de cette revue, nous avons proposé des améliorations au réseau perceptron multicouche standard dans le but d'améliorer ses capacités d'adaptation et d'apprentissage. La comparaison du réseau perceptron standard avec le réseau perceptron clusterisé prouve cette amélioration. Dans le prochain chapitre, nous allons utiliser le réseau perceptron clusterisé pour développer une commande adaptative pour le contrôle de la température dans le laboratoire de métrologie industrielle du département de génie mécanique de l'Université Laval.

CHAPITRE V

COMMANDE DE TEMPÉRATURE

5.1. INTRODUCTION

La commande neuronale (régulation et asservissement) est une appellation qui englobe une vaste gamme de systèmes servant à maintenir des paramètres tels que la température à une valeur prescrite dans toutes les conditions de service. L'usage de la commande neuronale est peu répandu dans les secteurs industriels, commerciaux et institutionnels. La commande neuronale procure de nombreux avantages et les raisons motivant son emploi varient avec les particularités des applications.

Pour posséder un bon système de commande et ainsi obtenir les avantages de la commande neuronale, les éléments qui composent le système sont d'une très grande importance. Les éléments composant une boucle de rétroaction sont: le senseur, l'actionneur, le régulateur et l'organe de commande. Dans cette étude, l'emphase sera mise sur le régulateur neuronal.

Le procédé étudié est un laboratoire dont nous voulons contrôler la température à partir d'un calculateur numérique. L'utilisation du régulateur neuronal a pour but de maintenir la température uniformément proche de la consigne dans les différentes sections du laboratoire de

métrologie industrielle (LMI) du département de génie mécanique de l'Université Laval (voir Figure 5.1).

Dans le problème de commande du procédé pour le contrôle de la température, aucun comportement spécifique n'est imposé aux variables d'entrées du procédé. En fait, le problème de régulation est fondamentalement celui de maintenir la température proche de la valeur de la consigne. L'adaptation pourrait devenir très importante, dépendamment de la manière dont le procédé est modélisé. En effet, il est difficile de déterminer le modèle approprié pour le contrôle de la température à cause des non-linéarités qui existent entre les variables d'entrées et les variables de sorties. Dans ces conditions, l'utilisation d'un réseau neuronique possédant de bonnes capacités d'entraînement en continu serait très pertinente pour bien combiner l'adaptabilité et la commande.

Dans plusieurs investigations des systèmes de commande, le procédé commandé est supposé statique. Cette hypothèse peut être justifiée si le temps de changement des variables de commande est suffisamment grand pour permettre au procédé de se stabiliser. Le système de commande pourrait être décrit à l'aide d'une relation typique de la forme:

$$\vec{U}(t) = \text{fonction(consigne, } \vec{T}(t),...,\vec{T}(t-n),\vec{U}(t-1),...,\vec{U}(t-n-1),\vec{v}) \quad (5\text{-}1)$$

où $\vec{v}$ est un vecteur de paramètres inconnus et qui peuvent changer en fonction du temps, $\vec{T}(t)$ et $\vec{U}(t)$ désignent respectivement le vecteur de température et le vecteur de commande à l'instant t.

D'après le modèle (5-1), on constate donc que si le procédé présente des variations, l'adaptation du modèle de commande en continu prend toute son importance. Le système de commande doit posséder de bonne capacité d'adaptation pour pouvoir suivre ces variations en ajustant les paramètres du vecteur $\vec{v}$ du modèle.

En conséquence, nous proposons dans ce qui suit, une nouvelle approche de neurocommande adaptative pour le contrôle de la température dans le laboratoire de métrologie industrielle (LMI). Cette approche consiste en un système de neurocommande dans lequel un réseau perceptron clusterisé est utilisé pour modéliser le système de commande. L'adaptation du système de neurocommande est assurée simplement par l'entraînement en continu du réseau perceptron clusterisé.

5.2. LE LABORATOIRE LMI

Le LMI est un laboratoire d'étalonnage de matériel d'essai. Il a comme rôle d'étalonner et d'ajuster le matériel d'essai, de mesurage et de diagnostic destiné à l'essai, à la fabrication, à l'entretien, etc, de produits. Il possède les étalons de travail et les systèmes d'étalonnage nécessaires pour réaliser un étalonnage conforme à des spécifications/tolérances données, habituellement celles d'un fabricant ou une norme publiée, et est capable d'établir des rapports d'exactitude valides. Le LMI dispose aussi des moyens pour contrôler ses étalons de travail entre les étalonnages, ainsi que des conditions environnementales appropriées. Il publie des résultats de mesure et indique si le matériel d'essai se situe dans les limites des spécifications/tolérances.

Le tableau 5.1 présente les conditions environnementales du LMI. Le système actuel de contrôle de la température dans le LMI consiste en un PID (Proportional, integral and derivative) qui commande un système de chauffage pour augmenter ou diminuer la température. Le senseur utilisé est un RTD (Resistance Temperature Device) situé au centre du laboratoire. Les paramètres du PID sont fixés à des valeurs afin de maintenir la température du LMI à 20°C . Ce système de contrôle permet d'avoir une température très proche de la consigne au niveau du centre du laboratoire (aux alentours du RTD). Toutefois, la température dans le reste du laboratoire varie et la différence par rapport à la consigne peut dépasser 1°C. En effet, le rôle du régulateur neuronal, que nous voulons installé, est d'essayer de rapprocher le plus possible la température, au niveau des différentes sections du LMI, de la consigne. La température désirée est celle de 20°C à 0.1 près.

