UNIVERSITÉ DU QUÉBEC

MÉMOIRE

PRÉSENTÉ À

L'UNIVERSITÉ DU QUÉBEC À CHICOUTIMI

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INGÉNIERIE

PAR

MOULAY ABDERRAHIM ENNAJI

ANALYSE ET CONCEPTION D'UN RÉSEAU DE NEURONES FORMELS POUR LE FILTRAGE D'UN SIGNAL DYNAMIQUE

SEPTEMBRE 1992

Bibliothèque
Paul-Émile-Boulet

UQAC
Université du Québec
à Chicoutimi

Mise en garde/Advice

Afin de rendre accessible au plus grand nombre le résultat des travaux de recherche menés par ses étudiants gradués et dans l'esprit des règles qui régissent le dépôt et la diffusion des mémoires et thèses produits dans cette Institution, **l'Université du Québec à Chicoutimi (UQAC)** est fière de rendre accessible une version complète et gratuite de cette œuvre.

Motivated by a desire to make the results of its graduate students' research accessible to all, and in accordance with the rules governing the acceptation and diffusion of dissertations and theses in this Institution, the **Université du Québec à Chicoutimi (UQAC)** is proud to make a complete version of this work available at no cost to the reader.

L'auteur conserve néanmoins la propriété du droit d'auteur qui protège ce mémoire ou cette thèse. Ni le mémoire ou la thèse ni des extraits substantiels de ceux-ci ne peuvent être imprimés ou autrement reproduits sans son autorisation.

The author retains ownership of the copyright of this dissertation or thesis. Neither the dissertation or thesis, nor substantial extracts from it, may be printed or otherwise reproduced without the author's permission.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABRÉVIATIONS

— n: nombre de noeuds à l'entrée d'un réseau de neurones;

— p: nombre de points (échantillons) par période;

— T: période d'un signal;

— f: fréquence du signal définie comme étant l'inverse de la période;

— f_e: fréquence d'échantillonnage (16 kHz);

— f_c: fréquence de coupure;

— f_{cb}: fréquence de coupure basse;

— f_{ch}: fréquence de coupure haute;

— a: coefficient utilisé dans le calcul de la pente de la sigmoïde;

— a,b: coefficients utilisés dans le calcul de l'équation de normalisation linéaire;

— A.I: intelligence artificielle "Artificial Intelligence";

— R.N: réseau de neurones;

— R.P.: rétropropagation ;

— R.P.S.: rétropropagation standard;

— R.P.R ou F.R.P..: rétropropagation rapide;

— R.P.C.: rétropropagation cumulée;

— dB: unité de mesure, le décibel;

— m: nombre total d'exemples (vecteur de dimension n) à apprendre;

— w: nombre de poids dans un réseau;

— ϵ: incertitude désirée;

— d: sortie désirée dans un réseau de neurones;

— y: sortie obtenue dans un réseau de neurones;

— e: erreur obtenue entre la sortie désirée et la sortie obtenue;

— s: sommation des données pondérées par les poids du réseau à l'entrée d'un neurone artificiel;

— n_{in}: nombre de noeuds à l'entrée du réseau de neurones;

— n_{out}: nombre de noeuds à la sortie du réseau de neurones;

— n_{cc}: nombre de couches cachées du réseau de neurones;

— n_h: nombre de noeuds dans la ou les couches cachées du réseau de neurones;

— c.c.: couche cachée;

— TSLA: "Time Supervised Learning Algorithm";

— A: amplitude d'un signal;

— R: rétropropagation;

— x(t): signal temporel à l'entrée d'un réseau de neurones;

— h(t): fonction de transfert temporelle d'un réseau de neurones;

— g(t): signal temporel à la sortie d'un réseau de neurones;

— BDSONS3: base de données du français canadien, réalisée au CCRIT à l'été 90;

— LMS: "Least Mean Square";

— P.B.: filtre passe-bas;

— P.H.: filtre passe-haut;

— C.B: filtre coupe-bande (filtre à réjection);

— w_b: connexion synaptique du noeud 1 (biais à un autre noeud);

— wij: connexion synaptique du noeud i au noeud j;

— W1: matrice des connexions synaptiques entre les noeuds de la couche d'entrée et la couche cachée ;

— W2: matrice des connexions synaptiques entre les noeuds de la couche cachée et de la couche de sortie;

— k: nombre d'exemples (patrons ou séquences) à apprendre;

— f(x): fonction non-linéaire utilisée à la sortie du neurone.

— y_{0i}: sortie des noeuds de la couche d'entrée;

— y_{1i}: sortie des noeuds de la couche cachée;

— y_{2i}: sortie des noeuds de la couche de sortie;

REMERCIEMENTS

Je tiens à remercier toutes les personnes qui m'ont supporté dans l'élaboration de ce mémoire, en particulier M. Jean Rouat, professeur au département des sciences appliquées, directeur de recherche, qui a su m'orienter dans ma démarche de travail; je le félicite pour sa patience exemplaire et ses bons conseils; M. Raymond Descout, gestionnaire multi-média au Centre Canadien de Recherche sur l'Informatisation du Travail, pour avoir accepté de siéger au sein du comité de maîtrise en tant que membre extérieur, malgré le peu de temps libre que lui confèrent ses responsabilités au Ministère des communications; M. Yong Chun Liu, ingénieur en télécommunications, étudiant au programme de maîtrise en ressources et systèmes, pour son aide appréciable aux questions reliées au champ de la parole; M. André Falgueyret, ingénieur et gestionnaire du système informatique au département des sciences appliquées à l'UQAC, pour son soutien en informatique; M. Sylvain Faucher, ingénieur au Centre Canadien sur l'Informatisation du Travail, pour nous avoir permis d'utiliser la base de données BDSON3 alors que le produit final n'était pas disponible; M. Réjean Ouellet, chercheur et membre du groupe de recherche GRIPS, pour m'avoir permis l'utilisation de certains logiciels; M. Eric Chabot , étudiant au programme de maîtrise en ingénierie, pour les discussions fructueuses et bénéfiques; M$^{me.}$ Annie Tremblay, enseignante à l'atelier Le Cran , pour avoir accepté de lire et de corriger le manuscrit; M. Danny Ouellet, responsable du laboratoire de génie informatique, pour m'avoir facilité l'accès au matériel; tous les membres du groupe ERMETIS (Équipe de Recherche en Microélectronique et Traitement Informatique des Signaux) et enfin, toutes les personnes ressources que j'aurais oubliées de citer.

RÉSUMÉ

Ce travail fait une analyse approfondie des capacités d'un réseau de neurones à propagation-avant ("feed-forward") à filtrer des signaux. Ce réseau est entrainé, par l'algorithme de la rétropropagation rapide [82, 83], à mémoriser et à généraliser un signal dynamique, signal dont les paramètres varient dans le temps. Plusieurs architectures de réseaux de neurones sont étudiées pour faire ressortir les qualités spécifiques de ces modèles en ce qui a trait au filtrage d'un signal dynamique.

Le modèle proposé met en évidence la capacité d'un réseau à propagation-avant (feed-forward) à mémoriser un signal dynamique, à généraliser au niveau de l'amplitude et des fréquences, à regénérer un signal déjà appris de façon indéfinie (capacité apparente sur des architectures récurrentes), à reproduire n'importe quel signal avec un ajout de "bruit" sous forme de composantes hautes fréquences et basses fréquences, à filtrer un signal comme le fait un passe-bas, un passe-haut, un passe-bande et un coupe-bande (performances moyennes).

Ce travail met l'accent sur la possibilité de réduire la taille des données à l'apprentissage pour éventuellement diminuer le temps de convergence, sans pour autant affecter les performances du modèle. Il met en évidence l'optimisation du nombre de noeuds dans le réseau et, établit une relation entre le signal à apprendre et le nombre de noeuds dans les couches d'entrée et cachée (s).

Notre modèle est comparé à ceux de Chin'ichi Tamura et Alex Waibel [81], de Ryato Kamimura [40, 41] et de Chris Bishop [10]. Quelques applications de ces modèles sont citées dans la conclusion.

Mots clés : Feed-forward — Propagation-avant — Rétropropagation rapide —Filtrage — Traitement du signal dynamique — Filtres : passe-bas , passe–haut, passe bande et coupe-bande —.

INTRODUCTION

Le traitement d'un signal dynamique suscite beaucoup d'intérêt chez les chercheurs oeuvrant dans le champ des réseaux de neurones. Il existe une littérature abondante sur ce sujet, notamment sur la reconnaissance du langage naturel et des séquences pour le contrôle d'un processus. Cependant, l'architecture utilisée pour traiter ce genre de problème est constituée principalement d'une structure récurrente. Car, cette forme permet de caractériser l'état d'une cellule de la couche de sortie au temps t en se basant sur l'information qui a précédé aux temps (t-1) (t-2), etc... Cette caractérisation de l'état de la cellule en question permet éventuellement de déterminer son état futur. Parmi les algorithmes qui se sont les plus illustrés, on trouve les architectures de Jordan [38], de Paul Werbos [86] et de Ryato Kamimura [40, 41]. Ce type de réseau de neurones est souvent associé à un traitement dynamique, traitement dont les paramètres varient dans le temps. Chacune des structures susmentionnées est unique dans son genre et permet d'effectuer une tâche spécifique. Elles se caractérisent entre autres par des points forts et des points faibles. Parmi les avantages apparents des structures récurrentes, on constate que :

— la descente du gradient dans l'apprentissage est rapide, décroissante et stable;
— la convergence est quasi-absolue;
— le temps de convergence est court;
— les états actuels des cellules de sorties sont mieux définis (connaissance des états précédents);
— le gradient est guidé dans la recherche du minimum.

Parmi les inconvénients, on remarque:

— le besoin d'une grande capacité de mémoire pour stocker l'information: matrices des poids pour les états précédents;
— la faiblesse dans la généralisation ;
— la faiblesse à composer avec des données bruitées [5].

Pour ce qui est de l'autre type d'architecture à propagation-avant, elle est connue pour son traitement statique. C'est un traitement qui s'apparente au précédent mais qui n'utilise pas de rétroaction (feed-back). La majorité des applications trouvées dans la littérature sur les réseaux de neurones utilise ce type de réseau: l'architecture à propagation-avant. Ce modèle est caractérisé par une faiblesse dans le traitement dynamique (signal dont les paramètres varient dans le temps) de l'information étant donné sa nature statique, mais ceci ne fait pas l'unanimité auprès des scientifiques concernés. Une pléthore de travaux ont été effectués sur ce type de réseaux, particulièrement depuis l'apparition de l'algorithme de la rétropropagation [22]. La principale force de ce type de réseau réside dans sa capacité de discrimination et de classification.

Notre problème consiste à étudier le filtrage d'un signal dynamique bruité avec un bruit gaussien ayant des rapports signaux / bruit variables (20 dB à –20 dB). Notre objectif consiste à supprimer si possible le bruit qui se trouve dans le signal dynamique et non pas à contrôler un procédé (qui se fait mieux par le biais d'une architecture récurrente [96]). C'est pour ces raisons et celles citées précédemment (grande capacité mémoire, faiblesse dans la généralisation et faiblesse à composer avec des données bruitées) que nous avons orienté notre recherche vers l'utilisation d'une architecture à propagation-avant.

Ce travail démontre la capacité d'un réseau de neurones à propagation-avant à mémoriser un signal périodique, à le regénérer de façon indéfinie (dépendamment du signal à l'entrée) et à généraliser au niveau de l'amplitude et des fréquences. Ces qualités nouvellement observées dans le cadre de ce travail portant sur une architecture de type propagation-avant ont permis de proposer un modèle de pseudo-filtre réducteur de bruit comme un passe-bas, un passe-haut, un passe-bande et un coupe-bande, basé sur ce type d'architecture.

La principale force du modèle proposé réside dans sa capacité de généraliser au niveau des fréquences et de l'amplitude.

CHAPITRE 1

REVUE DE LA BIBLIOGRAHIE

Dans ce chapitre, nous exposerons notre sujet: le filtrage d'un signal dynamique (signal dont les paramètres changent dans le temps), dans le contexte de l'évolution de la technologie des réseaux de neurones. Nous présenterons d'abord une vue globale de cette approche, ensuite, nous ferons la synthèse des travaux effectués dans ce champ qui pourraient éventuellement contribuer, sur le plan théorique et expérimental à la conception de notre modèle. Finalement, au fur et à mesure que nous analyserons ces travaux, nous dégagerons les lacunes et les points importants qui seront retenus dans cette approche. En terminant, nous ferons un rappel des notions du champ de la parole.

1.1 Qu'entend - on par réseaux de neurones ?

1.1.1 Intelligence artificielle — réseaux de neurones

L'idée de machines intelligentes est apparue au début de ce siècle, bien avant l'arrivée des ordinateurs numériques. Ce concept a été vulgarisé par les auteurs de science fiction dans leurs écrits et par les réalisateurs de films fantastiques. Ils rêvaient du jour où une machine parlerait, piloterait un avion, effectuerait le ménage dans une maison, ou voire même, construirait des machines complexes.

Très tôt, la recherche en I.A (Intelligence Artificielle) prît deux orientations: la première concerne la modélisation des appareils et les interactions du cerveau humain. Cette approche microscopique a été inspirée des travaux de Warren S. Mc Culloch et Walter Pitts en 1943. La seconde se base sur la modélisation des fonctions du cerveau humain et de l'intelligence d'un point de vue macroscopique (boîte noire).

Les deux approches ont vu le jour au même moment: au début des années '50. Elles utilisaient de très grosses approximations pour fins de modélisation. Vers la fin des années '60, elles prenaient une autre orientation, car les performances des systèmes d'ordinateurs à l'époque limitaient leur capacité et retardaient leur progrès, ce qui paralysa la recherche pendant quelques temps.

Une des premières tentatives de l'approche macroscopique "boîte noire" pour modéliser une machine intelligente fût le programme écrit par Marvin Minsky, pour démontrer des théorèmes (preuves) géométriques. Ce travail était à l'origine de plusieurs branches de recherches fondatrices en matière de traitement de l'information et de stockage des concepts et des connaissances sous forme de règles, schémas, scripts, réseaux sémantiques, etc.... Ces multiples succès sont à la base de ce qu'on appelle aujourd'hui " l'intelligence artificielle" ou I.A.

Au milieu des années '70, l'avènement d'ordinateurs bon marché et facilement accessibles couronnait de longues années de recherche. Cette tendance s'accroît dans les années '80 avec l'introduction des micro-ordinateurs (Personal Computer) et les stations individuelles de travail ayant les mêmes performances que certains "main frames". Dès lors, les recherches se multiplient dans les domaines des neuro-sciences, de la neuro-biologie, de la psychologie, des sciences cognitives et de beaucoup d'autres; en particulier ceux qui s'intéressent à modéliser et tester des théories en relation avec le fonctionnement du cerveau humain. Ces investigations laissent la place à l'exploration des configurations des réseaux de neurones résolvant ainsi les problèmes qui bloquaient autrefois l'évolution de certains travaux de recherche.

1.1.2 L'approche neurale — réseaux de neurones

L'approche des réseaux de neurones est une technologie de traitement de l'information. On peut la voir comme une approche d'un niveau supérieur, utilisant des langages procéduraux comme outils de programmation (Basic, Fortran, C, etc...) dans le but de

concevoir l'architecture du réseau, d'une part, et d'assumer son fonctionnement (apprentissage et reconnaissance) d'autre part. L'approche des réseaux de neurones n'utilise pas un langage particulier comme c'est le cas pour la technologie des systèmes experts (lisp, Prolog, etc... — programmation objet —); la simulation des réseaux de neurones s'exécute sur les mêmes systèmes d'ordinateurs (hardware). Cependant chacune de ces technologies i.e celle des réseaux de neurones et des systèmes experts, possède une approche spécifique du traitement de l'information, ce qui en fait un meilleur outil, adapté à la résolution de problèmes particuliers.

Le principe de fonctionnement des réseaux de neurones multi-couches consiste à établir des relations intrinsèques entre les données fournies à l'entrée et à la sortie du réseau. En d'autres termes, le réseau doit apprendre la fonction **F** qui existe entre l'entrée et la sortie de celui-ci. Cette relation se trouve incarnée dans les données présentées lors de l'apprentissage.

Les applications des réseaux de neurones s'étendent à plusieurs champs, dont la médecine, la géologie, la robotique, l'inspection structurale, le contrôle de processus, l'inspection dans l'industrie manufacturière, l'analyse et le traitement de l'image, les télécommunications, le filtrage du signal (parole ou image) , etc...

L'unité de base du traitement de l'information dans les réseaux de neurones est le neurone ou "Processing Element" (unité de traitement). Les recherches sur le cerveau ont identifié au-delà de 300 types différents de neurones. L'unité de traitement citée précédemment en fait partie. Chaque variété de neurone se trouvant dans le réseau opère à sa manière et caratérise l'implantation d'une fonction spécifique.

La reconnaissance de la parole doit tenir compte de la nature statistique et séquentielle du système de production. Les modèles de Markov cachés permettent de faire face à ces deux aspects . Cependant, le choix a priori de la topologie de ces modèles et leur manque en discrimination des patrons limitent leurs capacités et en constitue le point faible.

Récemment les modèles connexionnistes ont été reconnus comme étant une alternative

intéressante. Leur propriété principale réside dans la discrimination, par la capture des relations d'entrées-sorties. Ils ont prouvé leurs capacités avec les données statiques. Cependant, les aspects série et dynamique sont toujours difficiles à traiter dans ce genre de modèle. Plusieurs auteurs ont proposé des architectures. Nous aurons d'ailleurs l'occasion de les mentionner dans ce chapitre.

À l'heure actuelle, il existe plusieurs modèles de reconnaissance de la parole très performants, utilisant différentes technologies. Néanmoins, dès qu'ils se trouvent dans un milieu bruité, leurs performances s'affaiblissent [50, 51, 52] . Cela nécessite un outil adapté aux besoins pour y remédier, d'où l'intérêt du sujet de notre recherche: la conception d'un filtre pour un signal bruité.

1.2 Approche globale: technologie des réseaux neuronaux — Historique

L'importance du filtrage d'un signal bruité a motivé de nombreux chercheurs et ingénieurs depuis plusieurs décennies, notamment dans le domaine de l'instrumentation d'appareillage électrique et dans celui des télécommunications (la radiophonie, la téléphonie et plus particulièrement, avec le développement de l'informatique, le champ de la communication vocale homme-machine: systèmes de reconnaissance de la parole). Les aspects dynamiques et séquentiels de la parole la rendent difficile de traitement, surtout dans un milieu bruité (milieu industriel ou tout autre environnement de travail).

Maints groupes de recherche se sont penchés sur le problème du filtrage de bruit, utilisant des algorithmes et techniques développés dans des domaines tels que le traitement des signaux, les mathématiques et l'intelligence artificielle (I.A). L'I.A est d'ailleurs le théâtre d'une compétition entre deux écoles de pensée - approche symbolique et approche neurale - , querelle qui est bénéfique pour le développement de la théorie des réseaux de neurones formels.

Devant les capacités de discrimination et de classification d'un réseau de neurones formels, les auteurs consultés ont étudié différentes architectures et algorithmes

d'apprentissage pour des tâches multiples et ce, dans des domaines tels que la robotique, l'inspection et le contrôle (industrie manufacturière), la médecine, la géologie, les télécommunications, la reconnaissance et la synthèse de la parole, le contrôle de processus, la classification et le filtrage d'un signal, etc... Compte tenu de la prolifération des résultats assez intéressants, une synthèse s'impose.

Dans la littérature traitant des réseaux de neurones, peu de travaux sont consacrés au filtrage d'un signal dynamique utilisant des architectures à propagation-avant (feed-forward). Le plus populaire d'entre eux est celui proposé par l'équipe de Chin'ichi Tamura et Alex Waibel [81]. Les autres travaux affirment que le traitement d'un signal dynamique se fait uniquement par les réseaux de neurones récurrents [38, 86, 96]. Il s'agit d'un sujet controversé au sein de la communauté scientifique.

L'un des objectifs de ce mémoire est de lever le voile sur cet aspect du traitement dynamique d'un réseau à propagation-avant utilisant l'algorithme de la rétropropagation rapide.

L'évolution de la théorie des réseaux de neurones formels est liée directement au développement des travaux biologiques sur le cerveau humain. Ce dernier est l'organe de commande le plus complexe et le plus inconnu de la biologie de l'homme ou de l'animal. Les cellules nerveuses, appelées neurones, sont les éléments de base du système nerveux central. Ce dernier en possèderait environ cent milliards. Dans leur organisation générale et leur système biochimique les neurones possèdent de nombreux points communs avec les autres cellules. Ils présentent cependant des caractéristiques qui leur sont propres et qui se retrouvent au niveau des quatres fonctions spécialisées qu'ils assument:

— recevoir des signaux en provenance des neurones voisins;

— intégrer ces signaux;

— engendrer un influx nerveux;

— conduire et transmettre l'influx nerveux à un neurone capable de le recevoir.

La structure est constituée de trois parties: le corps cellulaire, les dendrites et l'axone.

En se basant sur ces notions biologiques, M. Pitts a modélisé un neurone artificiel pour la première fois en 1943. Quelques années plus tard, en 1949, M. Donald Hebb a proposé la règle stipulant que deux neurones s'activent avec une augmentation du potentiel membranaire et s'inhibent avec une diminution de celui-ci . Les années '50 ont vu l'apparition du "perceptron" de David Rumelhart (1957) [22], qui a marqué le monde de la recherche et a ouvert la porte d'un domaine jusque-là quasi-inaccessible: la reconnaissance automatique d'objets. Il s'en suivra une vague de publications dans le champ des réseaux de neurones, jusqu'à ce que Marvin Misky (1969) démontre les limitations du perceptron à discriminer des éléments non-linéairement séparables. En même temps, des résultats intéressants apparaissent avec le travail de M.J.Anderson (1969) [3], qui a étudié le principe de la mémoire distributive - mémoire utilisant une distribution de poids comme outil pour le stockage de l'information -, principe qui domine largement dans les réseaux de neurones. Simultanément, les travaux de M.Grossberg [31] et Kohonen sont publiés. Ils se sont penchés sur les réseaux de neurones de type associatif. Ces travaux seront la base du modèle d'Hopfield, (début des années 80) et des cartes de Kohonen (Kohonen Maps).

Les années'70 étaient plutôt tranquilles du point de vue des publications sur l'approche neurale, mais les laboratoires de recherche étaient tout de même actifs dans diverses universités. Il est à noter que ces années ont connu le développement de la micro-informatique. D'ailleurs, une conséquence de ce développement se manifeste durant les années '80, à travers la multitude de travaux utilisant l'approche neurale. Le coup d'envoi a été donné par Hopfield en 1982 [31]. La participation de ce scientifique a accentué l'intérêt pour l'approche neurale. Le réseau d'Hopfield n'a pas une origine biologique en tant que tel (comme c'est le cas pour l'architecture à propagation-avant), mais plutôt physique. Il est à remarquer que ce type de réseaux est le plus étudié et le mieux compris dans l'approche neurale [31]. La domination de ce type de réseau a duré environ cinq ans. En même temps, d'autres travaux continuent à explorer le perceptron et l'adaline [94]. Cette évolution

s'accompagne de l'apparition d'un certain nombre de problèmes dont le principal est le "Credit Assignement" (discussion en annexe A). Au fur et à mesure, plusieurs algorithmes sont alors proposés pour régler partiellement ce problème, jusqu'à la publication des travaux de David Rumelhart [22], qui donna à la recherche sur les réseaux de neurones un bon élan. Nous pouvons dire que c'est grâce à cet algorithme que l'approche neurale a connu le succès qu'elle a aujourd'hui. L'algorithme développé par Rumelhart, David et Williams [22] a connu un succès de taille, plus précisément avec le réseau ALVINN [72]. Dès lors, au fil des ans se développent de nombreux travaux pour améliorer les côtés problématiques de cet algorithme, contrôlant ainsi la stabilité, la convergence, la mémorisation et la généralisation. En outre, il y a même une version récurrente de la rétropropagation qui a été developpée par Paul Werbos en 1990 [86]. Parmi les algorithmes les plus populaires qui dérivent de la rétropropagation, on retrouve la rétropropagation standard D.Rumelhart [22], la rétropropagation cumulée [31] et la rétropropagation rapide [82], [83]. Chacun de ces algorithmes résout un aspect bien particulier du problème mais parfois en engendre d'autres tels que la dégradation de la généralisation. Pour ce qui est de la rétropropagation rapide, le gradient de l'erreur est assisté dans la recherche de l'orientation, ce qui réduit le temps d'apprentissage, et par conséquent celui de la convergence. Par contre, il cause le problème des minima locaux. Quant à la rétropropagation standard, elle n'a pas de repère pour la guider, ce qui se réflète dans la longueur de temps d'apprentissage. Cependant, elle réduit le problème des minima locaux. Ce qui est vrai pour la rétropropagation rapide, l'est aussi pour la rétropropagation cumulée, sauf que cette dernière s'accompagne d'une augmentation du temps d'apprentissage, puisque cet algorithme (la rétropropagation cumulée) fait une moyenne des données d'apprentissage à l'entrée avant d'effectuer une mise à jour des poids (la partie théorique est jointe en annexe A). Ce qui n'est pas le cas des deux autres algorithmes pour lesquels la mise à jour des poids se fait après le passage de chaque vecteur de données.

On se pose un certain nombre de questions sur le principe de fonctionnement de cet

algorithme, qui semble tenir compte de la contribution de chaque noeud dans l'évolution de l'erreur globale. Pour connaître ses origines, les travaux d'Halbert White sont une bonne référence [87, 90, 92]. Il a fait une étude approfondie établissant le parallèle entre les méthodes statistiques et la rétropropagation. Il a démontré que la rétropropagation n'est qu'un cas particulier des méthodes approximatives traitées par Robbin et Monro en 1951 [74].

En terminant cette partie, nous aimerions mentionner que le modèle de M.Jordan [38] fût développé pour une architecture séquentielle récurrente, fonctionnant sous un algorithme très intéressant, dont le champ d'application est la production de la parole. Certains algorithmes aussi performants les uns que les autres fonctionnent sur une architecture récurrente, tel celui qui a été developpé par Zipser et Williams [96] et qui s'adapte bien au traitement d'un signal dynamique [40].

Parmi les architectures les plus populaires dans le champ des réseaux de neurones, on trouve deux types: les réseaux mono-couches et les réseaux multi-couches. Le premier type se compose d'un seul neurone de décision; le plus souvent, c'est le perceptron ou l'adaline. Il est largement utilisé dans la classification des patrons et dans le filtrage adaptatif. Il est capable d'effectuer une classification linéairement séparable seulement. Il différencie deux états dans le plan xy par une séparation rectiligne. Ce type de réseau n'est pas en mesure d'effectuer une classification non-linéairement séparable, ce qui restreint ainsi son champ d'application. Le second type (multi-couches) se compose de plusieurs neurones pouvant être regroupés soit en couches hiérarchisées ou en une structure complètement interconnectée. Dans la structure non hiérachique, les noeuds sont entièrement connectés. Le réseau peut incorporer deux types d'apprentissage: l'apprentissage compétitif (cf. Crossberg , C.von Malsburg et Kohonen [31]) et l'apprentisssage associatif (cf. Hopfield). Le premier type incarne une forme de compétition entre les noeuds jusqu'à l'atteinte d'un état stable. Il est largement utilisé dans le spatio-temporel (faisant appel à la dérivée d'un signal) et dans le filtrage. Quant à l'apprentissage associatif , il est largement utilisé en

reconnaissance des formes (reconnaissance des patrons statiques).

La structure hiérarchisée se compose d'un ensemble de noeuds regroupés en couches soigneusement sélectionnées. La propagation de l'information se fait dans un seul sens et l'ajustement des poids (connexions synaptiques) dans l'autre. Les principales architectures sont le réseau à propagation-avant (feed-forward) et l'architecture récurrente de M.Jordan [38], développés presque en même temps que la rétropropagation [22]. L'architecture de Jordan fait un traitement séquentiel récurrent. L'architecture récurrente parallèle de Paul Werbos [86], Zipser et Williams [96], Halbert White [87, 90, 92], fait des retours sur la couche cachée. Cette dernière architecture est bien adaptée au traitement d'un signal dynamique. Les principaux champ d'application de ce type d'architecture sont le contrôle de processus, la classification, le traitement du signal et le filtrage.

Nous avons également vu l'apparition de travaux portant sur l'architecture à propagation-avant, sur l'étude de la fonction sigmoïde [55], sur le nombre de noeuds de la couche cachée [6], [31] et sur le nombre de patrons à mémoriser en fonction du nombre de noeuds à l'entrée [77]. Le groupe de recherche Chin'ichi Tamura et d'Alex Waibel [81] a démontré la capacité d'un réseau de neurones pour le filtrage de la parole bruitée. Il s'agit des données (216 mots phonétiquement équilibrés) de la langue japonaise.

La décennie des années '90 est marquée par l'amélioration de l'efficacité des réseaux de neurones dans plusieurs champ tel que la reconnaissance des formes. La comparaison des performances des modèles basés sur l'approche neurale illustre une nette amélioration par rapport à ceux basé sur l'approche classiques (K moyennes et les K plus proches voisins) [19, 100]. De plus, les années '90 ont connu l'évolution liée à la conception des systèmes hybrides: l'utilisation simultanée de plusieurs réseaux ou celle d'un système expert et d'un réseau de neurones.

Maintenant que nous avons donné un aperçu global de la technologie des réseaux de neurones, revenons au sujet précis de ce mémoire: le traitement d'un signal dynamique. Parmi les chercheurs qui se sont les plus illustrés dans ce domaine, on retrouve l'équipe de

Chin'ichi Tamura et Alex Waibel [81] qui a travaillé sur le filtrage de la parole bruitée, en utilisant un réseau à propagation-avant. Il y a aussi l'équipe de Ryotaro Kamimura [40] qui a démontré, sur une architecture récurrente, la capacité de mémoriser et de reproduire un signal périodique. Ce réseau utilise l'algorithme "TSLA" développé par Zipser et Williams [96].

1.3 Choix de l'architecture du réseau

1.3.1 Vue macroscopique

Un réseau de neurones est un ensemble d'unités de traitement, "processing elements", disposées d'une telle façon qu'elles forment une architecture. Ces architectures sont de types divers et voici les principaux:

1.3.1.1 Structure mono-couche

C'est une architecture ayant un ensemble de **n** entrées (vecteurs d'entrée de dimension n), une cellule de discrimination (s), et un élément de sortie (y) (cf. figure 1.1). Pour rendre ce réseau fonctionnel et dynamique, il faut un algorithme qui modifie les valeurs des connexions synaptiques (poids). Cet algorithme agit au niveau des connexions en changeant la valeur du lien entre deux ou plusieurs neurones.

Ce type de réseau effectue un filtrage linéaire (suppression du seuil: y=s); il s'apparente à celui des filtres adaptatifs classiques. Les travaux de Bernard Widrow en sont un bon exemple.

Lorsqu'on veut établir une classification avec ce type de réseau, il présente une limitation sur le type de données à traiter. La principale contrainte réside dans son incapacité à effectuer une classification telle que les éléments soient non-linéairement séparables (ex: réaliser l'opération du "**ou** exclusif").

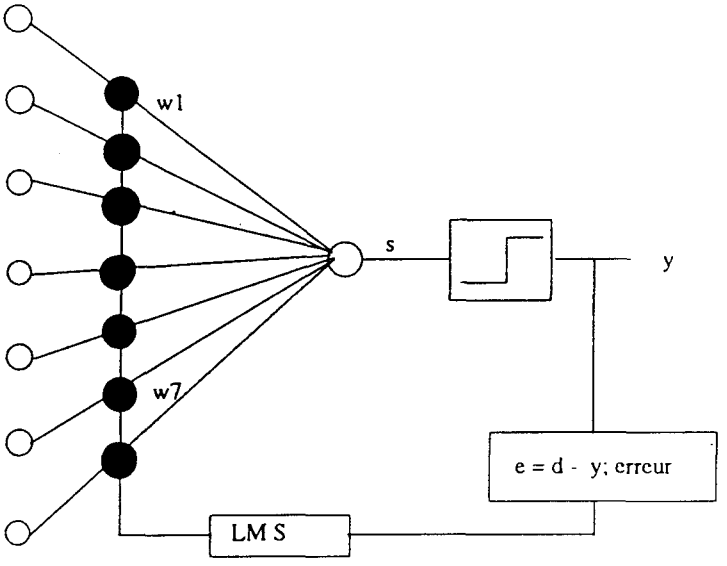

Figure 1.1 Réseau adaptatif : le perceptron.