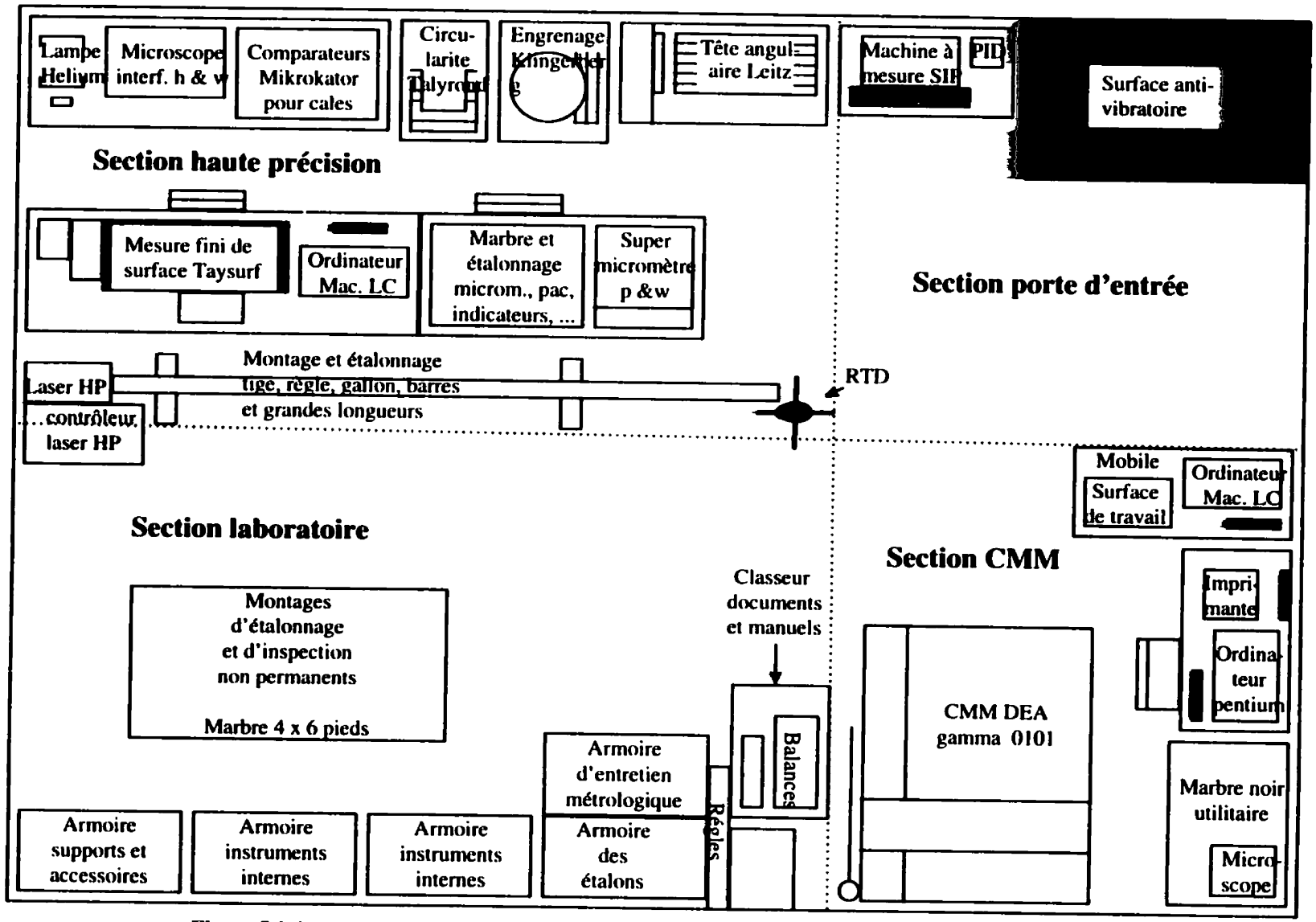


Figure 5.1. Laboratoire de métrologie industrielle du département génie mécanique de l'Université Laval

Tableau 5.1. Les conditions environnementales du LMI (S.O. # 700035)

Température	68° F ± 1/4° F. De 30 à 60 pouces du plancher
Humidité	50 % H.R. Maximum
Dépoussiérage	80 - 85 % utilisant l'épreuve D.O.P. de 0.3 Micron
Changement d'air	15 / HR. @ 1600 pieds cubes / Min.
Apport d'air frais	15 % @ 240 pieds cubes / Min.
Nombre de personnes	5 Maximum
Charge thermique	1500 Watts Maximum
Éclairage	100 Lumens par pied carré Minimum
Pression ambiante	0.02 à 0.01 pouces d'eau

Dans la prochaine section, une description du montage est effectuée.

5.3. APPAREILLAGE

Le montage expérimental utilisé pour l'acquisition des réponses des senseurs et la génération des commandes, est détaillé à la Figure 5.2. À l'aide de ce montage, il est possible de mesurer la température aux différentes sections du laboratoire, et d'agir par l'intermédiaire des actionneurs, pour maintenir la température proche de la consigne (20°C). Les différents senseurs et actionneurs sont reliés à un système d'acquisition et de commande.

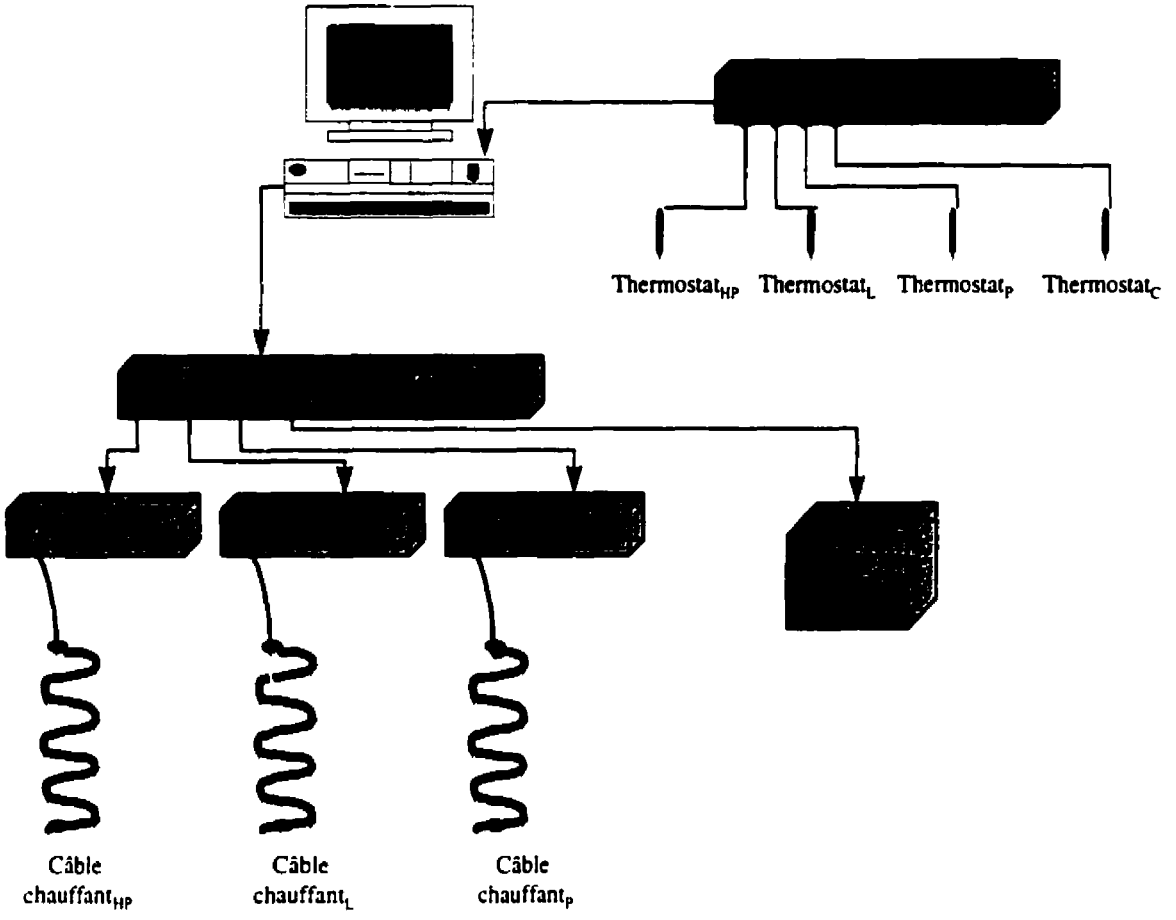


Figure 5.2. Dispositif expérimental

Un calculateur numérique, seul, ne peut pas obtenir des informations des senseurs et il ne peut pas communiquer non plus avec l'extérieur pour envoyer des commandes. Les plupart des senseurs et des actionneurs sont des appareils fonctionnant avec des signaux électriques analogiques. La présence d'une carte d'acquisition et de commande pour effectuer la communication entre l'extérieur et l'unité de contrôle du calculateur numérique est essentielle. Dans notre montage, ce dernier est principalement un micro-ordinateur 486 DX muni d'une carte d'acquisition AT-MIO-16E-10 de chez National Instrument et d'un logiciel d'acquisition et de commande appelé LabVIEW qui est un langage de programmation graphique spécifiquement destiné au contrôle et à la supervision d'instruments.