1.3.1.2 Structure multi-couche

1.3.1.2.1 Architecture entièrement connectée

C'est une architecture ayant un ensemble de cellules d'entrée (vecteur de dimension n), relié directement à un ensemble de cellules qui produisent une sortie. Ce type de réseau possède généralement des connexions symétriques. La mémorisation de l'information dans le réseau de neurones doit converger vers des états stables. Cependant, la principale contrainte de ce type de réseau demeure la capacité de mémorisation très limitée. Un réseau de **n** noeuds n'est pas en mesure de mémoriser plus qu'un nombre très faible de patrons différents (**0.14.n** dans l'article [59]). Donc, la taille du réseau grandit avec la taille des données, ce qui peut engendrer un phénomène d'oubli.

Le principal champ d'application de ce type de réseau est la reconnaissance des formes. Le modèle le plus populaire est celui d'Hopfield, développé en 1982 . Ce type d'architecture se range dans la catégorie des réseaux dits "associatifs". La structure de l'architecture est illustrée sur la figure 1.2.

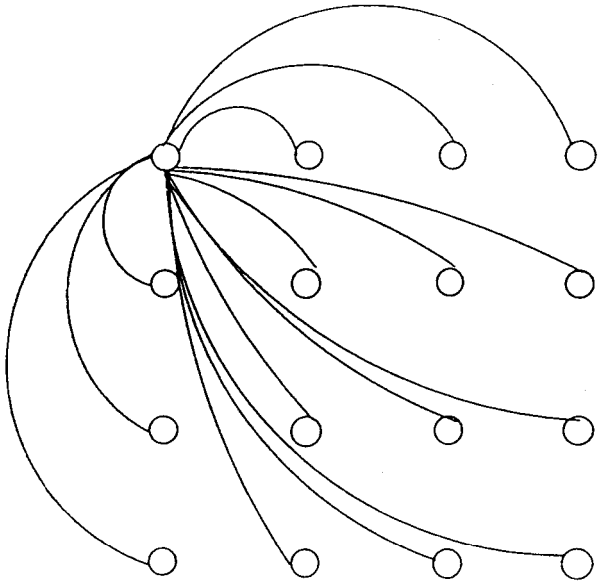

Figure 1.2 Modèle du réseau d'Hopfield: illustration des connexions pour un seul neurone.

1.3.1.1 Architecture hiérarchisée

Contrairement à l'architecture précédente, cette structure regroupe ses noeuds en plusieurs couches hiérarchisées dont les principales sont appelées couche d'entrée, couche cachée (une ou plusieurs) et couche de sortie. La première couche d'entrée est une couche linéaire n'effectuant aucune opération spéciale, mise à part la transmission de l'information d'entrée à la couche suivante. Dépendamment des réseaux de neurones, la ou les couche(s) cachée(s) possèdent des neurones qui sont modélisés par un sommateur à l'entrée et une fonction de transfert à la sortie. Cette dernière peut être linéaire ou non-linéaire. La couche de sortie peut être une couche linéaire ou non-linéaire, dépendamment de l'objectif du concepteur et des caractéristiques qu'il veut en faire ressortir. Le traitement se fait dans un seul sens et de façon séquentielle (couche à couche), d'où le nom d'architecture à propagation-avant (figure 1.3). Elle est reconnue pour son traitement statique (cette affirmation est néanmoins controversée dans la communauté scientifique). Même si elle est très peu utilisée pour le traitement dynamique de l'information, elle permet l'utilisation de plusieurs algorithmes d'apprentissage. Cependant, il existe une version récurrente de l'algorithme, développé par Paul Werbos [86], qui s'adapte au traitement dynamique de

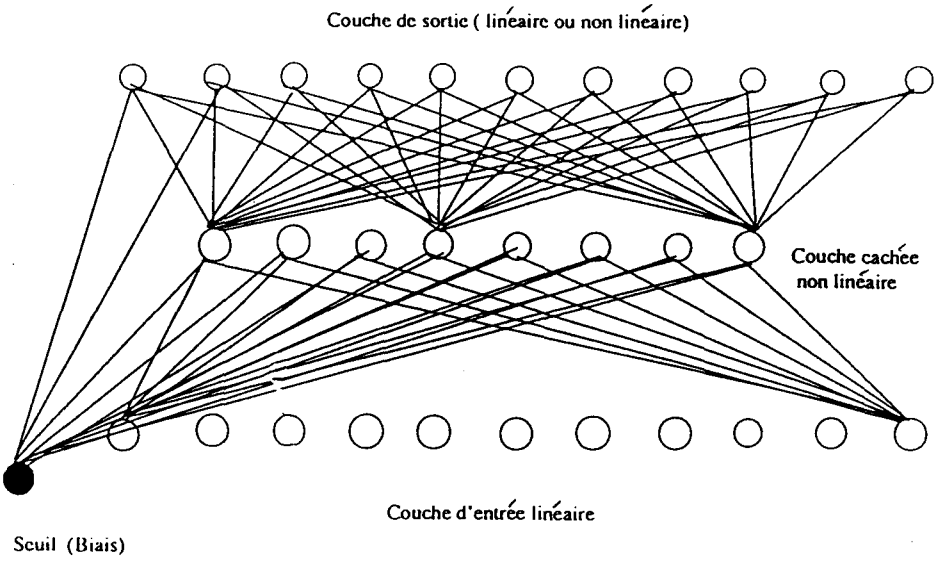

Figure 1.3 Modèle de réseau multi-couches à propagation-avant "feed-forward".

l'information. Toutefois, cette dernière donne de faibles performances en présence de bruit. La capacité d'un réseau à mémoriser l'information est directement liée au nombre de noeuds dans la couche cachée et non pas au nombre de noeuds dans tout le réseau comme c'est le cas pour le type de J. Hopfield (1982).

Cette architecture est assez souple pour permettre la possibilité de changer la fonction de transfert, pour éventuellement étudier son influence en fonction de la discrimination, de la convergence, de la mémorisation et de la généralisation.

Il est à remarquer que nous avons à peine évoqué l'architecture récurrente. Elle fonctionne selon le même principe que la propagation-avant, excepté qu'elle garde en mémoire plusieurs états antérieurs. La mémorisation de l'information précédente a pour conséquence d'aider à la convergence, et aussi de réduire le temps d'apprentissage [22].

Un autre exemple de structure récurrente est celui du modèle de Jordan illustré par la figure 1.4 (toutes les connexions n'ont pas été reportées pour fins de simplification). Cette structure est principalement utilisée dans la production de séquences. Ces dernières dépendent de l'information précédente pour déterminer l'état de la prochaine sortie du réseau de neurones. Cette structure n'est pas utilisée pour le filtrage d'un signal bruité.

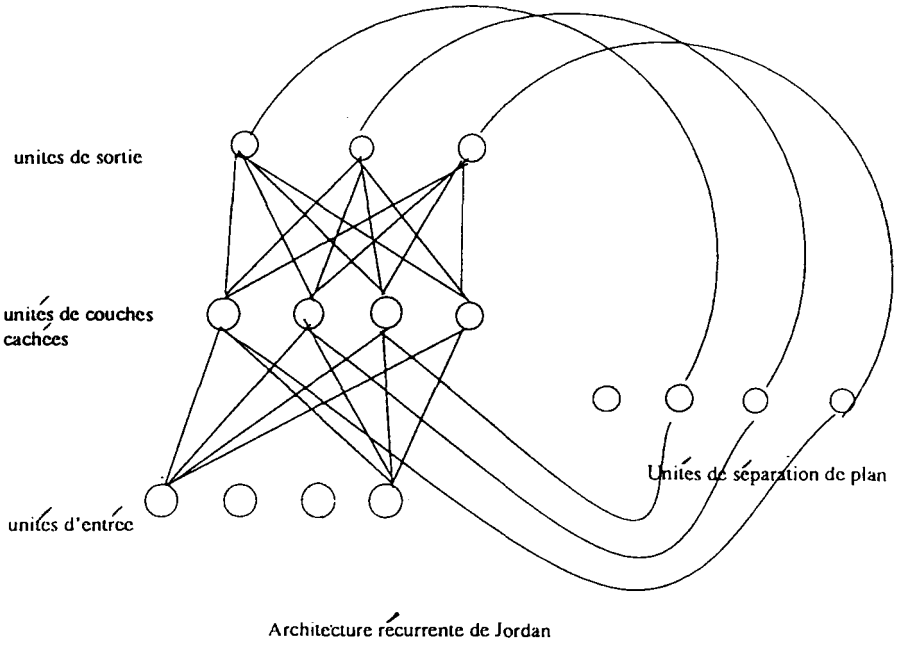

Figure 1.4 Réseau récurrent séquentiel de Jordan.

1.4 Qu'est ce qu'un algorithme d'apprentissage ?

L'algorithme d'apprentissage est l'opération qui permet de modifier les poids selon l'évolution de l'erreur entre la sortie obtenue et la sortie désirée. Il contrôle tout le comportement (la dynamique) futur du réseau de neurones. La capacité de mémorisation et de généralisation du réseau réside dans la configuration de ses poids. L'algorithme d'apprentissage est de deux types: l'apprentissage supervisé et l'apprentissage non-supervisé. Le premier type a besoin des données à l'entrée et à la sortie du réseau pour apprendre l'information se trouvant dans les données, tandis que le second a besoin des données à l'entrée seulement, et celles à la sortie sont générées par un principe tout à fait différent comme par exemple l'auto-organisation ("self-organisation").

1.5 Approche spécifique: réseaux de neurones - capacité de traiter un signal - filtrage -

Dans la section précédente, nous avons replacé notre sujet dans le cadre de l'évolution de cette technologie. Par la suite, nous ferons un recul pour ne discuter que des travaux qui touchent spécifiquement notre sujet.

L'utilisation des réseaux de neurones pour une application aussi particulière que le filtrage nécessite un double choix : l'architecture du réseau et l'algorithme d'apprentissage. L'architecture a fait état d'une brève analyse dans plusieurs travaux dont les principaux seront exposés dans cette section.

Dans la section précédente, nous avons vu que les réseaux de neurones multi-couches comprennent l'architecture à propagation-avant (feed-forward). Celle-ci peut s'adapter à différents algorithmes d'apprentissage: supervisé ou non. En général, l'apprentissage non-supervisé porte sur des architectures différentes de la propagation-avant et comprend l'auto-adaptation "self-adaptation", l'apprentissage compétitif (la règle de Kohonen ou la règle de Grossberg) et le spatio-temporel qui utilise la règle de Kosko/klopf. Quant à l'apprentissage par rétropropagation (supervisé), il utilise presque exclusivement l'architecture à propagation-avant (feed-forward), connue pour son traitement statique (classification des patrons). Cette affirmation ne fait pas l'unanimité des chercheurs. Ce type de réseau est utilisé pour la compression des données, dans le calcul de prédiction, dans le filtrage, ...etc. Il y a un autre type de réseau déduit du précédent dit "récurrent". C'est une architecture qui s'apparente à la propagation-avant et qui comprend des connexions récurrentes. Ce type de réseau a été décrit par Paul Werbos [86], Zipser et Williams [96]. Il est caractérisé par sa facilité à traiter un signal dynamique (signal de contrôle d'un procesus). Cependant, il a un faible rendement en présence du bruit.

1.5.1 Une convergence rapide: fonctions radiales, Chris Bishop 91 [10]

Dans un article publié en 1991 [10], Chris Bishop analyse la convergence d'un algorithme utilisant les fonctions radiales sur une architecture à propagation-avant, qu'il applique sur des signaux purs. Les neurones utilisés sont modélisés par un sommateur à l'entrée et une fonction de transfert à la sortie. Dans cette étude, la fonction utilisée est la loi normale (loi de Gauss). L'algorithme consiste à prendre un vecteur de données à l'entrée du réseau et à effectuer un produit scalaire avec les vecteurs des poids qui, à leurs tours sont mis à jour par un algorithme de gradient. Ensuite, les entrées de la couche cachée

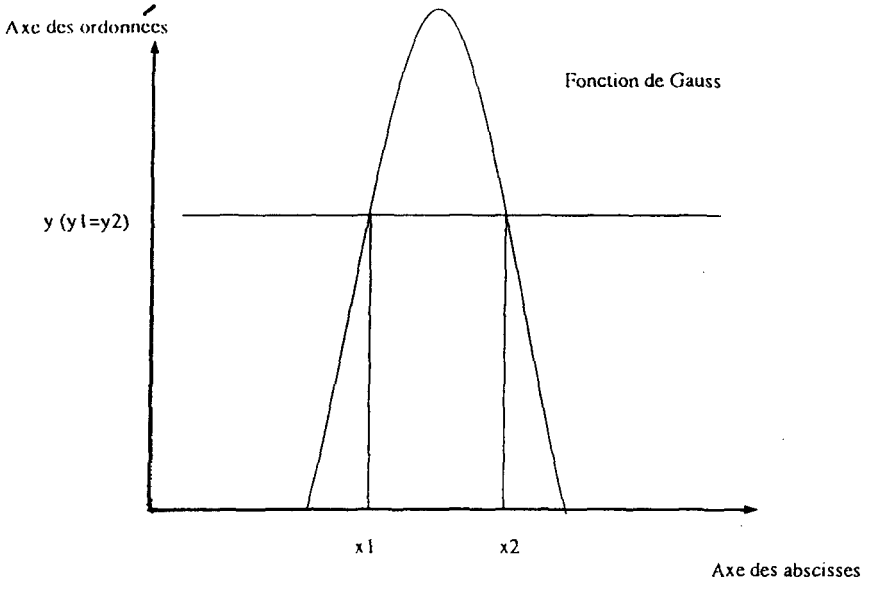

Figure 1.5 Illustration de la confusion que peut générer une fonction normale (deux antécédents pour une même image).

(résultats du produit scalaire) sont soustraites d'un élément central (élément qui appartient au vecteur initial; le rang de ce dernier correspond à la position du noeud dans la couche cachée). Finalement, le résultat est évalué par la fonction normale pour cheminer vers la sortie. L'adaptation des poids se fait par un algorithme de gradient qui tient compte de l'évolution de l'erreur. L'algorithme semble être performant . Cependant, le choix des éléments centraux pose un problème. De plus, le fait d'utiliser une loi normale comme fonction de transfert génère de la confusion lors de la discrimination. Ce phénomène est illustré par le graphique 1.5.

Le problème de confusion est caractéristique des fonctions de transfert qui ont au moins deux antécédents pour une même image (x1 et x2 ont la même ordonnée y (y=y1=y2)). Ce phénomène n'existe pas avec la fonction sigmoïde possèdant une dérivée croissante ou décroissante telle qu'illustrée dans la figure 1.6

1.5.2 Mémorisation et regénération de patrons périodiques, Ryotaro Kamimura [40]

Ryato Kamimura [40] a étudié la mémorisation des patrons périodiques par un réseau

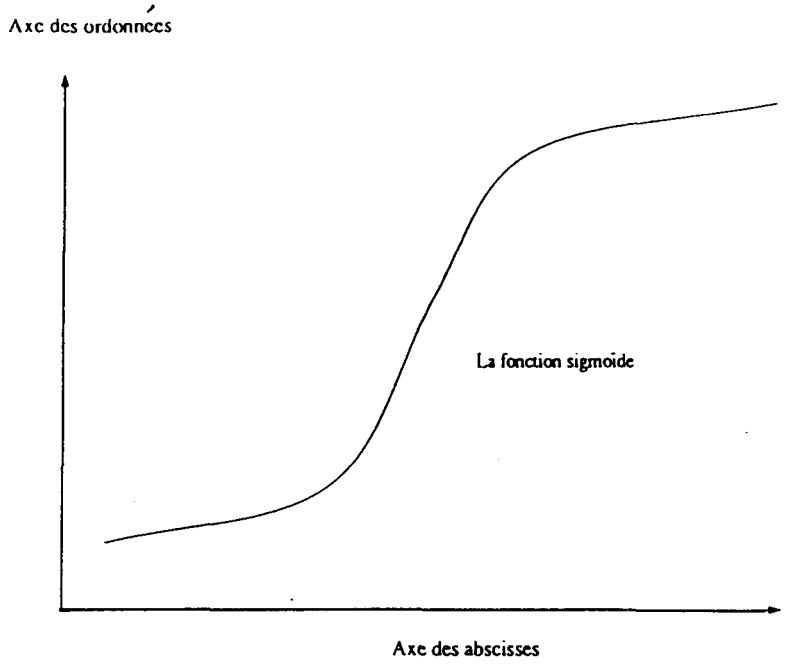

Figure 1.6 Illustration de la fonction sigmoïde

récurrent. L'algorithme d'apprentissage utilisé est celui de Zipser et Williams [96]. Le modèle mémorise adéquatement un patron périodique (séquence de signal) et le regénère avec une précision satisfaisante. Le nombre de noeuds dans la couche cachée varie de 8 à 11 unités. Le modèle est moins performant (introduction d'un léger déphasage au milieu et à la fin du signal) en présence de patrons complexes, notamment en réalisant des opérations mathématiques élémentaires telles que l'addition et la multiplication sur des signaux sinusoïdaux. L'aspect de généralisation du réseau est moins élaboré dans cet article. Ce travail demeure néanmoins une bonne référence pour le modèle developpé dans le cadre de notre recherche.

La principale chose à retenir de ce travail est la capacité du réseau à mémoriser un signal périodique et à le regénérer sans dégradation.

1.5.3 Filtrage d'un signal dynamique: parole bruitée, Chin'ichi Tamura & Alex Waibel [81]

Chin'ichi Tamura et Alex Waibel [81] ont étudié la capacité de filtrage d'un réseau à propagation-avant avec l'utilisation d'une architecture à propagation-avant de 4 couches

dont chacune possède soixante noeuds. Les couches d'entrée et les couches de sortie possèdent des noeuds ayant une fonction de transfert linéaire (d'où l'appellation "couches linéaires"), contrairement aux couches intermédiaires qui sont non-linéaires. L'algorithme utilisé est celui de David Rumelhart et Williams [22]: rétropropagation standard. Le modèle a effectué l'apprentissage supervisé sur une base de données comportant 216 mots phonétiquement équilibrés de la langue japonaise. Le protocole d'apprentissage consistait à bruiter la parole avec deux types de bruit (stationnaire, avec des propriétés constantes dans le temps, ou non-stationaire) avec un rapport signal à bruit de – 20 dB. L'apprentissage d'une durée de trois semaines se faisait à l'aide d'un ordinateur de haute performance. Le modèle est performant en ce qui concerne les données qui lui sont présentées à l'apprentissage. La taille de la base de données et de l'architecture (240 noeuds) ainsi que la durée de l'apprentissage (3 semaines) demeurent néanmoins des problèmes considérables.

Les performances du modèle sont bonnes (tendance à agir comme un filtre passe-bas). Le test d'audition a montré que le modèle du réseau de neurones est nettement supérieur au modèle de la suppression d'énergie spectrale, car 56 % des auditeurs l'ont confirmé contre 43 % pour l'autre [81].

1.5.4 Taille d'un réseau qui donnerait une bonne généralisation - Eric Baum & David Haussler [6]

Eric Baum et David Haussler [6] ont étudié du point de vue mathématique la capacité d'un réseau de neurones à mémoriser et à généraliser des patrons. Ils ont démontré la capacité de généralisation d'un réseau en fonction du nombre total de poids dans l'architecture. Ils ont par ailleurs fait une synthèse rigoureuse des théories proposées auparavant par Cover en 1965 et par Vopnik & Chevonenkis en 1971, 1982. Les réseaux utilisés sont basés sur un type de neurone ayant une fonction de transfert discrète (le perceptron ou l'adaline). Ce modèle donne les limites (inférieure et supérieure) i.e le nombre de noeuds qu'un réseau doit avoir pour pouvoir donner une généralisation valable. Nous n'avons pas jugé nécessaire de reproduire ces équations et leurs domaines d'application.

Par contre, nous pouvons les illustrer par un exemple: si on désire reconnaître des patrons avec une précision de 90 %, ce qui correspond à une incertitude (ϵ) de 10%, on doit choisir un nombre de connexions synaptiques au moins 10 fois plus grand qu'on a d'exemples (patrons) à apprendre. C'est la règle du "pouce" [94], suggérée par Bernard Widrow [94] et qui s'écrit comme suit: $W = m \cdot \frac{1}{\epsilon}$ avec

— m : le nombre d'exemples à apprendre;

— W : le nombre total des poids contenus dans le réseau de neurones;

— ϵ : l'incertitude désirée.

Ce travail nous fournit les outils mathématiques pour établir les limites dans le choix des paramètres. Cependant l'application de ces résultats est directement liée aux types de neurones choisis et dont les états sont binaires. Néanmoins, ceci nous donne une idée sur l'étendu de notre choix.

1.5.5 Peut-on réduire le nombre de couches dans un réseau de neurones formels ?— Michael A. Sartori et J. Ansaklis [77]

Michael A. Sartori et J.Ansaklis [77] ont comparé les performances de trois modèles basés sur une architecture à propagation-avant et ayant trois couches, dont une cachée. Le travail consiste à étudier une méthode simple pour dériver les limites et la taille d'un réseau de neurones multi-couches. Le présent modèle a besoin de deux couches de poids seulement pour utiliser (implémenter) n'importe quelle fonction d'apprentissage. Ce résultat est confirmé par les travaux d'Halbert White [87, 90, 92] et d'Eric Baum [5]. Le modèle est performant à condition d'avoir k-1 neurones dans la couche cachée, avec k représentant le nombre de patrons. Il est à remarquer que la plupart des travaux traitent des données statiques. Ceux qui traitent du signal dynamique se font rares.

Le modèle de N.J. Nilsson [66] utilise un réseau à propagation–avant, composé de 3 couches dont une cachée ayant k-1 neurones. Cette configuration peut utiliser un

apprentissage avec des données réelles (appartenant à l'ensemble R) à l'entrée et binaires (0,1) à la sortie.

Le modèle proposé par Eric Baum [5] utilise un réseau à propagation–avant de 3 couches, dont une cachée et ayant k neurones. Cette configuration peut utiliser un apprentissage avec des données réelles à l'entrée et binaires (0,1) à la sortie. Cette méthode est moins efficace pour résoudre k équations linéaires.

Le modèle de S.C. Huang et Y.F Huang [33] utilise un réseau à propagation–avant composé de 3 couches, dont une cachée ayant k-1 neurones. Cette configuration peut utiliser (implémenter) exactement un apprentissage avec des données réelles à l'entrée et des données à la sortie n'étant pas nécessairement scalaires.

Les trois modèles sont très similaires, néanmoins leurs performances diffèrent.

1.6 Qu'est-ce que la parole ?

1.6.1 L'appareil phonatoire: description sommaire

La parole humaine est produite par le système phonatoire. On peut l'assimiler à un tuyau acoustique de section variable, terminé à une extrémité par la glotte (c'est la partie du larynx comprise entre les cordes vocales, au sommet de la trachée), et à l'autre par les lèvres. Sa longueur est voisine de 17 cm; sa section est constamment déformée pendant la production de la parole (la "phonation") par les mouvements des lèvres, des mâchoires, de la langue et du voile du palais; cette section peut varier de 0 (lors d'une fermeture complète) à 20 cm^2. Le conduit nasal constitue un trajet auxiliaire pour la transmission de la voix. Il va du voile du palais aux narines. Sa longueur est voisine de 12 cm. Le couplage acoustique entre les conduits buccal et nasal est contrôlé par le voile. Lors de l'émission de sons non-nasalisés, le voile interdit le passage des sons dans le larynx. La cavité laryngée est divisée en trois sections situées entre la glotte et les lèvres (cavité pharyngienne, buccale et, en dérivation, la cavité nasale).

1.6.2 Principe de fonctionnement de l'appareil phonatoire

Nous pouvons assimiler l'appareil phonatoire à un instrument à vent dans lequel nous trouvons schématiquement: une source d'énergie (semblable à la soufflerie de l'orgue), un système d'excitation servant à mettre le flot d'air en vibration et enfin un système de résonateurs qui vont modifier le spectre du signal d'excitation.

La source d'énergie provient des muscles thoraciques et abdominaux; l'air est expulsé par la contraction de la cage thoracique; au cours de l'expiration, il traverse l'orifice séparant les cordes vocales (la glotte) et chemine ensuite dans le conduit vocal. Ce conduit comporte des résonateurs en série (il en est ainsi dans le pavillon laryngo-buccal) et en dérivation (c'est le cas du signal nasal). De même que dans une flûte, l'excitation produite à l'embouchure est filtrée par le tuyau de l'instrument — au gré de l'exécutant qui fait varier les résonances, modifiant ainsi la hauteur et le timbre du son — de même dans la production de la parole, le signal d'excitation (dont nous verrons les origines dans le paragraphe suivant) est-il filtré par le conduit, donnant naissance aux divers sons du langage. Ce sont les muscles articulatoires, commandant les mouvements des mâchoires, de la langue, du voile, des lèvres, du palais..., qui vont régir l'évolution de la configuration du conduit vocal et, par la suite, de ses caractéristiques acoustiques.

1.6.3 Excitation du système phonatoire

L'excitation du système phonatoire dépend étroitement des sons émis:

—La production des voyelles et de certaines consonnes (dites voisées) exige une excitation par mise en vibration des cordes vocales. La fréquence fondamentale F_0 de la voix (pitch) est celle à laquelle la glotte s'ouvre et se ferme périodiquement. Elle a pu être mesurée directement à l'aide de caméras ultrarapides (plus de 4000 images par seconde). Le signal s(t) émis par la glotte est sensiblement triangulaire; la durée d'ouverture représente 30 à 70 % de la période totale T_0. La puissance des harmoniques successives décroit au taux de 12 db par octave. Après la traversée du conduit vocal, le signal g(t) présente l'allure

visible d'un certain nombre d'impulsions glottales donnant lieu à des sortes de sinusoïdes amorties dont la "structure fine" dépend de l'intensité des diverses harmoniques.

— L'excitation peut être causée par un flot turbulent d'air créé en certains points de resserrement (constrictions) du conduit vocal. Le signal d'excitation n'est plus périodique: il s'agit d'un bruit, phénomène aléatoire, dont le spectre est pratiquement plat (l'énergie est répartie uniformément dans le spectre). Les contrictions peuvent se produire en divers endroits du conduit: langue derrière les dents, dents derrière les lèvres... Un tel mode d'excitation est à l'origine de certaines consonnes telles que les fricatives (ainsi: /s/, /f/, /ch/). Ces fricatives sont non voisées. Elles possèdent leurs correspondantes voisées, soit /z/, /v/, /ʒ/, pour lesquelles l'excitation est due à la fois à du bruit et à une vibration des cordes.

—L'excitation peut être due à une brusque variation de pression; le signal s(t) ressemble alors à l'échelon unité. La puissance dans le spectre diminue ici d'environ 6 db par octave. Dans la production d'une plosive, on observe d'abord une fermeture complète en un point particulier du conduit (cette fermeture se traduit par un silence), la pression augmente, pour retomber brusquement au moment de l'ouverture. Ici encore on distinguera des plosives (ou occlusives) non-voisées (ou sourdes): /p/, /t/, /k/ auxquelles on peut associer les correspondantes voisées (ou sonores) : /b/ /d/, /g/. Les nasales (/m/, /n/) sont des consonnes voisées. Elles correspondent à une fermeture partielle, à l'avant, du conduit buccal, l'abaissement du voile, du palais faisant du conduit nasal le canal de transmission.

— Dans le chuchotement, la vibration des cordes vocales est remplacée par un bruit dû à un flot d'air turbulent prenant naissance dans la région de la glotte partiellement fermée.

1.7 Analyse et discussion de la bibliographie

Dans ce chapitre, nous avons essayé de situer la technologie des réseaux de neurones par rapport au contexte général de l'intelligence artificielle. Ensuite, nous avons vu à travers la bibliographie présentée que très peu de travaux utilisent un réseau de neurones

formels pour le traitement d'un signal dynamique. Par contre, la plupart des travaux se rejoignent pour dire que les réseaux récurrents sont à même de relever ce défi, à savoir le traitement d'un signal dynamique. Ceci nous amène à nous poser la question suivante: quelle est la différence entre un réseau récurrent et un réseau non-récurrent? En guise de réponse, nous soulignons deux faits: 1° la rétroaction "feed-back" [71] qui incarne l'information précédente [22]; 2° la façon dont l'algorithme tient compte, dans son traitement, de l'information précédente [40]. Le fonctionnement primaire des deux structures est similaire. Donc, la rétroaction ou la technique utilisée pour tenir compte de l'information qui a précédé est principalement la cause de cette capacité à traiter un signal dynamique. Dans le cas où il y a rétroaction des connexions, nous pouvons en considérer l'effet sous forme de seuil changeant (i.e biais variant au cours du temps) dont la valeur peut être supérieure à l'unité (le seuil dans la structure à propagation-avant étant fixe et égal à 1). Cette modification se fait après l'apprentissage. Car, une fois l'apprentissage terminé, la matrice des poids est figée et invariable. Cette rétroaction va effectivement engendrer un ajout d'une constante qui varie dans le temps (qui est fonction de l'information précédente) et, par conséquent, générer un autre indice qui sera utile pour la caractérisation de l'état en question. Ce phénomène n'apparaît pas dans l'architecture à propagation-avant. Cependant, s'il s'agit d'un signal de contrôle caractérisé par un patron périodique, il est toujours préférable d'avoir le plus grand nombre d'états possibles.

Lorsqu'il s'agit, non pas de la rétroaction [58, 78, 80, 76, 15, 14, 97], mais d'une autre technique —technique qui tient compte de l'information précédente —, le problème revient à faire varier (augmenter) la valeur du seuil. L'analyse effectuée précédemment — rétroaction — peut s'appliquer sur ce cas-ci.

Maintenant, si nous voulons explorer l'architecture à propagation-avant pour explorer la limite de sa capacité, nous ferons l'apprentissage d'un signal périodique avec un choix de stratégie judicieux tel que l'apprentissage séquentiel, avec l'établissement d'une relation apparente entre les différents paramètres du réseaux. Parmi les points à étudier: la

distribution des angles que forment les données d'entrées avec les vecteurs des connexions synaptiques (poids); l'effet de la rétroaction sur la caractérisation de l'état de la sortie; l'algorithme d'apprentissage le plus performant en terme de mémorisation des relations d'entrée-sortie et de la généralisation; la corrélation entre les données d'apprentissage et la généralisation du réseau; l'effet de la normalisation des données — phase du pré-traitement — sur la généralisation de l'amplitude et des fréquences; l'influence du type des signaux sur la mémorisation et la généralisation; quel (s) paramètre (s) influence (ent) la stabilité de l'évolution de l'erreur et la convergence de l'algorithme d'apprentissage; la possibilité de réduire la taille du fichier d'apprentissage (temps de convergence); l'importance de la sigmoïde sur les performances du réseau — mémorisation et généralisation — et explorer la limite du filtrage des réseaux de neurones, particulièrement la structure à propagation-avant.

Maintenant que nous avons situé le sujet dans son contexte global, nous allons nous pencher dans les prochains chapitres sur le choix des paramètres du modèle, en vue de concevoir des filtres basés sur l'approche neurale.

CHAPITRE 2

ANALYSE ET CHOIX DES PARAMÈTRES ESSENTIELS AU FONCTIONNEMENT DU RÉSEAU DE NEURONES FORMELS: ARCHITECTURE, ALGORITHME D'APPRENTISSAGE ET MÉMORISATION DE FONCTIONS SINUSOÏDALES

Dans ce chapitre, nous présenterons le choix de l'architecture réduite (optimale en terme de noeuds et permettant une convergence rapide) utilisée pour la conception d'un filtre ayant des caractéristiques bien déterminées. Ensuite, nous détaillerons le choix de l'algorithme d'apprentissage en tenant compte de paramètres spécifiques tels que la convergence, la stabilité de l'erreur, la mémorisation et la généralisation.

Nous évoquerons l'importance de la phase de pré-traitement dans la technologie des réseaux de neurones. De plus, nous étudierons les paramètres spécifiques qui influencent la mémorisation, la généralisation d'amplitude et la généralisation de fréquence. Nous allons étudier la possibilité de réduire la taille des données à l'apprentissage.

Premièrement, pour nous familiariser avec la théorie des réseaux de neurones, nous avons utilisé les tables de vérité des portes logiques. Cette technique nous a permis d'étudier la mémorisation d'un réseau de neurones en fonction de patrons statiques. Dans un deuxième temps, nous avons utilisé des patrons périodiques (séquences de signal), pour voir si le réseau est en mesure de mémoriser et de reproduire un patron périodique avec toute l'information qu'il possède. Dans un troisième temps, nous avons analysé des segments de voyelles. Nous avons pris 3 périodes de chaque signal afin de produire un fichier, tout en respectant les critères établis dans ce travail (apprentissage sur p^2 échantillons).

Tout cela suppose une analyse appropriée du champ de la parole. Certaines relations sont déduites dans notre travail et seront respectées tout au long. Les techniques seront élaborées en détail dans ce chapitre.

Nous tenons à rappeler que le style télégraphique est utilisé pour décrire les conditions expérimentales, afin de sortir clairement l'influence des différents paramètres à étudier.

2.1 Choix de l'architecture du réseau

2.1.1 Étude microscopique

Dans cette partie, nous discuterons du type de neurone utilisé dans le modèle de réseau à proposer, du type de fonction de transfert à proposer ainsi que du nombre de couches cachées.