LabVIEW est basé sur l'utilisation d'un logiciel de " drivers" appelé NI-DAQ. La Figure 5.3 représente les liens LabVIEW, NI-DAQ et la carte d'acquisition. Pour l'acquisition des données et la génération des commandes, il est nécessaire de réaliser un programme qui doit pouvoir:

- acquérir des signaux de voltages provenant des senseurs,
- réaliser un traitement des signaux acquis, et
- générer des commandes de régulation afin de maintenir la température proche de la consigne.

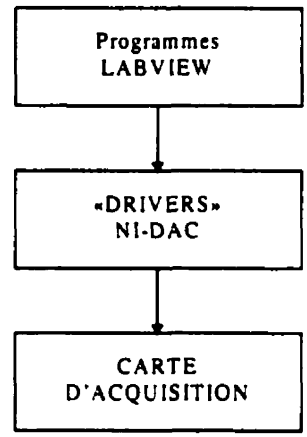
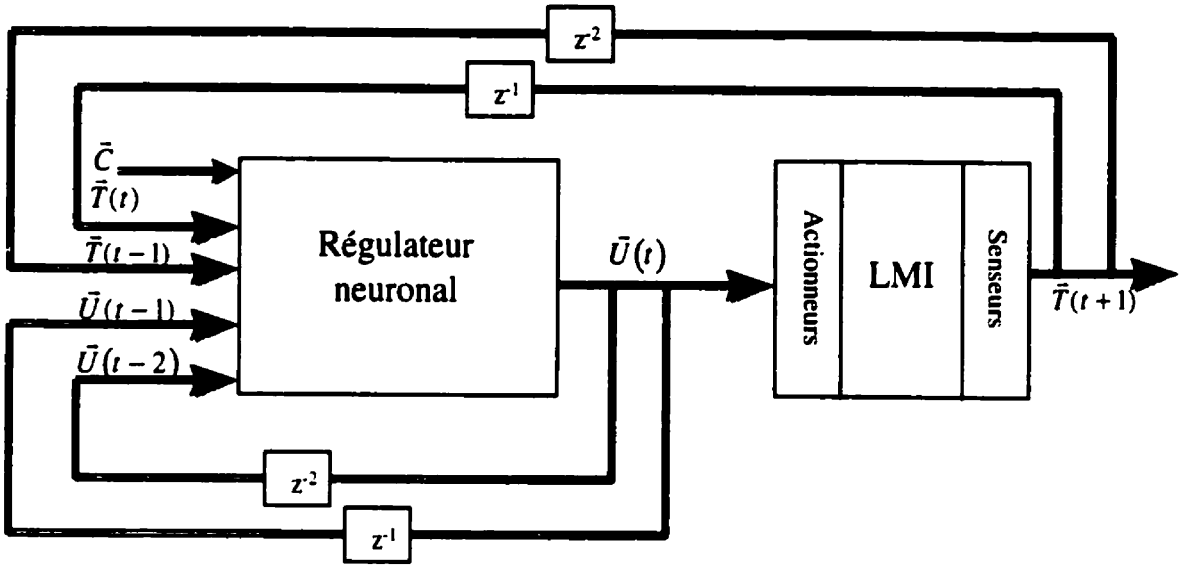
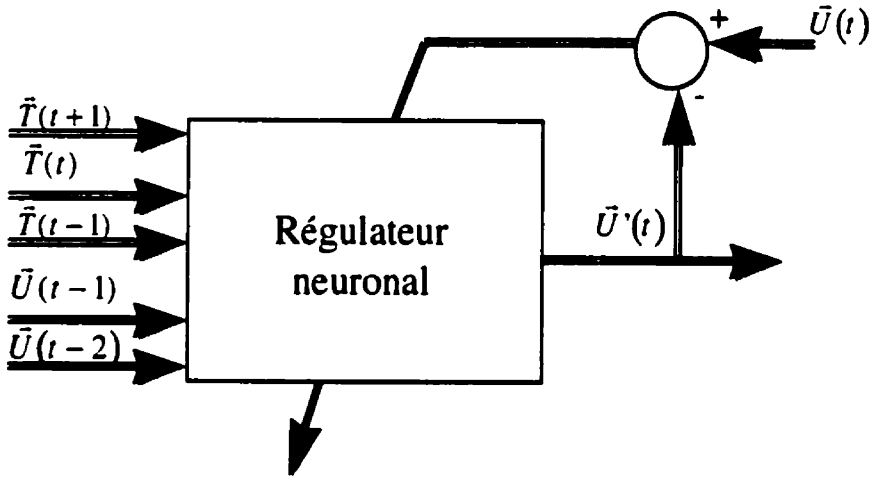


Figure 5.3. Liens entre LabVIEW, NI-DAQ et la carte d'acquisition

Le système de neurocommande adaptative pour le contrôle de la température dans le LMI est illustré à la Figure 5.4. Il est constitué principalement du procédé et du régulateur neuronal. À chaque itération de commande, les senseurs fournissent le vecteur de température $\vec{T}(t)$. Ensuite, le vecteur de commande $\vec{U}(t)$ est calculé par le régulateur en fonction du vecteur consigne $\vec{C}$, et des vecteur $\vec{T}(t), \vec{T}(t-1), \vec{U}(t-1)$ et $\vec{U}(t-2)$. L'adaptation du régulateur neuronal est assurée par l'entraînement du réseau perceptron clusterisé qui compose le régulateur, en utilisant chaque nouvel exemplaire d'Entrées/Sorties disponible au fur et à mesure que le procédé est commandé (tel que montré à la figure 5.4b).



(a)



(b)

Figure 5.4. (a) Le système de neurocommande pour le contrôle de la température, (b) schéma de neurocommande adaptative

$$\vec{U} = [U_{HP}, U_L, U_P, U_C]^T$$

$$\vec{T} = [T_{HP}, T_L, T_P, T_C]^T$$

$\vec{C}$: Vecteur consigne

U_{HP} : Commande au niveau de la section *Haute Précision*

U_L : Commande au niveau de la section *Laboratoire*

U_P : Commande au niveau de la section *Porte d'entrée*

U_C : Commande du système de chauffage

T_{HP} : température au niveau de la section *Haute Précision*

T_L : température au niveau de la section *Laboratoire*

T_P : température au niveau de la section *Porte d'entrée*

T_C : température au *Centre du Laboratoire*

5.3.1. Les senseurs

Un senseur est un capteur qui permet de mesurer des paramètres (température, pression, humidité, etc.) qui sont utilisés par le régulateur pour générer des commandes. Dans notre étude, les senseurs utilisés sont des thermostats qui ont des sorties en tension analogique variant entre 2.60 et 2.86 Volts CC. Les thermostats sont au nombre de quatre. Ils ont pour fonction de mesurer la température au centre du laboratoire et dans les trois sections suivantes (voir Figure 5.1):

1) section *Haute Précision*,
2) section *Laboratoire*, et
3) section *Porte d'entrée*.

Chacun de ces trois derniers senseurs est situé au milieu de la section correspondante à 20 cm du plafond. La *section CMM* n'a pas été considérée car elle est utilisée pour d'autres fins de recherche qui empêchent de contrôler la température dans cette section. Le thermostat utilisé au centre du laboratoire se trouve à coté du RTD à la même distance du plafond que les autres.