D'après les études faites sur le cerveau humain, il y aurait au delà de 300 types de neurones. Ils ne sont pas tous modélisés et parfois sont très difficilement modélisables. Celui que nous avons choisi dans le cadre de ce travail est représenté par un sommateur à l'entrée et une fonction de transfert à la sortie. La fonction de transfert est un paramètre important car c'est elle qui donne au réseau de neurones sa capacité de traitement non-linéaire. Elle peut être de plusieurs types: trigonométriques (sinus, cosinus, etc...); ou hyperboliques (tangente hyperbolique, sigmoïde, etc...). La fonction sigmoïde constitue la version analogique continue de la fonction échelon développée par Mc Cullogh et Pitts en 1943. D'autres types de fonctions présentent une version discrète, à l'instar de celle que l'on retrouve dans le perceptron ou l'adaline [95].

Notre but est de concevoir un filtre réducteur de bruit à même de s'adapter à des besoins spécifiques (signal dynamique). Un modèle qu'il est possible d'implanter en temps réel. Afin de rencontrer ces critères, on a besoin d'une taille de réseau réduite, pour éventuellement réduire le temps de calcul lors de l'apprentissage. En vue d'atteindre cet objectif, nous avons décidé d'utiliser un réseau de neurones composé d'une seule couche cachée. La pertinence de ce choix repose sur les critères scientifiques rigoureux établis

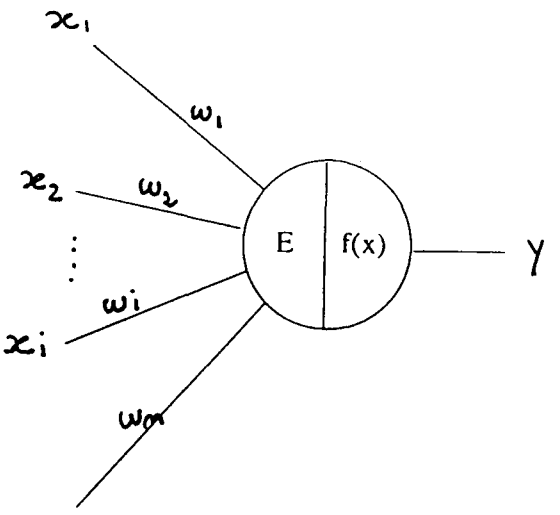

Figure 2.1 Modélisation d'un neurone artificiel : modèle de Mc Culloch et Pitts.

par Eric Baum [6], et Halbert White [87, 90, 92]. Le choix d'un réseau de neurones avec une seule couche cachée peut réaliser n'importe quelle fonction non-linéaire. Nous exploiterons cette idée dans ce travail.

2.1.1.1 Choix des paramètres de l'architecture du réseau de neurones

2.1.1.1.1 Choix du type de neurone

Comme susmentionné, il existe au delà de 300 types de neurones dans le cerveau humain. Ils ne sont pas tous modélisés dans la littérature et parfois ne sont même pas modélisables. Le type de neurone utilisé dans le cadre de ce travail, est celui représenté par un sommateur à l'entrée et une fonction analogique à la sortie du neurone. Cette fonction non-linéaire peut discriminer plusieurs états (caractère continu de la sigmoïde) par opposition à l'adaline ou le perceptron qui ne discriminent que deux états (s'applique au neurone), soit 0 et 1 ou -1 et 1. La figure 2.1 ci-dessus illustre la représentation d'un neurone.

2.1.1.1.2 Choix du nombre de couches cachées de l'architecture du réseau de neurones

Comme nous l'avons cité précédemment, notre travail a été basé sur les recherches de Paul Werbos (1990) [86] et celles d'Halbert White [87, 90, 92], recherches qui affirment

qu'une seule couche cachée peut incarner n'importe quelle fonction non-linéaire. Afin de pouvoir en tirer des conclusions, nous avons vérifié ces affirmations à l'aide de simulations. Nous avons effectivement constaté que cela est vrai, à condition que certains paramètres doivent vérifier certaines conditions, comme le nombre de noeuds dans la couche cachée, le type de neurone, l'algorithme d'apprentissage, etc...

Dans notre travail, nous avons utilisé une couche d'entrée linéaire, une couche cachée (une seule) et une couche de sortie, toutes deux non-linéaires.

2.1.1.1.3 Choix des noeuds dans l'architecture du réseau de neurones

Dans ce paragraphe, nous analyserons la relation qui existe entre le nombre de noeuds à l'entrée d'un réseau de neurones et le nombre d'échantillons dans la période d'un signal périodique.

Soit p le nombre d'échantillons (éléments) dans une période d'un signal sinusoïdal et n le nombre de noeuds à l'entrée du réseau, définissant une fenêtre du signal — la fenêtre est définie par un vecteur (portion de signal) de dimension n. Afin que le réseau puisse mémoriser tous les décalages de phase du signal périodique, il faut au moins qu'il apprenne sur p^2 échantillons . Cette analyse est valable pour un noeud d'entrée. Alors que se passe-t-il lorsque qu'il y a plusieurs noeuds dans cette couche? Ceci peut être modifié, dépendamment du nombre de noeuds dans la couche d'entrée du réseau de neurones. L'exemple suivant illustre bien cette dépendance.

Exemple

Soit p le nombre d'échantillons dans une période d'un signal sinusoïdal, n le nombre de noeuds à l'entrée du réseau de neurones, et f_e la fréquence d'échantillonnage, une fenêtre est définie comme étant un vecteur de dimension n. Pour introduire maintenant un décalage de phase dans le signal périodique, on a utilisé une fenêtre qui englobe plus qu'une période de signal c'est-à-dire que n est supérieur à p, pour éventuellement introduire un décalage artificiel de phase. L'exemple numérique suivant va illustrer cette idée.

Il est à remarquer que le choix des paramètres du signal a été effectué de façon à accomoder la visibilité du signal, d'où le choix d'une fréquence de signal de 1200 Hz, d'une fréquence d'échantillonnage de 16 Khz (base de données: BDSONS3) et d'un nombre de noeuds à l'entrée du réseau égal à 19.

Les valeurs numériques des paramètres du signal sont:

p = 16000/1200 =13.33 (on a choisi 13), n = 19 et f_e = 16 kHz.

Alors, essayons d'analyser le contenu et les configurations de chaque entrée du réseau de neurones (tableau 2.1):

Dans la colonne qui s'intitule "contenu du vecteur", les chiffres 13,6 peuvent s'expliquer comme étant les 13 points du signal c'est-à-dire une période, suivie des 6 premiers points de la période suivante. Ceci constitue la configuration d'une fenêtre. La prochaine rangée est composée des 7 points qui restaient de la période précédente, suivie des 12 premiers points de la période suivante, ce qui constitue une deuxième configuration différente, et ainsi de suite jusqu'à la caractérisation de toutes les configurations.

Chaque fenêtre est composée d'une période et demie du signal.

Remarque:

En suivant l'évolution du premier noeud de la couche d'entrée en fonction de ce qu'il verra comme entrée, nous constatons que le réseau voit passer tous les éléments de la séquence, un à un. Ceci implique que tous les décalages sont représentés (p^2). Or, si cela est vrai pour le premier noeud de la couche d'entrée, ça l'est aussi pour n'importe quel autre noeud de cette couche.

Donc le réseau verra tous les décalages si nous faisons l'apprentissage sur n.p échantillons, avec n étant supérieur à p. Nous sommes conscients du fait qu'il y aurait redondance dans l'information apprise, car l'information utile pour le réseau est contenue dans les p^2 premiers échantillons.

Etat du vecteur #	contenu du vecteur	numéro du vecteur
n = p + p / 2	13,6	01
	7,12	02
	1,13,5	03
	8,11	04
	2, 13 , 4	05
	9, 10	06
	3, 13 , 3	07
	10, 9	08
	4, 13, 2	09
	11, 8	10
	5, 13, 1	11
	12, 7	12
	6, 13	13
État initial	13, 6	14

Tableau 2.1 Configuration de séquences présentées à l'entrée d'un réseau de neurones.

2.1.1.1.4 Choix du nombre de noeuds dans la couche d'entrée

Si le nombre de noeuds à l'entrée est supérieur au nombre d'échantillons par période, un choix judicieux de ce dernier serait alors n = 1,5 p. Ce choix a été expérimenté à l'occasion d'une simulation, et les résultats obtenus confirment la relation précédente. L'expérience dans sa totalité sera traitée dans une section ultérieure.

De plus, il faut aussi noter que nous avons vérifié sur la même structure, le cas où le nombre d'échantillons par période (p) est supérieur au nombre de noeuds dans la couche d'entrée (n), en particulier, lorsque n < 1,5 p, nous avons constaté que les résultats sont légèrement bruités.

2.1.1.1.5 Choix du nombre de noeuds dans la couche cachée

Pour choisir ce paramètre, nous nous sommes d'abord référés aux travaux de Nielsson [66], d'Eric Baum [6] et de Panos Antsaklis [77], qui affirment que si on a k patrons (vecteurs de données à l'entrée), on aura alors besoin d'au moins (k-1) hyperplans pour

séparer les patrons — noeuds dans la couche cachée. Ce constat est lié au type de données —statiques— traitées dans les travaux en question. Cependant, notre travail utilise des données dynamiques (ayant des paramètres variables dans le temps dans le cas de la parole et constants dans le cas d'un signal périodique), nous avons donc modifié le nombre d'éléments dans la couche cachée en fonction de nos besoins.

Les simulations nous ont montré qu'il est toujours possible de réduire le nombre de noeuds dans la couche cachée dépendamment de la complexité des données utilisées.

2.1.1.1.6 Choix du nombre de noeuds dans la couche de sortie

En principe, lorsqu'on souhaite traiter un signal (filtrage) par un réseau de neurones, on prend une dimension des vecteurs à l'entrée et à la sortie qui est la même. De plus, pour faciliter le traitement de ce signal, on doit le fragmenter (diviser) en plusieurs séquences (vecteurs de dimension n). Donc, après le passage de l'information dans le réseau, nous devrions retrouver la même information, sous la même forme et par conséquent sous le même vecteur. La dimension du vecteur à l'entrée du réseau est donc la même qu'à sa sortie.

2.1.1.1.7 Importance de la normalisation sur la mémorisation et la généralisation de l'amplitude

La fonction non-linéaire que l'on trouve à la sortie d'un neurone représente une application, au terme mathématique, de R vers [0,1] ou [-1,1]. Si nous n'avons pas de fonction non-linéaire dans la couche de sortie (ce qui revient à un sommateur à la sortie de chaque unité de cette couche) et si les données (à l'entrée et à la sortie) ne sont pas normalisées, l'algorithme "forcera" un peu plus pour produire des valeurs de poids plus grandes, afin de retrouver le signal à la sortie. Ce constat a été observé lors d'une simulation avec un réseau n'ayant que des unités linéaires. Si, au contraire, on a des fonctions non-linéaires dans cette couche, il faut absolument normaliser les données (à l'entrée et à la

sortie), car il sera impossible de retrouver le signal tel qu'on l'avait à l'entrée (chaque unité de sortie du réseau donnerait une valeur se trouvant dans l'intervalle [0,1]).

Ainsi, l'utilisation de la fonction de transfert non-linéaire (la sigmoïde) nécessite absolument une normalisation des données. Au cas où cette dernière s'impose, il y en aura de deux types: une relative et une absolue. Nous aurons l'occasion de les étudier en détails dans une prochaine section.

2.2 Choix de l'algorithme d'apprentissage

Dans cette section, nous présenterons l'algorithme d'apprentissage choisi pour notre travail. Nous démontrons à l'annexe A avec une analyse et une comparaison aux autres algorithmes, que la rétropropagation n'est qu'une généralisation de l'algorithme de LMS "Least Mean Square". L'annexe A porte principalement sur la formulation mathématique qui régit cette approche (algorithmes - équations) en ce qui a trait à l'apprentissage supervisé seulement.

2.2.1.1 Étude de la convergence de trois algorithmes d'apprentissage basés sur la rétropropagation

Les algorithmes étudiés dans cette expérience sont: la rétropropagation standard, la rétropropagation cumulée et la rétropropagation rapide.

Pour étudier la convergence de ces algorithmes, nous avons établi des conditions communes aux trois algorithmes telles que :

— L'apprentissage se fait sur les mêmes données: signaux sinusoidaux périodiques, de fréquences respectives f_1 (1200 Hz), f_2 (2400 Hz) et d'amplitude A (3000). Les fréquences utilisées sont liées par une relation de linéarité (f_2 est le double de f_1). Ce choix est quelconque; et la visibilité de l'information en constitue le principal critère.

— L'architecture utilisée est similaire (fixe) et seul l'algorithme d'apprentissage change (trois algorithmes). Elle est composée de 3 couches dont une cachée. Chacune de ces dernières contient respectivement 19, 10 et 19 noeuds. La fonction sigmoïde est

utilisée comme fonction non-linéaire, à la sortie des neurones de la couche cachée et de la couche de sortie.

— Le niveau d'erreur global (convergence) est le même (ϵ = 0.015).

Pour mesurer la convergence d'un algorithme basé sur la rétropropagation, nous proposons deux stratégies que nous décrivons brièvement:

2.2.1.1.1 1ère stratégie:

Le nombre d'itérations peut être fixé à une valeur initiale. Nous analyserons alors le niveau de précision et l'évolution de l'erreur globale. Cette technique a l'avantage de préciser lequel des trois algorithmes mémorisera le premier l'information.

2.2.1.1.2 2ème stratégie:

Le niveau de précision ϵ est fixé au départ. Le nombre d'itérations est contrôlé par la convergence de l'algorithme. Dans le cas où il n'y pas convergence, il peut parfois atteindre des niveaux incontrôlables (selon la capacité du système).

En comparant les deux stratégies, nous pouvons dire qu'elles sont toutes deux réalisables et observables. Cependant, elles ne sont pas toutes contrôlables, si nous tenons compte de tous les paramètres. Ce faisant, nous avons choisi la première stratégie pour étudier la mémorisation du réseau.

Les résultats de l'expérience sont représentés sur les figures 2.2, 2.3 et 2.4.

Discussion:

Nous avons constaté que le fait de réaliser l'apprentissage sur une fréquence basse, en l'occurence 1200 Hz (figure 2.2), favorise la convergence plus rapidement que pour une fréquence plus haute (2400 Hz) (figure 2.3). Pourtant, nous nous attendions au phénomène contraire, car plus la fréquence est élevée, plus l'information est redondante et par conséquent l'algorithme passe plus de temps à rechercher la précision qu'à définir le patron (portion du signal).

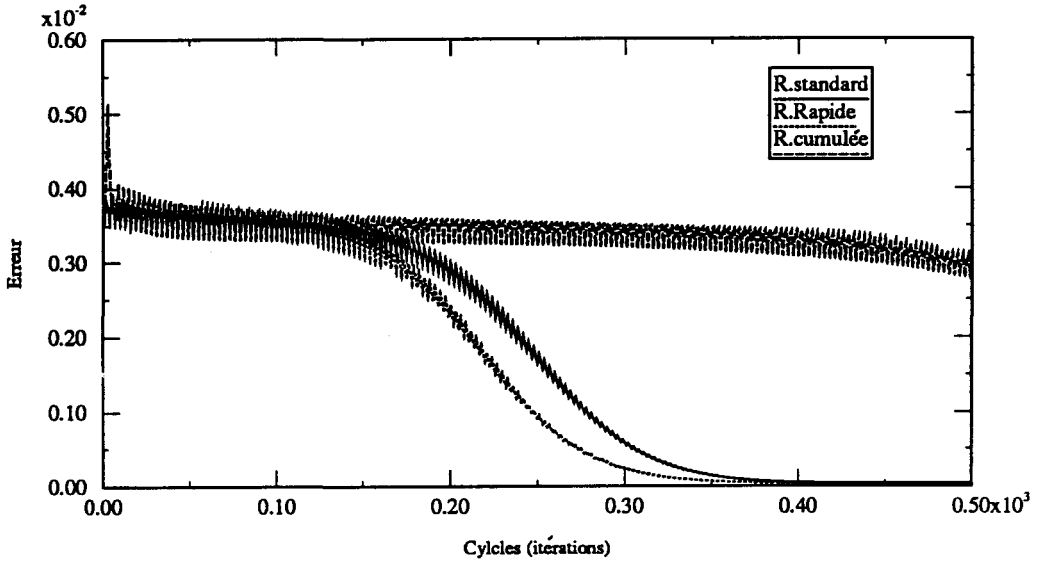

Figure 2.2 Évolution de l'erreur lors de l'apprentissage d'un signal sinusoïdal périodique de fréquence 1200 Hz pour trois algorithmes dérivés de la rétropropagation .