5.3.2. Les actionneurs

L'actionneur est l'appareil qui reçoit les commandes du régulateur et qui agit sur le procédé. Pour le processus de contrôle de la température dans le LMI, c'est des câbles chauffants qui agissent comme actionneurs. Ils agissent d'après les indications du régulateur neuronal pour augmenter la température dans les trois sections citées ci-dessus. Chacun des câbles chauffants est placé au plafond (au dessus des tuiles) de la section à contrôlée. La Figure 5.5 présente le pourcentage d'utilisation de la puissance de chacun des câbles chauffants en fonction du voltage en entrée.

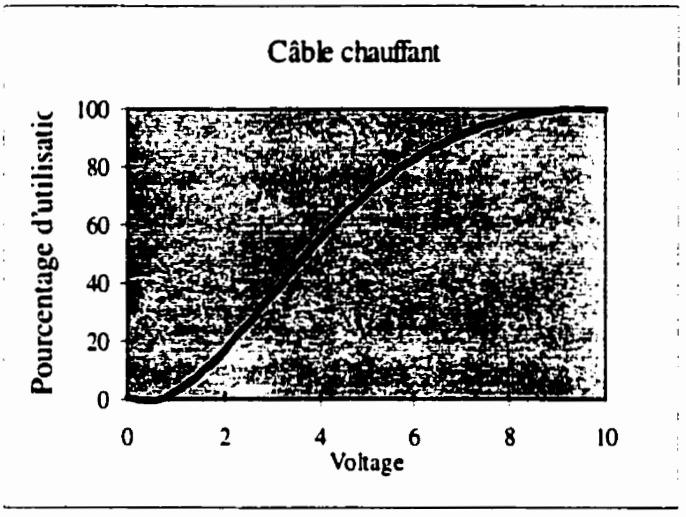


Figure 5.5. Pourcentage d'utilisation de la puissance des câbles chauffants en fonction du voltage d'entrée

5.3.3. L'organe de commande

L'équipement utilisé comme organe de commande est un calculateur numérique de type PC. En général, les industries n'emploient pas beaucoup de calculateurs numériques pour régulariser leurs procédés. La plupart des entreprises emploient des boucles de régulation analogiques incorporées dans des contrôleurs industriels, comme par exemple le Moore 352. Ces appareils sont équipés de régulateur à actions Proportionnelle Intégrale Dérivée (analogique), qui

sont en général au nombre de deux. Donc, la plupart des appareils ne peuvent pas gérer plus de deux boucles de régulation et le choix du régulateur se limite aux actions Proportionnelle, Proportionnelle Intégrale, Proportionnelle Dérivée, et au PID qui est une combinaison des trois actions. L'action Proportionnelle peut être représentée du point de vue électronique (analogique) par un amplificateur possédant un certain gain k_p. Les actions Intégrale et Dérivée sont aussi fabriquées avec des composantes électroniques telles que amplificateurs opérationnels, condensateurs et résistances disposées de manières différentes pour obtenir l'effet désiré. En outre, l'utilisation des stratégies de commande de type neuronal dans un contrôleur industriel n'est pas fréquente.

Si le régulateur PID analogique est un circuit électrique contenant des amplificateurs opérationnels, condensateurs et résistances incorporés parmi d'autres circuits formant le contrôleur industriel, le régulateur neuronal, lui, équivaut à quelques lignes de programmation faisant partie d'un programme effectuant la commande d'un processus. Bref, un régulateur neuronal est un programme reliant la sortie mesurée du procédé, à une commande calculée par ce dernier pour obtenir la valeur désirée (la consigne). Pour changer le régulateur, il suffit de changer le programme de régulation, c'est-à-dire de modifier, au pire, quelques lignes du programme de commande. Dans le cas d'un régulateur analogique, il faut modifier le circuit électronique si l'on veut changer de régulateur.

5.3.4. L'étalonnage

Comme nous venons de le voir, les signaux de sortie des senseurs sont des tensions analogiques variant entre 2.60 et 2.86 Volts. Donc, pour une valeur de la température au niveau du centre du laboratoire ou de l'une des trois sections, nous obtiendrions une valeur de tension numérique que nous devrons convertir en température (en degré C). En effet, il faut obtenir une relation entre la tension numérique et la température correspondante. Pour obtenir cette relation, il faut étalonner (calibrer) les quatre senseurs. L'étalonnage des senseurs a été fait par rapport à une sonde de référence. La forme générale de l'équation utilisée pour la conversion est donnée par:

$$T = a_1(\log V)^5 + a_2(\log V)^4 + a_3(\log V)^3 + a_4(\log V)^2 + a_5(\log V) + a_6 \qquad (5\text{-}2)$$

où,

T : est la température, et

V : est le voltage.

Les valeurs des coefficients a_i se trouvent dans le tableau ci-dessous (Tableau 5.2).

Tableau 5.2. Les valeurs des coefficients a_i de l'équation (5-2) pour les quatre senseurs

	a_1	a_2	a_3	a_4	a_5	a_6
Thermostat 1	1061883	-5202448	10193639	-9985088	488958	-957547
Thermostat 2	510099	-2513713	4953925	-4880537	2403599	-473354
Thermostat 3	909700	-4464675	8763238	-8598709	4217850	-827384
Thermostat 4	1165310	-5734551	11286271	-11104661	5462107	-1074457

5.4. RÉSULTATS EXPÉRIMENTAUX

5.4.1. Entraînement du réseau neuronique

Les données d'entraînement d'un réseau de neurones consiste en une liste d'entrées/sorties du phénomène à modéliser. Plus spécifiquement, dans le cas du modèle à réaliser dans le cadre de cette étude, la liste est formée par les vecteurs d'entrées $\vec{C}, \vec{T}(t), \vec{T}(t-1), \vec{U}(t-1)$ et $\vec{U}(t-2)$, et les vecteurs de sorties $\vec{U}(t)$ correspondants. Le but de l'entraînement, est que le réseau de neurones soit capable ensuite de fournir le vecteur de sortie $\vec{U}(t)$ lorsqu'on lui présente les vecteurs d'entrées $\vec{C}, \vec{T}(t), \vec{T}(t-1), \vec{U}(t-1)$ et $\vec{U}(t-2)$, et ceci non seulement pour ces valeurs apprises mais aussi pour d'autres valeurs non apprises. C'est le principal enjeu de la technique des réseaux neuroniques, qui ont justement cette capacité de modélisation. Le Tableau 5.3 contient les données utilisées pour l'apprentissage du réseau neuronique.

Tableau 5.3. Données d'entraînement a priori du réseau perceptron clusterisé (les températures sont en °C)

ENTRÉES																			SORTIEES		
$T_{HP(t+1)}$	$T_{L(t+1)}$	$T_{P(t+1)}$	$T_{C(t+1)}$	$T_{HP(t)}$	$T_{L(t)}$	$T_{P(t)}$	$T_{C(t)}$	$T_{HP(t-1)}$	$T_{L(t-1)}$	$T_{P(t-1)}$	$T_{C(t-1)}$	$U_{HP(t-1)}$	$U_{L(t-1)}$	$U_{P(t-1)}$	$U_{C(t-1)}$	$U_{HP(t-2)}$	$U_{L(t-2)}$	$U_{P(t-2)}$	$U_{C(t-2)}$	$U_{HP(t)}$	$U_{L(t)}$	$U_{P(t)}$
19.71	20.28	20.51	19.85	20.33	20.75	20.54	19.98	20.68	20.98	20.55	20.15	95	55	0	105	95	90	0	110	70	40	0
19.51	19.86	20.38	19.83	19.71	20.28	20.51	19.85	20.33	20.75	20.54	19.98	70	40	0	100	95	55	0	105	70	40	0
20.00	19.89	20.46	19.98	19.73	19.77	20.48	19.99	19.51	19.86	20.38	19.83	115	40	0	105	70	40	0	100	105	68	0
20.11	19.97	20.48	20.02	20.00	19.89	20.46	19.98	19.73	19.77	20.48	19.99	105	68	0	105	115	40	0	105	115	68	0
20.19	20.12	20.47	20.01	20.11	19.97	20.48	20.02	20.00	19.89	20.46	19.98	115	68	0	103	105	68	0	105	102	68	0
20.09	20.07	20.48	19.94	20.19	20.12	20.47	20.01	20.11	19.97	20.48	20.02	102	68	0	103	115	68	0	103	100	68	0
19.97	20.06	20.45	19.91	20.00	20.12	20.50	20.00	20.09	20.07	20.48	19.94	95	65	0	103	100	68	0	103	103	65	0
20.10	19.99	20.56	20.12	20.11	20.02	20.43	19.95	20.05	20.10	20.37	19.85	103	62	0	103	103	62	0	103	103	62	0
20.15	19.96	20.54	20.16	20.10	19.99	20.56	20.12	20.11	20.02	20.43	19.95	103	62	0	105	103	62	0	103	101	60	0
20.04	19.96	20.54	20.06	20.04	20.05	20.46	20.04	20.15	19.96	20.54	20.16	101	62	0	102	101	60	0	105	100	62	0
20.02	19.93	20.45	20.03	20.04	19.96	20.49	20.02	19.96	19.87	20.52	20.05	104	68	0	100	104	65	0	100	104	68	0
19.78	19.96	20.54	19.94	20.31	20.36	20.67	20.08	20.93	20.69	20.84	20.41	100	60	0	99	105	65	0	102	90	50	0
19.63	19.72	20.44	19.85	19.78	19.96	20.54	19.94	20.31	20.36	20.67	20.08	90	50	0	90	100	60	0	99	90	50	0
20.18	19.94	20.56	20.09	20.10	19.37	20.41	20.04	19.81	19.71	20.43	19.91	110	65	0	104	110	60	0	100	110	65	0
19.95	19.99	20.51	20.02	20.08	19.99	20.52	20.16	20.18	19.94	20.56	20.09	102	68	0	102	110	65	0	102	95	68	0
19.74	19.94	20.42	19.89	19.95	19.99	20.51	20.02	20.08	19.99	20.52	20.16	95	68	0	100	102	68	0	102	95	68	0
19.92	19.90	20.43	20.00	19.73	19.85	20.34	19.87	19.74	19.94	20.42	19.89	100	68	0	95	95	68	0	95	106	68	0
20.13	20.11	20.53	20.11	20.02	20.01	20.50	20.02	19.92	19.9	20.43	20.00	106	68	0	105	106	68	0	103	106	68	0
20.17	20.17	20.54	19.96	20.13	20.11	20.53	20.11	20.02	20.01	20.5	20.02	106	68	0	105	106	68	0	105	106	68	0
20.03	20.11	20.52	19.91	20.17	20.17	20.54	19.96	20.13	20.11	20.53	20.11	106	68	0	105	106	68	0	105	100	64	0
19.95	20.04	20.46	19.98	20.03	20.11	20.52	19.91	20.17	20.17	20.54	19.96	100	64	0	102	106	68	0	105	95	60	0
19.97	20.01	20.43	19.90	20.11	20.11	20.42	19.87	20.30	20.21	20.44	19.97	100	60	0	102	105	65	0	102	95	55	0
19.87	19.86	20.41	19.89	19.97	20.01	20.43	19.9	20.11	20.11	20.42	19.87	95	55	0	102	100	60	0	102	90	50	0
19.95	19.78	20.51	20.15	19.87	19.86	20.41	19.89	19.97	20.01	20.43	19.9	90	50	0	102	95	55	0	102	90	50	0
19.85	19.82	20.53	20.01	19.89	19.82	20.54	20.16	19.95	19.78	20.51	20.15	95	55	0	105	90	50	0	104	95	60	0
19.87	19.87	20.52	20.07	19.85	19.82	20.53	20.01	19.89	19.82	20.54	20.16	95	60	0	102	95	55	0	105	100	65	0
19.93	19.87	20.45	19.92	19.87	19.87	20.52	20.07	19.85	19.82	20.53	20.01	100	65	0	100	95	60	0	102	105	68	0

La période d'échantillonnage (saisie des températures au niveau des quatre senseurs) et de commande, est de 2 minutes. L'architecture du réseau perceptron clusterisé utilisé est 20 x 10 x 10 x 4. Les différentes valeurs des paramètres d'entraînement sont données au Tableau 5.4. La somme des erreurs au carré entre la sortie du réseau et la valeur désirée obtenue avec les données d'apprentissage est égale à 0.1098. Le réseau perceptron clusterisé est adapté récursivement. Dix itérations d'apprentissage sont effectuées sur le nouvel exemplaire. Les valeurs des paramètres d'adaptation sont les mêmes que celles utilisées dans l'apprentissage à priori (Tableau 5.4).

Tableau 5.4. Les valeurs des paramètres d'entraînement à priori et en continu du réseau perceptron clusterisé

Paramètre d'entraînement	
Tolérance	0.101
C_1	1
C_2	0.0005
C_3	1
α	0.9
η	0.9
η_1	0.9

5.4.2. Comparaison avec le régulateur à base de PID

Pour mettre à l'épreuve le régulateur neuronal face à des variations qui peuvent survenir lors d'une véritable opération de contrôle, des perturbations ont été volontairement ajoutées lors du contrôle de température dans le laboratoire LMI.

Les résultats des essais de perturbation étudiés avec les deux systèmes de commande (PID et régulateur neuronal) sont illustrés dans les tableaux Tableau 5.5 et Tableau 5.6. Dans ces derniers, E_M représente l'écart maximal entre les valeurs de la température dans les quatre zones (*Haute Précision*, *Laboratoire*, *Porte d'entrée* et *Centre du laboratoire*). Les différents essais de

perturbation sont présentés aux deux systèmes de commande dans les mêmes conditions et en respectant l'ordre de présentation. Dans le cas où la lampe Helium et l'ordinateur sont allumés, nous avons simulé la présence d'une personne au niveau de la section *Haute Précision* en utilisant un séchoir qui dégage la même chaleur que celle dégagée par un corps humain.