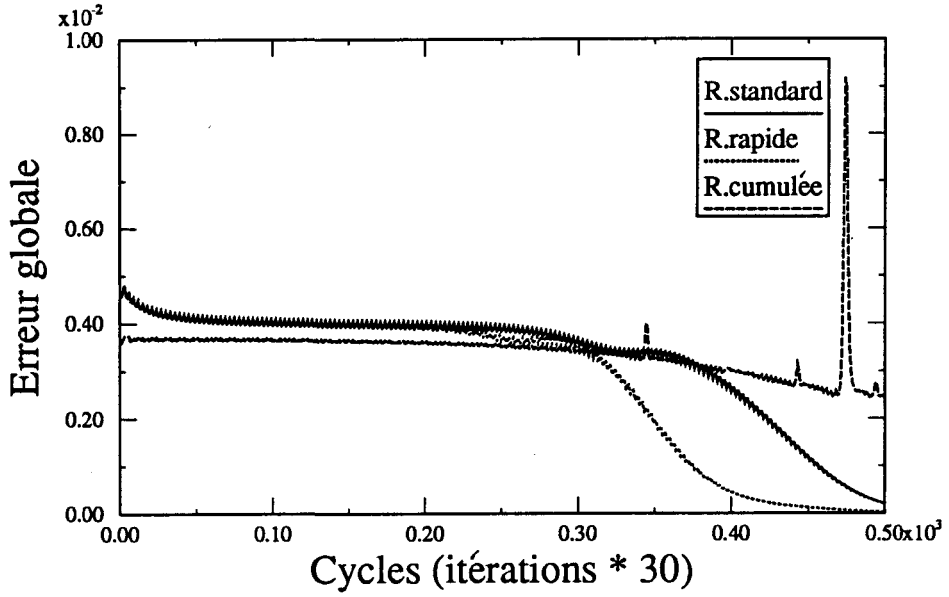

Figure 2.3 Évolution de l'erreur lors de l'apprentissage d'un signal sinusoïdal de fréquence 2400 Hz (normalisation absolue) pour trois algorithmes dérivés de la rétropropagation.

Cependant le réseau semble préférer les patrons qui varient faiblement dans le temps

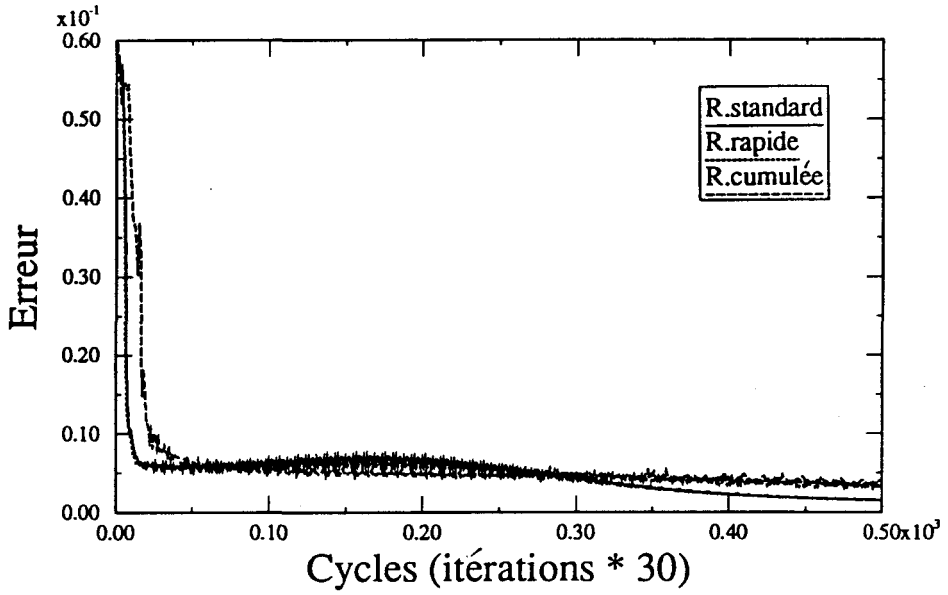

Figure 2.4 Évolution de l'erreur lors de l'apprentissage de deux signaux sinusoïdaux de fréquences respectives 1200 Hz et 2400 Hz, pour trois algorithmes dérivés de la rétropropagation (tester la mémorisation).

(légèrement croissant ou décroissant dans le temps), car une période caractérisée par 13 noeuds (échantillons) est plus facile à apprendre qu'une période caractérisée par 6 noeuds dans une fenêtre de 19 noeuds. On assiste alors, à une duplication de l'information pour le cas de 2400 Hz. Le réseau de neurones prend plus de temps pour caractériser une information de haute fréquence (beaucoup de variation dans le temps) qu'il en prend pour une information, de basse fréquence.

Dans le cas où nous avons plusieurs types de signaux — basses et hautes fréquences, la convergence semble plus drastique aux premières itérations pour les trois algorithmes (figure 2.4), et l'évolution de l'erreur demeure en constante diminution pour la rétropropagation et ses dérivées. Elle demeure néanmoins stable pour la rétropropagation standard et rapide, mais se révèle instable pour la rétropropagation cumulée. L'erreur est plus grande que dans le cas précédent (rapport de 2.5 pour 1).

2.2.1.2 Étude de la non-linéarité d'un réseau de neurones: distorsion d'amplitude et de phase

2.2.1.2.1 Étude de la distorsion d'amplitude et de phase

Le but de cette section est de démontrer la non-linéarité du réseau de neurones, en étudiant la distorsion pouvant être introduite par ce dernier. Normalement, un système linéaire sans distorsion aura une réponse en amplitude constante et une réponse en phase linéaire. Si la réponse en amplitude n'est pas constante, on a une distorsion d'amplitude; si la réponse en phase n'est pas linéaire alors, on a une distorsion de phase.

Étude de la réponse en fréquence du réseau de neurones

Réponse d'amplitude

Pour déduire la fonction de transfert du réseau, nous lui donnons une impulsion à l'entrée et nous analysons la réponse à la sortie. Habituellement, un système linéaire donne la réponse d'amplitude du système telle que représentée par l'équation suivante (2):

soit x(t) le signal d'entrée, h(t) la fonction de transfert du filtre et g(t) la sortie du filtre.

on a:

$$g(t) = x(t) * h(t); \tag{1}$$

ou dans le domaine fréquentiel:

$$G(f) = X(f).H(f); \tag{2}$$

N.b: une convolution dans le domaine temporel est équivalente à un produit dans le domaine fréquentiel.

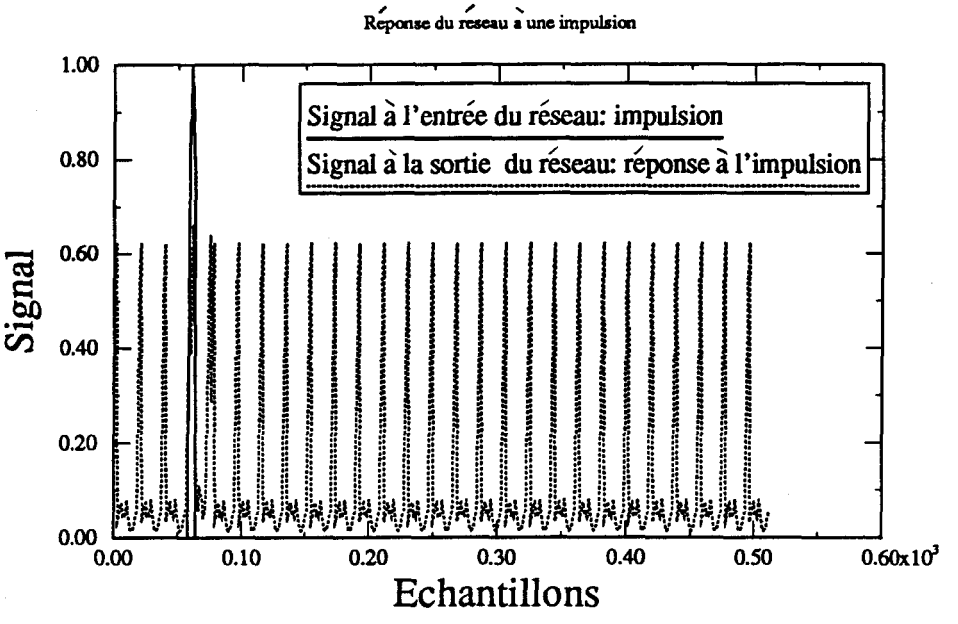

Figure 2.5 Réponse temporelle du réseau à une impulsion.

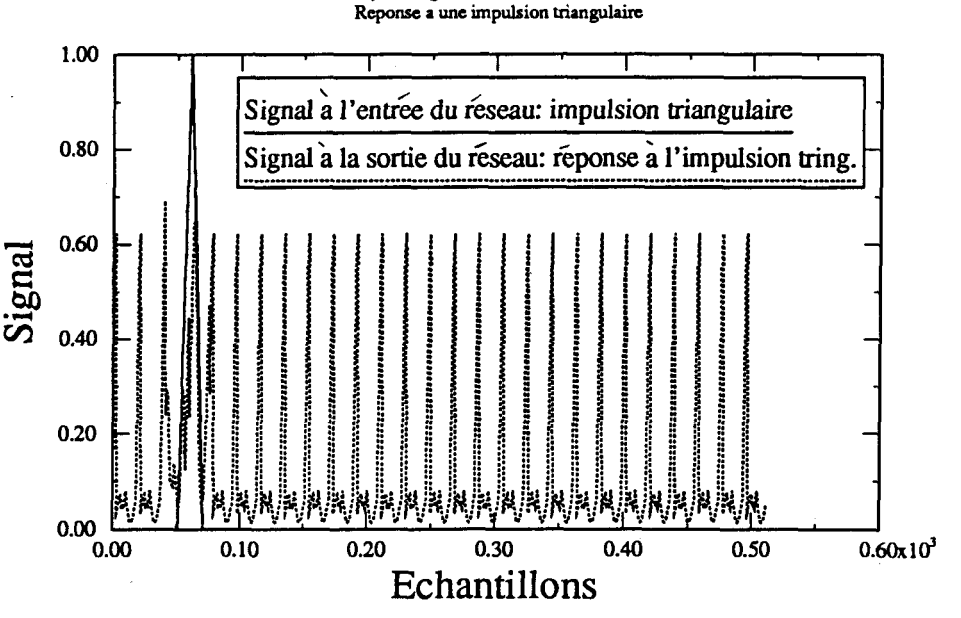

Figure 2.6 Réponse du réseau à une impulsion triangulaire

En principe, la réponse à une impulsion donnera la fonction de transfert h(t) qui caractérise la fonction du filtre. Analysons cette réponse sur le graphique 2.5:

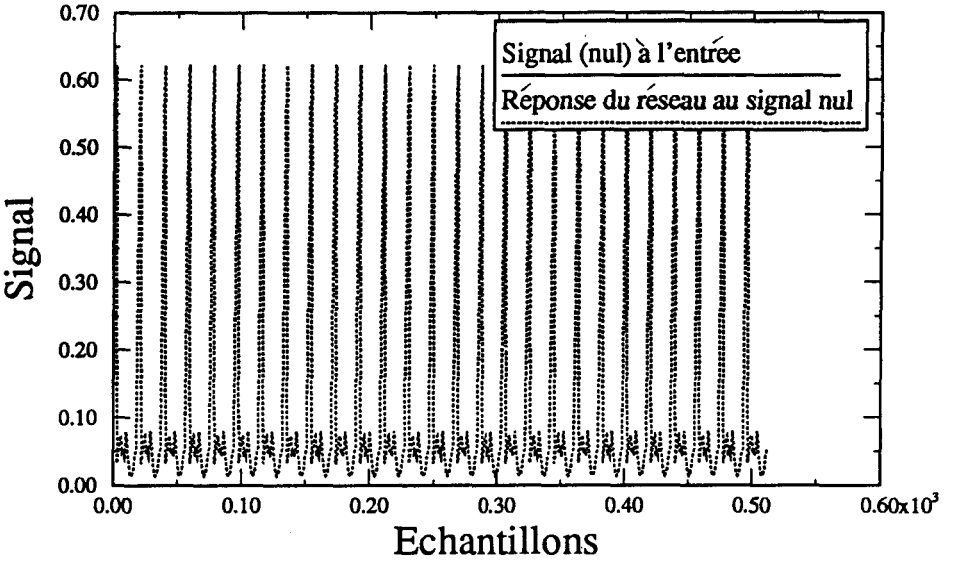

Figure 2.7 Contribution interne du réseau.

Discussion

Nous constatons que la réponse du réseau à une impulsion, est un signal périodique qui caractérise la contribution du biais (seuil) et du noeud dont l'entrée est égale à 1, d'où ressort en même temps l'importance rêvetue par le seuil. Le fait d'introduire une impulsion dans le réseau ne semble pas trop affecter la sortie de celui-ci. La seule chose qu'on a constaté est que le réseau a influencé le niveau d'amplitude de la réponse à la sortie. De même, la contribution des connexions synaptiques de l'impulsion (et celles de la couche de sortie) influence le résultat. La réponse à un signal nul vient confirmer cette remarque (figure 2.7).

Soient y_0, y_1, y_2 et w_b respectivement les sorties des neurones de la couche d'entrée, cachée et de sortie et la connexion synaptique entre le biais et un neurone déterminé.

$$y_{1i} = f\left((w_b * 1) + \sum_{j=1}^{n_{in}} (w_{1j} * 0) + w_{1k} \right);\ ouf = \frac{1}{(1+e^{-x})}$$

l'indice k représente la position de l'impulsion dans le vecteur d'entrée(1)

y_1 *est la sortie de la couche cachée*

$$y_2 = f\left(\left(\sum_{i=1}^{n_h} w_{2i} * y_{1i} \right) + (w_b * 1) \right); \quad (3)$$

ou

y_2 *est la valeur de sortie de la couche de sortie*

La valeur de sortie de la couche cachée est principalement composée du produit scalaire des valeurs de poids provenant du biais et celles provenant de l'impulsion, qui, à leur tour évaluées par la sigmoïde, deviennent plus faibles. Par contre, la sortie du réseau est composée principalement des contributions du biais et de celles de l'impulsion; le biais contribue au niveau des noeuds de la couche cachée et de la couche de sortie. Les figures 2.5, 2.6 et 2.7 illustrent l'effet de l'impulsion sur la réponse du réseau. La figure 2.5 montre que l'influence de l'impulsion est limitée dans la fenêtre en question, entraîne très peu de changement dans ce qui a succédé ou précédé. De plus, nous ne remarquerons pas de variation majeure au niveau de l'amplitude. Par contre, la figure 2.6 illustre que l'impulsion triangulaire a généré une sortie dans le même genre que précédemment excepté que l'influence ne se manifeste que sur les fenêtres précédant et succédant immédiatement la fenêtre contenant l'impulsion, aussi semble-il l'amplitude a subit une variation (légère augmentation). La figure 2.7 illustre la réponse du réseau à un signal nul, constat observé auparavant sur les figures 2.5 et 2.6. La réponse à une impulsion n'est pas une sinusoïde amortie (enveloppe exponentielle) — domaine temporel —, mais plutôt un signal périodique quelconque, nous en déduisons alors, que le système n'est pas linéaire.

2.3 Notion de filtre: définition et caractéristiques

2.3.1 Définition

Étant donné un phénomène dont le spectre s'étend sur un intervalle de fréquences quelconque, un filtre est un dispositif susceptible de privilégier les composantes du signal dont les fréquences se situent dans certains domaines, tout en affaiblissant les autres. La plupart des filtres utilisés sont linéaires. Nous rappelons que le fonctionnement d'un système linéaire est tel que la réponse obtenue par la superposition d'un certain nombre de signaux est égale à la somme des réponses qu'il obtiendrait de chacun de ces signaux.

2.3.2 Utilisation des filtres

Les cas où il est souhaitable de filtrer un signal sont nombreux. Nous pouvons les classer en deux catégories: le filtrage du bruit et la mise en forme des signaux. Le signal qui véhicule l'information est fréquemment distordu, c'est à dire déformé. Cette distorsion peut prendre naissance à la prise du signal (émission), ou pendant la transmission. Elle peut être dûe au bruit thermique (mouvement désordonné des électrons dans les résistances et autres composantes électroniques; ces mouvements sont d'autant plus intenses que la température est plus élevée, d'où l'adjectif thermique). Elle peut encore être due aux performances insuffisantes des lignes de transmissions ou du récepteur. Lorsque cette distorsion présente un caractère aléatoire, on dit que le message est entaché de bruit. Le bruit est un signal parasite dont le spectre peut être étendu, par exemple du côté des hautes fréquences. Dans ce cas, on sera donc fréquemment amené à filtrer le signal. À cet effet, on utilisera un filtre adéquat pertinent; ce filtre présentera une atténuation négligeable pour les basses fréquences (l'essentiel de l'information véhiculée par le signal étant supposé se trouver dans cette région du spectre). Par contre, il atténuera fortement les fréquences élevées, supposées véhiculer peu d'information.