Tableau 5.5. Les résultats des essais de perturbation avec le système de commande à base de PID (les températures sont en °C)

	T_{HP}	T_L	T_P	T_C	E_M
Pas de perturbation	19.34	19.77	20.79	20.26	1.45
Helium et ordinateur allumés	19.74	19.73	20.69	20.15	0.96
Helium, ordinateur et CMM allumés	19.74	19.79	20.51	20.10	0.77
Porte ouverte et CMM allumée	19.32	19.74	20.44	20.23	1.12
Porte ouverte	19.30	19.76	20.55	20.20	1.25
$\sum(T-20)^2$	1.5232	0.2951	1.8564	0.1930	

L'étude des variations de la température au sein du laboratoire LMI nous a permit de ressortir les constatations suivantes:

1) Lorsque la température au *Centre du laboratoire* est de 20°C (20.01), la température:

- au niveau de la section *Haute Précision* : 19.04°C
- au niveau de la section *Laboratoire* : 19.58°C
- au niveau de la section *Porte d'entrée* : 20.54°C

2) Pour avoir une température de 20°C au niveau de la section *Porte d'entrée*, la température au *Centre du laboratoire* doit avoisiner 19.4°C (19.31)

3) Pour une même valeur ampérique (voltage) de la commande du système de chauffage, la température peut varier de 0.1 à 0.15°C

Tableau 5.6. Les résultats des essais de perturbation avec le régulateur neuronal (les températures sont en °C)

	T_{HP}	T_L	T_P	T_C	E_M
Pas de perturbation	20.05	19.82	20.28	19.73	0.55
Helium et ordinateur allumés	20.17	19.91	20.19	19.76	0.43
Helium, ordinateur et CMM allumés	20.15	19.92	20.24	19.73	0.51
Porte ouverte et CMM allumée	19.87	19.81	20.18	19.85	0.37
Porte ouverte	19.90	19.80	20.18	19.80	0.38
$\sum(T-20)^2$	0.0808	0.1230	0.2369	0.2659	

Les tableaux ci-dessus (Tableau 5.5 et Tableau 5.6), montrent que le système de commande utilisant le régulateur neuronal a réussi à compenser les perturbations mieux que le système de commande à base de PID. La température est demeuré au voisinage de 20°C à 0.20 près durant la plupart des perturbations. Le régulateur neuronal s'adapte très bien, et assez rapidement aux nouvelles conditions et les températures sont amenées proche de la consigne.

En effet, l'utilisation du régulateur neuronal pour ajuster la température dans les trois sections du laboratoire LMI, apporte une grande amélioration à la performance du système de commande. L'amélioration de la performance que nous venons de noter résulte en réalité de deux excellentes propriétés du régulateur neuronal à base de réseau perceptron clusterisé. Ce sont les capacités d'adaptation en continu de ce dernier et la stabilité des systèmes de neurocommande. La stabilité est une propriété très importante que tout système de commande doit posséder.

Le facteur déterminant pour la stabilité d'un système de commande comme le régulateur neuronal, est la stabilité même de l'adaptation du régulateur neuronal. En effet, un tel système est constitué d'une boucle d'adaptation qui modifie les paramètres du réseau neuronique en minimisant l'erreur SE (Figure 5.4b).

La température élevée au niveau de la section Porte d'entrée, empêche le régulateur neuronal de maintenir la température proche de 20°C au niveau du centre du laboratoire.

Toutefois, les résultats seraient plus pertinents en absence des instalations au-dessus de la CMM qui empêchent l'air en provenance du système de chauffage de circuler au niveau de la section Porte d'entrée.

5.5. CONCLUSION

Le régulateur neuronal développé dans le cadre de cette étude, possède d'excellentes capacités d'adaptation en continu. Celles-ci sont assurées par les capacités d'adaptation du réseau perceptron clusterisé.

Le comportement adaptatif du réseau perceptron clusterisé a été prouvé à travers les résultats présentés dans ce chapitre et dans le chapitre précédent. Les problèmes d'instabilité qui sont généralement associés aux approches traditionnelles de commande adaptative ne sont pas rencontrés avec le système de neurocommande à base de réseau perceptron clusterisé.

CHAPITRE VI

CONCLUSION GÉNÉRALE

Ce mémoire avait pour objectif l'amélioration de l'adaptation en continu du réseau perceptron multicouche et d'utiliser l'approche neuronique pour le contrôle de la température dans le laboratoire de métrologie industrielle (LMI) du département de génie mécanique de l'Université Laval.

L'étude de l'adaptation du réseau perceptron multicouche avec la méthode d'apprentissage traditionnelle de rétropropagation a montré qu'il y a une dégradation des informations stockées dans le réseau au fur et à mesure que celui-ci est adapté. Ce problème constitue un handicap majeur pour le développement de stratégies efficaces de neurocommande adaptative. Nous y avons consacré beaucoup de temps pendant lequel nous avons exploré la littérature sur l'organisation, l'apprentissage et la mémorisation dans le cerveau biologique. La révélation la plus importante de cette dernière revue est l'existence dans le réseau neuronique biologique d'une organisation stupéfiante du traitement de l'information qui pourrait être à l'origine des phénomènes d'apprentissage et de la mémorisation. Il est difficile d'obtenir une telle organisation dans le réseau perceptron multicouche en utilisant l'algorithme d'apprentissage traditionnel de rétropropagation.

Par conséq
développement d'u
Le nouvel algorith
d'auto-organisation
l'information (la c
spatiale et par les d
dans l'architecture

Même si le
du réseau perceptro
perceptron multico
perceptron clusteris

- Contraireme
  montré une

- Le réseau
  comparaison
  perceptron c
  présentés, r
  intelligente
  peu affectée

Lors d'une
température dans l
améliorations que l
laquelle le réseau p
adaptative était supe
capacités d'adaptat
capacités et proprié

d'adaptation du régulateur neuronal lui est donnée par le réseau perceptron clusterisé et son algorithme d'apprentissage en continu.

Donc l'application des réseaux neuroniques, quoique récente, est vouée à un avenir très prometteur, soit pour modéliser et pour commander des systèmes complexes. À cause de la limitation des techniques de commande traditionnelles, la plupart des systèmes de commande et de supervision se sont tournés vers l'intelligence artificielle, plus précisément vers les systèmes experts. Mais grâce à leur capacité de modélisation et d'adaptation, les réseaux neuronaux vont, dans un proche avenir, possiblement déclasser les systèmes experts, vue leur analogie envers le cerveau biologique humain.

Comme perspectives de ce travail, nous suggérons les extensions suivantes:

- développer l'architecture d'un grand réseau neuronique à base de plusieurs réseaux perceptrons clusterisés. Ce grand réseau constituera une mémoire gigantesque pour le stockage des données.
- augmenter la vitesse et la précision dans l'approche de neurocommande adaptative, et appliquer cette dernière à d'autres procédés comme celui de contrôle de l'humidité dans le LMI.

RÉFÉRENCES

[1] McCulloch W. S. Et Pitts W. 1943: A logical calculus of the ideas imminent in nervous activity, Bulletin of Mathematical Biophysics, Vol. 5, 1943, pp. 115-133.

[2] Hebb D. O. 1949: The organisation of behavior, John Wiley &Sons, New-york,1949.

[3] Rosenblatt R. 1959: Principles of neurodynamics, Spartan Books, New-York, 1959.

[4] Rosenblatt F.1958 : The perceptron: a probalistic model for information storage and organisation in the brain, Psych. Rev., Vol. 65 1958, pp. 386-408.

[5] Minsky M. et Papert S. 1969: Perceptrons: an introduction to computational geometry, MIT Press, 1969

[6] Rumelhart D. E. et al. 1988: Learning internal representations by error propagation, Parallel Distributed Processing: Explorations in the Microstructures of Cognition, Vol. 1: Fondations, Rumelhart D. E. et McClelland J. L. (Eds.), Cambridge, MA:MIT Press, 1988, pp. 318-362.

[7] Hornik K. et al. 1988: Multi-layer feed-forward networks are universal approximatiors, Discussion paper, Departement of Economics, University of California, San Diego, La jolla, CA, 1988.

[8] Rumelhart R. et al. 1986: Learning internal representations by error propagation, Parallel Distributed Processing: Explorations in the Micostructure of Cognition, Vol. 1, 1986, pp. 318-364.

[9] McClelland L. L. et Rumelhart D. E. 1988: Explorations in parallel distributed processing, mit Press, 1988.

[10] Haykin S. 1994: Neural network, a comprehensive foundation, Macmillan College Publishing Company Inc., 1994.

[11] Cooper L. et Steinberg D. 1970: Introduction to methods of optimisation, W. B. Saunders Company, 1970.

[12] Gill P. et al. 1981: Practical optimization, New-York: Academic, 1981.

[13] Adby P. R. et Dempster M. A. H. 1974: Introduction to optimization methods, Chapman and Hall, London, 1974.

[14] William H. et al. 1988: Numerical recipes in C the art of scientific computing, Cambridge University Press, 1988.

[15] Azouzi R. et Guillot M. 1996: On-line learning and optimization of manufacturing processes using a novel hybrid neural network applied to machining, Submitted for review in IEEE Transactions on Systems, Man and Cybernetics, 1996.

[16] Kohonen T. 1982: Self-organizing formation of topology correct feature map, Biol. Cybern., Vol. 43, 1982, pp. 59-69.

[17] Kohonen T. 1990: The self-organizing map, Proceedings of the IEEE, Vol. 78, No. 9, September 1990, pp. 1464-1480.

[18] Chryssolouris G. et Guillot M. 1990: A comparison of statisticaland AI approaches to selection of parameters in intelligent machining, Transactions of the ASME, Vol. 112, 1990.

[19] Fu L. and Rilett. L. R. 1995: Dynamic O-D travel time estimation using an artificial neural network. Vehicle Navigation and Information Systems Conference (VNIS) Proceedings of the 6th 1995 Vehicle Navigation and Information Systems Conference Jul 30-Aug 2 1995 1995 Seattle, WA, XXXUSAXXX IEEE Piscataway NJ USA p 236-242.

[20] Hartov A. et al. 1994: CONTROL Problem in hyperthermia; American Society of Mechanical Engineers, Heat Transfer Division, (Publication) HTD Advances in Heat and Mass Transfer in Biological Systems Proceedings of the 1994 International Mechanical Engineering Congress and Exposition Nov 6-11 1994 v 288 1994 Chicago, IL, USA Sponsored by: ASME ASME New York NY USA p 119-126.

[21] Zaldivar J.M. et al. 1992: Control of batch reactors using neural networks; Chemical Engineering and Processing v 31 n 3 Jul 1992 p 173-180.

[22] Eki Y. et al. 1997: New method of off-line adjustement of feed-forward control for fossil-fired plant main control system; Electrical Engineering in Japan (English translation of Denki Gakkai Ronbunshi) v 119 n 1 Apr 15 1997 Scripta Technica Inc New York NY USA p 55-61.

[23] Shmueli D. and al. 1996: Neural network analysis of travel behavior: evaluating tools for prediction. Transportation Research Part C: Emerging Technologies v 4 n 3 Jun 1996 Elsevier Science Inc. Tarrytown NY USA p 151-166.

[24] Drouin A. 1992: Application des réseaux neuronaux pour la commande numérique d'un système expérimental liquide-niveau, Mémoire présenté pour l'obtention du grade de maîtrise ès sciences appliquées, Université Laval.

[25] Thibault J. et Grandjean B. P. A. 1991: Neural networks in process control - a survey, IFAC Inter. Symp. ADCHEM-91, Toulouse, France, 14-16 octobre 1991, pp. 295-304.

[26] Nguyen D. H. et Widrow B. 1990: Neural networks for self-learning control systems, IEEE Control Systems Magazine, 10, 3, pp. 18-23.

[27] Rigal R. 1987 : Motricité humaine, fondement et applications pédagogiques. Tome1, Presses de l'Université du Québec.

[28] Durbin R. 1989: On the correspondance between network models and the nervous system, The Computing Neuron, Eds. Durbin R., Miall C. et Mitchison G., Addison-Wesley Publication Co, 1989.

[29] Dubois O. et al. 1994: Adaptive neural network control of the temperature in an oven, IEE Colloquium (Digest) Proceedings of the IEE Colloquium on Advances in Neural Networks for Control and Systems May 25-27 1994 n 136 May 25-27 1994 London, UK IEE Stevenage Engl p 8/1-8/3.

[30] Zurada J. M. 1992 : Introduction to artificial neural systems, West publishing company.

[31] Lippmann R. P. 1987: An introduction to computing withneural nets, IEEE ASSP Magazine , Avril 1987.

[32] Rangwala S. S. et Dornfeld D. A. 1989: Learning and optimization of machining using computing abilities of neural networks, IEEE Transactions on Systems, MAN, and Cybernetics, Vol. 19, No. 2, Mars/Avril 1989.

[33] Liu H. et al. 1989: Neural network architecture for robot hand control, IEEE Control systems Magazine, 1989.

[34] Tryba V. et Goser K. 1991: Self-organizing feature maps for process control in chemistry, Artificial Neural Networks, Kohonen T., Makisara K., Simula O. et Kangas J. (Eds), Elsevier Science Publishers B. V. (North-Holland), 1991.

ANNEXE

CALCULS DES DÉRIVÉES PARTIELLES DE L'ERREUR PAR RAPPORT AUX PARAMÈTRES DU RÉSEAU PERCEPTRON STANDARD ET DU RÉSEAU PERCEPTRON CLUSTERISÉ

L'entraînement d'un réseau neuronique repose sur la minimisation de l'erreur SE_e en ajustant les paramètres d'apprentissage $w_{i,j,c}$ *et* $\theta_{j,c}$. Ceci nécessite le calcul des dérivées partielles de SE_e par rapport aux paramètres d'apprentissage. Nous allons exposer ces calculs pour un réseau perceptron standard et pour un réseau perceptron clusterisé. Rappelons d'abord que SE_e et $I_{j,c}$ sont données par:

$$SE_e = \sum_{i=1}^{n_{cc}} (O_{i,cc} - D_{i,e})^2 \qquad \text{(A-1)}$$

$$I_{j,c} = \sum_{i=1}^{r_{c-1}} w_{i,j,c} O_{i,c-1} + \theta_{j,c} \qquad \text{(A-2)}$$

A.1. RÉSEAU PERCEPTRON STANDARD

Tout d'abord, nous déterminons les dérivées partielles suivantes:

$$\partial I_{j,c} / \partial w_{i,j,c} = O_{i,c-1} \qquad \text{(A-3)}$$

$$\partial I_{j,c} / \partial \theta_{j,c} = 1 \qquad \text{(A-4)}$$

$$\partial O_{j,c} / \partial I_{j,c} = O_{j,c}\left(1 - O_{j,c}\right) \qquad \text{(A-5)}$$