Par mise en forme, nous entendons toutes les modifications que l'on peut faire subir au spectre du signal. Contrairement au cas précédent, on pourra souhaiter relever les

hautes fréquences (c'est-à-dire augmenter leur niveau relatif). Certaines applications exigent encore l'accentuation de zones du spectre. Bien entendu, toute modification du spectre du signal entraîne automatiquement un changement dans son évolution au cours du temps.

2.4 Reformulation du problème

Maintenant que les réseaux de neurones formels semblent s'auto-organiser autour d'exemples appris, serait-il possible de mémoriser un signal périodique, de le reproduire indéfiniment, puis de réaliser le filtrage du bruit sans pour autant affecter le contenu spectral du signal ? Cette question fera l'objet du présent chapitre.

La méthodologie utilisée dans notre travail consiste à nous familiariser avec la théorie des réseaux de neurones, en étudiant la distribution des poids dans le réseau, la mémorisation ainsi que la généralisation du réseau en fonction du nombre de noeuds utilisés dans la couche cachée, la fonction non-linéaire utilisée dans les noeuds (couche cachée et couche de sortie) et l'algorithme d'apprentissage.

Nous sommes conscients du manque de documentation sur le champ du filtrage avec l'approche neurale, mais la simulation nous sera très utile pour l'analyse des performances de cette approche.

Dans les paragraphes suivants, nous allons analyser les principaux paramètres qui gouvernent cette approche.

2.5 Analyse des résultats portant sur la normalisation des données

Pour étudier l'effet de la mémorisation et la généralisation, nous avons effectué une séries d'expériences résumées dans ce qui suit.

Données de l'expérience:

— données d'apprentissage: signal sinusoïdal d'une fréquence de 1200 Hz et d'une amplitude de 1000;

— fréquence d'échantillonnage est de 16 khz;

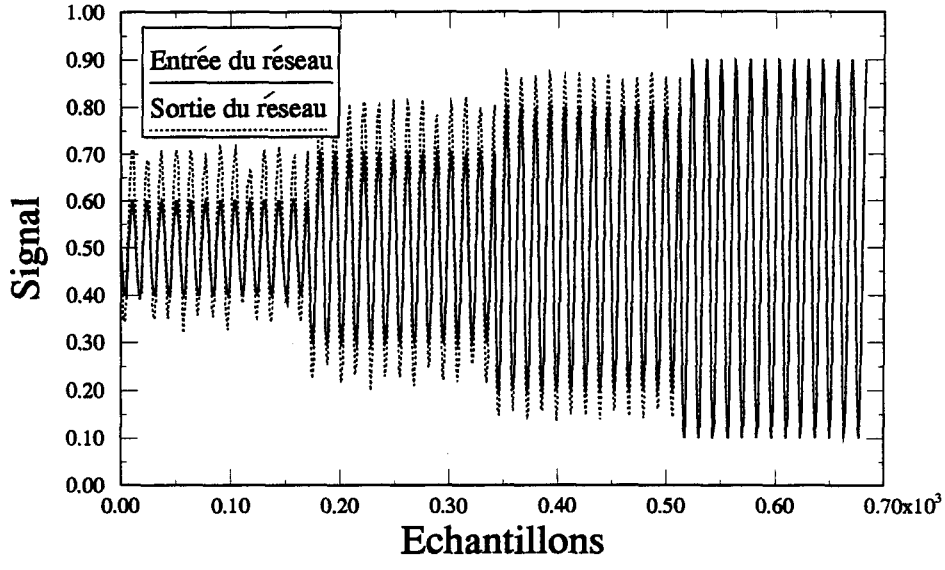

Figure 2.8 Illustration de l'effet de la normalisation relative: superposition de l'entrée et de la sortie du réseau de neurones — fichier n'ayant pas été présenté dans l'apprentissage.

— réseau à propagation-avant;

— nombre de noeuds dans les couches d'entrée et de sortie est de 19 noeuds, alors qu'il est de 10 dans la couche cachée;

— algorithme d'apprentissage: rétropropagation rapide;

— normalisation relative c-à-d. faite par rapport à deux valeurs extrêmes du fichier.

Nous remarquons que le réseau génère une erreur variant de plus de 15% pour les premières portions du signal. Par contre, elle est négligeable pour la dernière portion, en raison de la normalisation relative.

Nous constatons que l'erreur obtenue par la normalisation absolue est faible par rapport à celle que l'on a obtenue par la normalisation relative. Elle se situe autour de 2 %(figures 2.10 et 2.11).

Cette expérience illustre bien l'effet de la normalisation (absolue et relative) sur la mémorisation d'une part et sur la généralisation d'autre part.

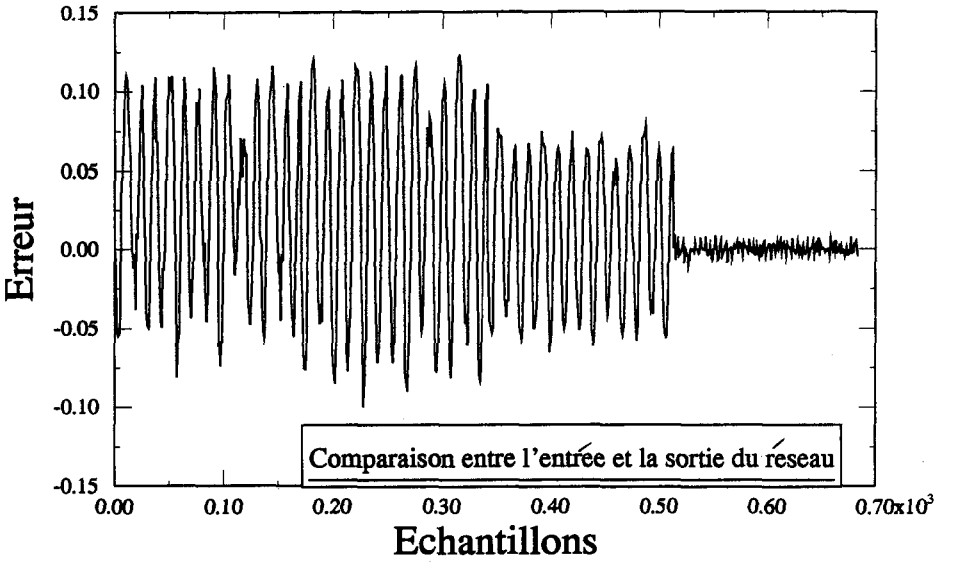

Figure 2.9 Les performances de cette approche sont mises en évidence lorsqu'on calcule l'erreur entre le signal à l'entrée et celui à la sortie du réseau de neurones (normalisation relative).

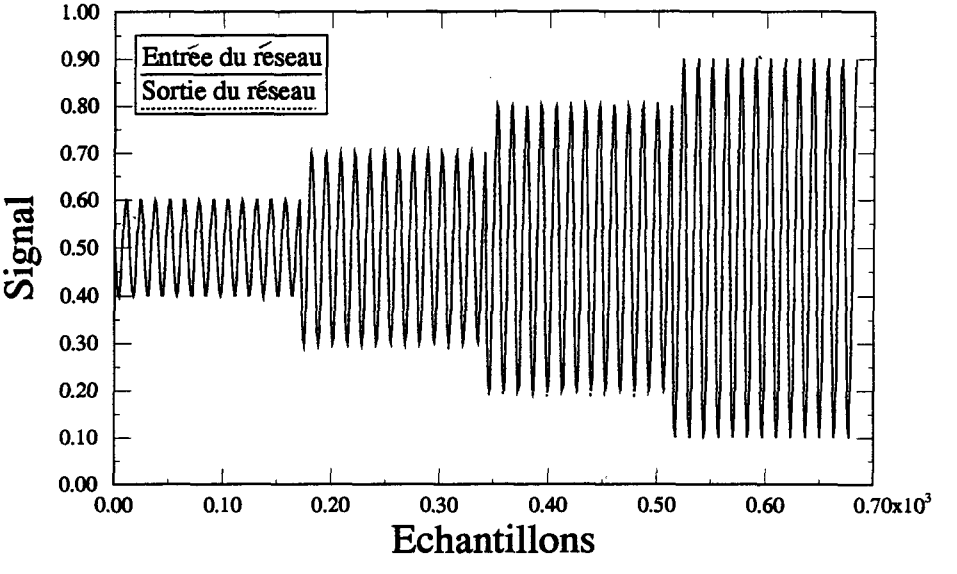

Figure 2.10 Illustration de l'effet de la normalisation absolue: superposition de l'entrée et de la sortie du réseau de neurones — fichier du signal non appris.

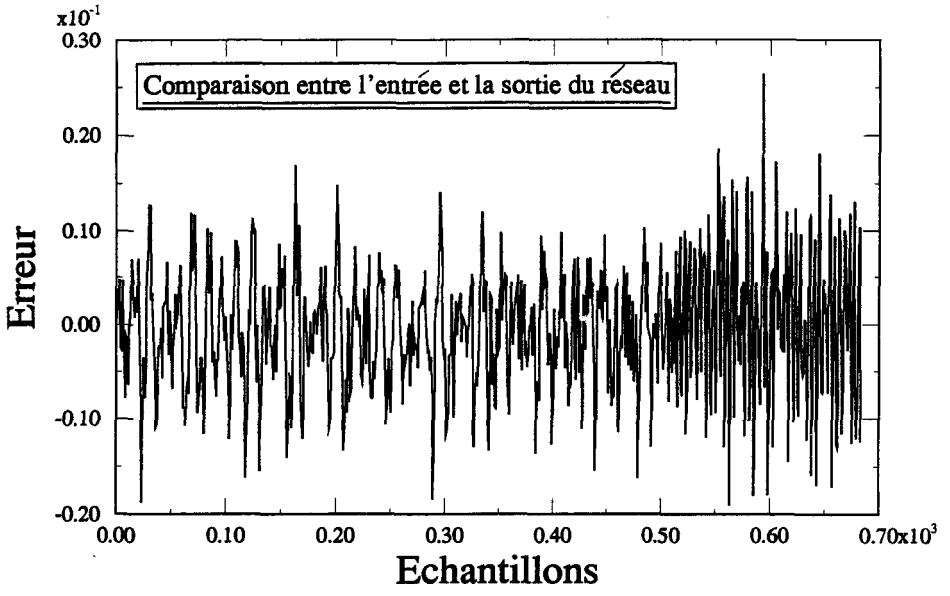

Figure 2.11 Les performances de cette approche sont mises en évidence lorsqu'on calcule l'erreur entre le signal à l'entrée et celui à la sortie du réseau de neurones (normalisation absolue).

2.5.1 Normalisation: analyse de l'expérimentation

Pour réaliser cette expérience, nous avons pris un réseau de neurones multi-couches de type propagation-avant, ayant 49 neurones regroupés en 3 couches. Chacune des couches d'entrée et de sortie contient 19 noeuds; la couche cachée contient 10 noeuds. Le biais (seuil) complète le nombre du 49 ^{ième} noeud. Il est à remarquer que ce noeud particulier, activé toujours à un, possède des connexions avec les couches cachées et la couche de sortie seulement. Ce réseau est entraîné avec l'algorithme de la rétropropagation rapide [22]. Le signal d'apprentissage est composé d'un signal sinusoïdal périodique, d'une fréquence de 1200 Hz et d'une amplitude normalisée A ;

Dans un premier temps, nous avons effectué une normalisation relative c'est-à-dire que nous avons normalisé par rapport aux valeurs extrêmes qui se trouvent dans le fichier d'apprentissage: le maximum et le minimum.

Le résultat obtenu montre que le fichier appris est regénéré sans aucune difficulté. Par contre, dès que le niveau d'amplitude change, soit en accroissant ou en diminuant par

rapport à celui appris, le signal regénéré contient des erreurs qui dépassent 15% en valeur normalisée. Une remarque mérite d'être faite: le signal ayant le plus grand niveau est regénéré sans aucune difficulté. La figure 2.9 (figure 2.8) illustre clairement ce phénomène.

L'analyse des résultats donne à penser que le signal d'amplitude élevé est regénéré sans aucune difficulté, car le niveau d'amplitude est similaire à celui déjà appris après normalisation. Par contre les niveaux moins élevés sont faiblement représentés par le réseau. De ce fait, une quantité importante d'information est perdue lors du traitement.

Si nous examinons de près l'équation de normalisation, nous avons ce qui suit :

$$y(x) = a.x + b;\ \ avec\ a = \frac{(0.9 - 0.1)}{(maximum - minimum)}\ \ et\ \ b = 0.1 - a\ .\ minimum. \quad (4)$$

Exemple d'application:

Si le maximum est de 40 et le minimum est de - 40

alors a = 0.8 / 80 = 0.01 et b = 0.1- 0.01*(- 40) = 0.5

d'où pour une valeur de x_1 = 40 —> y_1 = 0.01 * 40 + 0.5 = 0.9 = maximum permis

Or, plus le dénominateur est grand, plus la valeur de **a** tend vers une petite valeur et plus la normalisation tiendrait compte de **b** (par conséquent de petites valeurs). Cette normalisation (absolue) nous permet d'incorporer plus d'informations, en termes d'amplitude, dans un réseau de neurones. Une conséquence de ce qui précède serait non seulement une convergence lente de l'algorithme, mais aussi une plus grande capacité de mémorisation.

Si on souhaite mémoriser une seule amplitude d'un signal périodique, on doit opter pour une normalisation relative. La précision de cette méthode dépend directement des valeurs se trouvant dans le fichier d'apprentissage.

Pour conclure ce paragraphe, nous dirons que l'incorporation d'une grande quantité d'information, exige une normalisation par rapport aux grandes valeurs, ce qui donnerait de faibles valeurs de a et de b et, par conséquent, de faibles valeurs normalisées qui tiennent compte de toutes les variations.

Une analyse appropriée de la nature des données nous a révélé qu'un choix des valeurs expérimentales comprises entre +/- 40000 serait intéressant et tiendrait compte de la variation des données que l'on souhaiterait traiter: les données de la parole (pour un convertisseur A/D à 16 bits).

2.6 Étude de la mémorisation d'un réseau de neurones artificiels

Pour étudier ce facteur, nous avons commencé par analyser la mémorisation des patrons statiques tels que la table de vérité des portes logiques (**ET, OU, OU EXCLUSIF**); cette analyse a permis de découvrir la relation qui peut exister entre le nombre de noeuds dans la couche cachée et le nombre de patrons à mémoriser. Ensuite, nous avons transposé ces résultats au niveau des patrons ou des séquences sinusoïdales périodiques (introduction du facteur temps) en y apportant les modifications requises.

L'une des alternatives intéressantes pour analyser un réseau de neurones consiste à faire une analyse microscopique de la distribution des angles que font les vecteurs de données avec les vecteurs des poids. Une série d'expériences a été mise en place et pour plus de détails le lecteur peut se référer à l'annexe B.

Dans ce qui va suivre, nous avons élaboré 4 expériences qui résument les paramètres qui affectent la mémorisation d'un réseau de neurones.

2.6.1 Expérience # 1: importance du choix des paramètres d'un réseau de neurones

Le but de cette expérience consiste à étudier le lien entre le nombre de noeuds à l'entrée du réseau et l'information contenue dans le fichier d'apprentissage (fréquence du signal et fréquence d'échantillonnage).

Données de l'expérience:

— données d'apprentissage: signal sinusoïdal périodique dont les caractéristiques sont décrites comme suit :

  — fréquence du signal de 1200 Hz;
  — amplitude de 3000 ;
  — normalisation absolue (par rapport à 40000);
  — fréquence d'échantillonnage de 16 kHz;
  — 171 d'échantillons;
  — 13.33 échantillons par période;
  — réseau à propagation-avant;
  — 13 noeuds à l'entrée et à la sortie;
  — 10 noeuds dans la couche cachée;
  — rétropropagation rapide comme algorithme d'apprentissage;
  — nombre d'itérations a été fixé à 20 000 i.e 1052 cycles;
  — nombre de noeuds à l'entrée du réseau de neurones égal au nombre d'échantillons par période.