On note que toutes les dérivées partielles ci-dessus peuvent être calculées localement au niveau de chaque neurone durant l'opération normale du réseau. Posons:

$$\partial SE_e / \partial I_{j,c} = -\delta_{j,c} \quad et \quad \partial SE_e / \partial O_{j,c} = \xi_{j,c} \qquad \text{(A-6,7)}$$

alors les dérivées partielles de SE_e par rapport aux paramètres d'apprentissage peuvent être exprimées comme suit:

$$\partial SE_e / \partial w_{i,j,c} = \left(\partial SE_e / \partial I_{j,c}\right)\left(\partial I_{j,c} / \partial w_{i,j,c}\right) = -\delta_{j,c} O_{i,c-1} \qquad \text{(A-8)}$$

et

$$\partial SE_e / \partial \theta_{j,c} = \left(\partial SE_e / \partial I_{j,c}\right)\left(\partial I_{j,c} / \partial \theta_{j,c}\right) = -\delta_{j,c} \qquad \text{(A-9)}$$

Les quantité δ et ξ sont calculées dans (A-8 et 9) en retropropageant l'erreur observée à la sortie, à travers tout le réseau. Considérons la couche de sortie du réseau. Alors:

$$\delta_{j,cc} = -\partial SE_e / \partial I_{j,cc} = -\left(\partial SE_e / \partial O_{j,cc}\right)\left(\partial O_{j,cc} / \partial I_{j,cc}\right) \qquad \text{(A-10)}$$

En utilisant (A-1,2,3,4 et 5), cette dernière devient:

$$\delta_{j,cc} = O_{j,cc}\left(1 - O_{j,cc}\right)\left(D_{j,e} - O_{j,cc}\right) \qquad \text{(A-11)}$$

Les quantités $\xi_{i,cc}$ sont calculées comme suit:

$$\xi_{j,cc} = \partial SE_e / \partial O_{j,cc} \qquad \text{(A-12)}$$

En utilisant (A-1, 3, 4 et 5), on a:

$$\xi_{j,cc} = O_{j,cc} - D_{j,e} \qquad \text{(A-13)}$$

Pour les autres couches:

$$\xi_{i,c} = \partial SE_e / \partial O_{i,c} = \sum_{j=1}^{n_{c+1}} \left[\left(\partial SE_e / \partial I_{j,c+1} \right) \left(\partial I_{j,c+1} / \partial O_{i,c} \right) \right] \qquad \text{(A-14)}$$

En utilisant (A-1 et 7), cette équation peut être simplifiée comme suit:

$$\xi_{i,c} = -\sum_{j=1}^{n_{c+1}} \left[\delta_{j,c+1} w_{i,j,c+1} \right] \qquad \text{(A-15)}$$

Les quantités $\delta_{i,c}$ sont calculées comme suit:

$$\delta_{i,c} = -\left(O_{i,c} \left(1 - O_{i,c} \right) \xi_{i,c} \right) \qquad \text{(A-16)}$$

A noter que les $\xi_{i,c}$ dépendent uniquement des δ se trouvant dans la couche (c+1). Ainsi, dans une même couche, les ξ peuvent être calculées en parallèle. En conclusion, les dérivées partielles de SE_e par rapport aux paramètres d'entraînement sont données par:

$$\partial SE_e / \partial w_{i,j,c} = -\delta_{j,c} O_{i,c-1} \qquad \text{(A-17)}$$

$$\partial SE_e / \partial \theta_{j,c} = -\delta_{j,c} \qquad \text{(A-18)}$$

Le développement présenté dans cette section s'applique pour un exemplaire e donné. Pour la méthode standard d'entraînement par rétropropagation, les quantités δ sont directement

utilisées pour adapter les paramètres d'apprentissage à la suite de la présentation de l'exemplaire e.

A.2. RÉSEAU PERCEPTRON CLUSTERISÉ

Dans le réseau perceptron clusterisé, le calcul des dérivées partielles de SE_e par rapport aux paramètres d'apprentissage s'effectue d'une manière analogue à celle présentée ci-dessus. Cependant, certaines modifications doivent être apportées. Rappelons d'abord que dans ce réseau, l'entrée à chaque neurone, est donnée par:

$$I_j = \varphi_j \sum_i W_{ij} X_i C_{ij} + \theta_{j,c} \qquad \text{(A-19)}$$

avec,

$$C_{ij} = e^{-C_1 d_{ij}^2} \qquad \text{(4-20)}$$

$$\varphi_j = e^{-C_2 D_j^2} \qquad \text{(4-21)}$$

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \qquad \text{(4-22)}$$

$$D_j = \|\bar{o} - \bar{v}\| = \sqrt{\sum_{p=0}^{k} \left(o_p - v_{j,p}\right)^2} \qquad \text{(4-23)}$$

Par conséquent, tous les calculs impliquant I_j sont modifiés. Cela affecte les équations (A-3,8 et 15). Tous les autres calculs sont effectués tel que montré à la section précédente.

D'après (A-19), la dérivée partielle de I_j par rapport aux paramètres d'apprentissage est donnée par:

$$\partial I_{j,c} / \partial w_{i,j,c} = \varphi_{j,c} O_{i,c-1} C_{i,j,c} \qquad \text{(A-24)}$$

$$\partial I_{j,c} / \partial \theta_{j,c} = 1 \qquad \text{(A-25)}$$

Nous pouvons maintenant re-formuler les expression des quantités $\delta_{j,c}$ pour un réseau perceptron clusterisé, qui deviennent alors:

Pour la couche de sortie:

$$\delta_{j,cc} = O_{j,cc}(1 - O_{j,cc})(D_{j,e} - O_{j,cc}) \qquad \text{(A-26)}$$

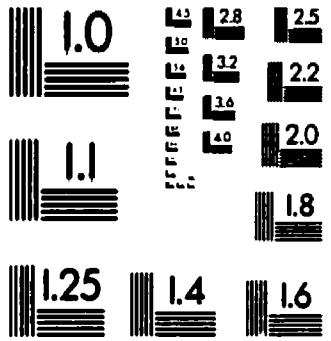
Pour les autres couches:

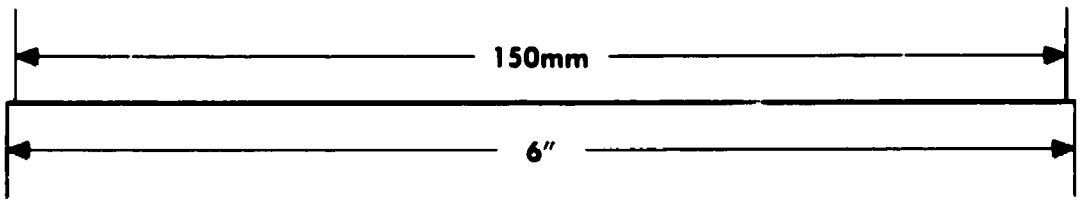
$$\xi_{j,c} = -\sum_{k} \delta_{k,c+1} \varphi_{k,c+1} \left[W_{k,j,c+1} C_{k,j,c+1} + \left[C_2^2 (O_{j,c} - V_{k,j,c+1}) / \log \varphi_{k,c+1} \right] \sum_{i} W_{k,i,c+1} O_{i,c} C_{k,i,c+1} \right] \qquad \text{(A-27)}$$

Finalement, les dérivées partielles de SE_e par rapport aux paramètres d'entraînement sont données par:

$$\partial SE_e / \partial w_{i,j,c} = -\delta_{j,c} \varphi_{j,c} O_{i,c-1} C_{i,j,c} \qquad \text{(A-28)}$$

$$\partial SE_e / \partial \theta_{j,c} = -\delta_{j,c} \qquad \text{(A-29)}$$





APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989