2.6.1.1 Résultats

La simulation nous a permis de voir que le réseau de neurones n'est pas en mesure de regénérer ce signal périodique à sa sortie. Le signal obtenu à la sortie du réseau est très dégradé, tel qu'illustré dans la figure 2.12.

2.6.1.2 Interprétation des résultats

Les résultats obtenus sont très dégradés. Ces dégradations au niveau de l'amplitude et de la fréquence sont causées par le choix des paramètres du réseau de neurones. Le fait de choisir un nombre de noeuds dans la couche d'entrée égal au nombre d'échantillons par période affecte les performances du modèle. Lors de la phase d'apprentissage, le réseau

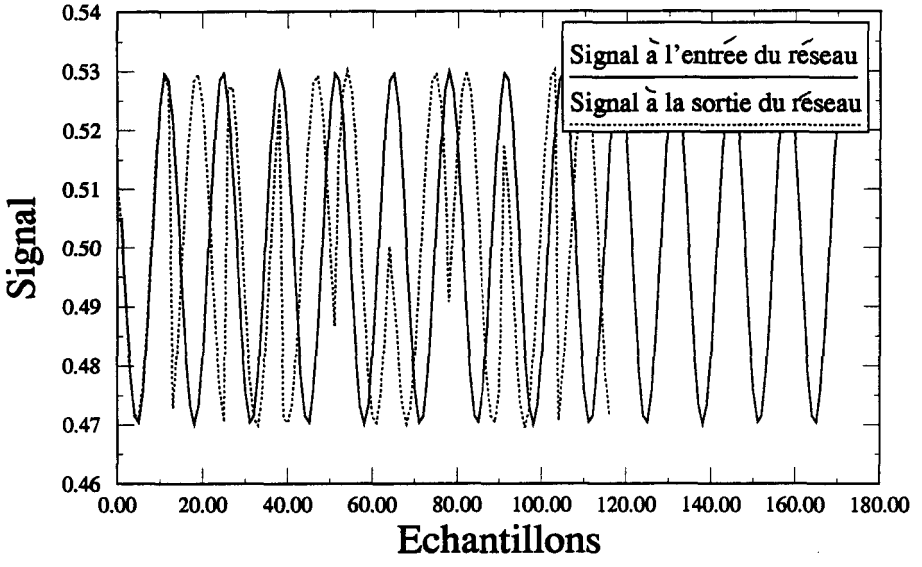

Figure 2.12 Importance du choix des paramètres du R.N: nombre de noeuds à l'entrée (13 noeuds); superposition des signaux à l'entrée et à la sortie du réseau.

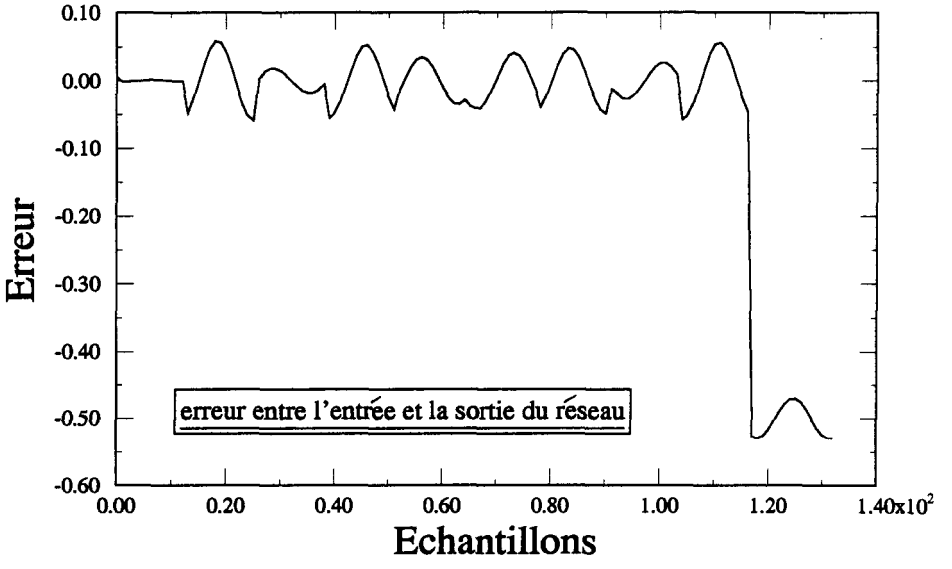

Figure 2.13 Importance du choix des paramètres: erreur entre le signal regénéré et le signal de référence.

voit passer à son entrée le même patron ou la même séquence de données (plus précisement toutes les 3 fois — le nombre d' échantillons par période est de 13.33 pts —). Celui-ci

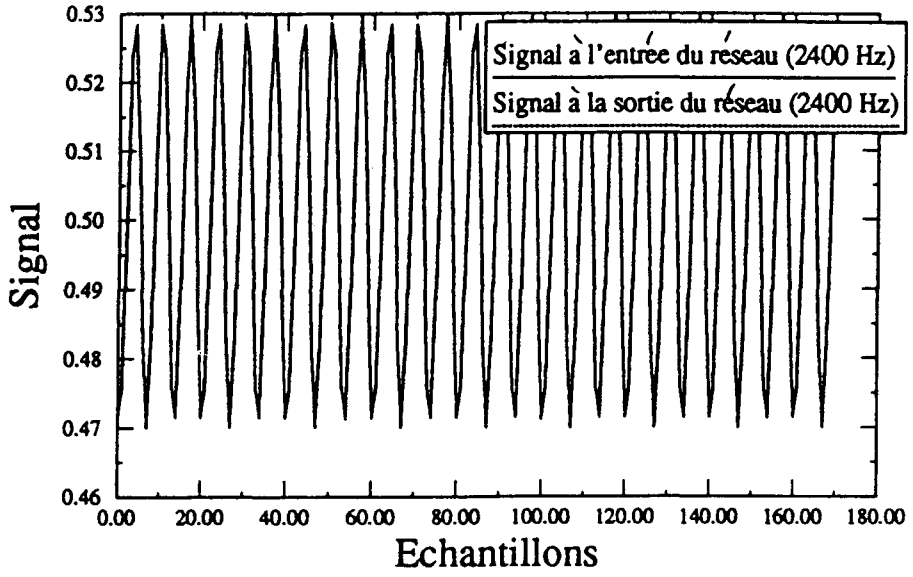

Figure 2.14 Importance du choix des paramétres: nombre de noeuds à l'entrée est largement supérieur à celui des échantillons par période.

va réserver un certain nombre de noeuds cachés pour la mémorisation de ces patrons et le reste des noeuds cachées produira un bruit parasite avec les conséquences que l'on connait.

Ce phénomène peut survenir à chaque fois que la fréquence du signal est un multiple du nombre de noeuds dans la couche d'entrée du réseau de neurones.

Afin de résoudre ce problème et d'éviter la confusion du réseau, il faudra introduire un décalage dans les patrons du signal. Ceci permettrait au réseau de voir plus l'information, sous divers angles pour mieux la mémoriser. Ceci est bien illustré sur les deux derniers graphiques où la fréquence (2400 Hz) est plus grande que la précédente (1200 Hz), ce qui a comme conséquence de réduire le nombre de points par période afin de permettre d'avoir des décalages du signal.

La conclusion déduite de cette expérience, est que le choix des paramètres du réseau est directement lié aux performances du traitement.

2.6.2 Expérience # 2: étude de l'importance du décalage pour la mémorisation d'un signal périodique

Dans cette expérience, notre but consiste à tester l'effet de l'introduction du décalage dans le signal d'entrée et à étudier son influence sur la regénération du signal. Les conditions d'expérimentation sont identiques à celles que nous avons citées dans l'expérience # 1. Le seul paramètre qui a changé, est le nombre de noeuds à l'entrée (19), soit n le nombre de noeuds à l'entrée du réseau et p le nombre d'échantillons par période. Nous avons choisi la relation qui suit :

$$n = \frac{3}{2} \cdot p \; ; \tag{5}$$

2.6.2.1 Résultats de l'expérience

Le réseau est capable de regénérer un signal sinusoïdal sans aucune difficulté. Les résultats obtenus sont illustrés sur le graphique 2.15 (figure 2.16), représentant le signal à l'entrée et à la sortie du réseau, puis le signal d'erreur. Les résultats de cette expérience sont positifs.

2.6.2.2 Interprétation des résultats

Le fait de générer des décalages a aidé le réseau à apprendre les relations intrinsèques au signal et, par conséquent, à les reproduire sans aucune difficulté. Les paramètres du signal (fréquence et amplitude) sont alors regénérés. L'erreur est faible, de 0.15% [erreur = (signal — sortie)/(signal) .100]. Elle est toujours accentuée au niveau du dernier élément de la fenêtre (19 ^{ème} noeud). Nous constatons que la discontinuité est marquée (ce phénomène confirme la réputation du traitement statique des réseaux de type propagation-avant, du moins dans une fenêtre.). Les résultats sont meilleurs que ceux de l'expérience précédente.

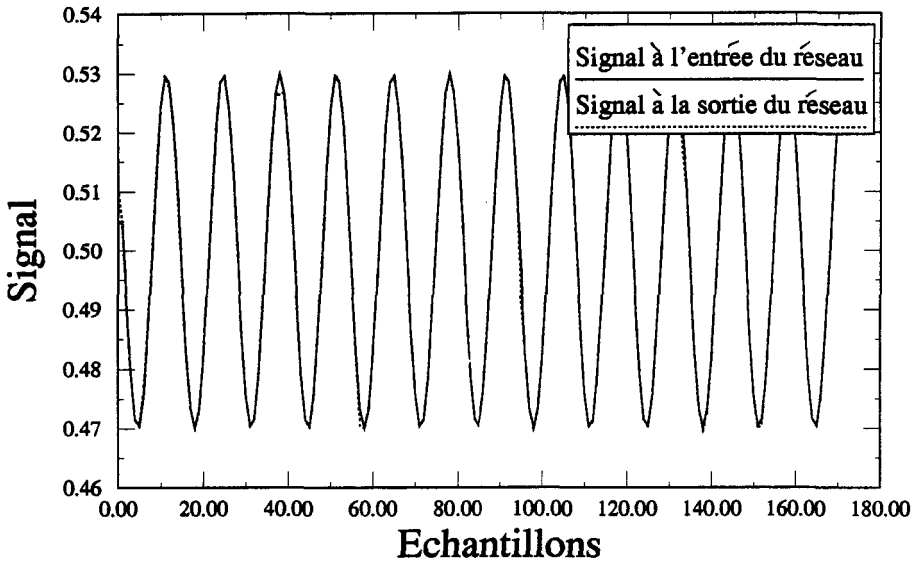

Figure 2.15 Performances du réseau lorsqu'on lui présente le signal d'apprentissage à l'entrée avec divers décalages: superposition du signal d'entrée et celui de la sortie.

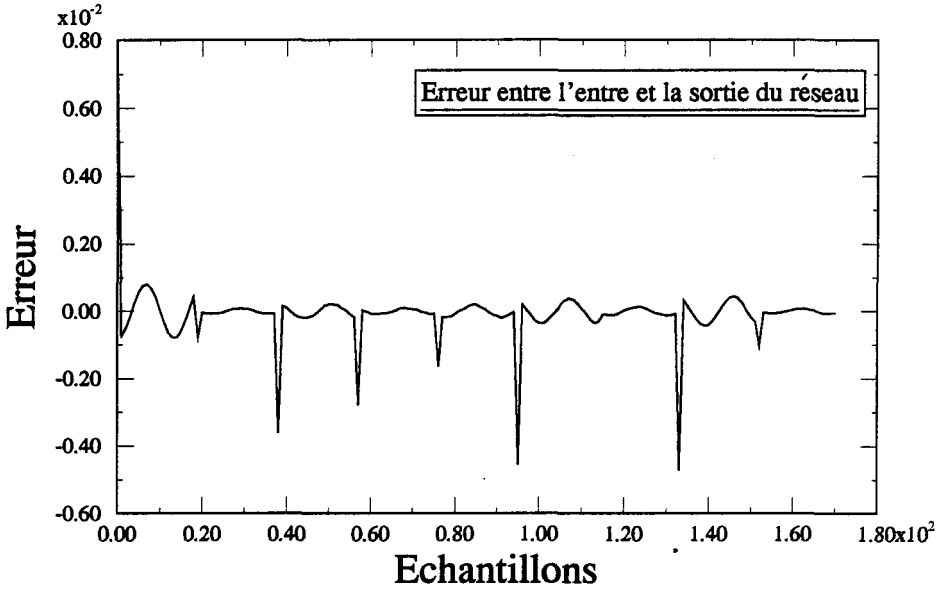

Figure 2.16 Performance du réseau pour la regénération d'un signal périodique (l'erreur est presque nulle).

Au cours de cette expérience, nous avons émis un certain nombre d'hypothèses reliant le nombre de noeuds dans la couche d'entrée au nombre de noeuds par période. Que se

passe-il quand n > p et quand n < p ?

Cas 1: n < p .

Nous avons effectué une simulation qui a révélé que lorsque nous avons un seul signal à mémoriser, il n'y a aucune difficulté au niveau de la regénération. Par contre, lorsque nous avons plusieurs signaux à mémoriser et à regénérer simultanément, ceci est plus difficile; les signaux regénérés sont légèrement bruités. Ce constat pourrait éventuellement être corrigé par une augmentation du nombre d'échantillons dans le signal à apprendre. Ceci est d'autant remarquable que la fréquence est basse. Par exemple, un signal sinusoïdal d'une fréquence de 50 Hz, échantillonné à 16 kHz, donnerait une période étalée sur 323 échantillons. Or, pour que le réseau puisse apprendre tous les décalages, il faut qu'il apprenne sur 323^2 échantillons, ce qui est énorme. Donc, deux stratégies sont possibles : soit l'augmentation du nombre d'itérations, soit celle du nombre de noeuds dans le réseau (ce paramètre est directement lié à l'augmentation de la fréquence d'échantillonnage).

Cas 2: n > p.

Dans ce cas, le nombre de noeuds à l'entrée est supérieur à celui du nombre d'échantillons par période. Une fenêtre peut contenir au moins un multiple (supérieur à 1) de la période. Dans ce cas, le temps de convergence est très réduit. La redondance est bénéfique et permet une bonne mémorisation (voir l'exemple fourni dans la section qui s'intitule "Choix des noeuds dans l'architecture du réseau de neurones", l'exemple qui suit illustre ces notions).

En nous basant sur l'exemple précédent et en suivant l'évolution de ce que verrait le premier noeud dans la couche d'entrée, nous constatons qu'il voit passer tous les éléments de la séquence du signal les uns après les autres; ce qui implique que tous les décalages sont présentés. Si ceci est vrai pour le premier noeud, il en est de même pour n'importe quel noeud de cette couche.

Normalement, le réseau verra toutes les formes du signal si et seulement si l'apprentissage est effectué sur p^2 échantillons. Cependant, le produit n.p serait une bonne

approximation de ce critère. Si n est supérieur à p, alors il y aura redondance d'information lors de l'apprentissage; par contre, si p est inférieur à n, il y aura oubli d'information; ce dernier cas peut survenir lorsque la fréquence est basse.

Nous concluons qu'il est souhaitable de faire l'apprentissage sur p^2 ou bien sur p.n échantillons, lorsque n est supérieur à p.

2.6.3 Expérience # 3: étude de la mémorisation du réseau de neurones vis-à-vis des basses fréquences

Les conditions expérimentales sont identiques à celles de l'expérience # 1 exception faite de la fréquence et de l'amplitude du signal. La fréquence est de 50 Hz au lieu de 1200 Hz et l'amplitude de 10000 au lieu de 3000.

Données de l'expérience:

— 19 noeuds dans la couche d'entrée;

— 10 noeuds dans la couche cachée;

— algorithme d'apprentissage: la rétropropagation rapide;

— 323 échantillons dans le fichier d'apprentissage, soit une période complète du signal.

2.6.3.1 Résultats de l'expérience

Cette expérience a démontré que lorsque nous faisons l'apprentissage sur un seul signal périodique, le réseau mémorise les propriétés de ce signal qu'il regénère aisément (voir la convergence de l'algorithme dans la figure 2.19), bien que le nombre de noeuds dans la couche d'entrée soit nettement inférieur au nombre d'échantillons par période. Les résultats sont représentés sur le graphique 2.17.

2.6.3.2 Interprétation des résultats

Le signal présenté lors de l'apprentissage est regénéré presque intégralement, ce qui vient confirmer les remarques faites lors de l'expérience précédente. Nous remarquons que lorsque n est inférieur à p, le réseau est en mesure de regénérer le signal. La fréquence

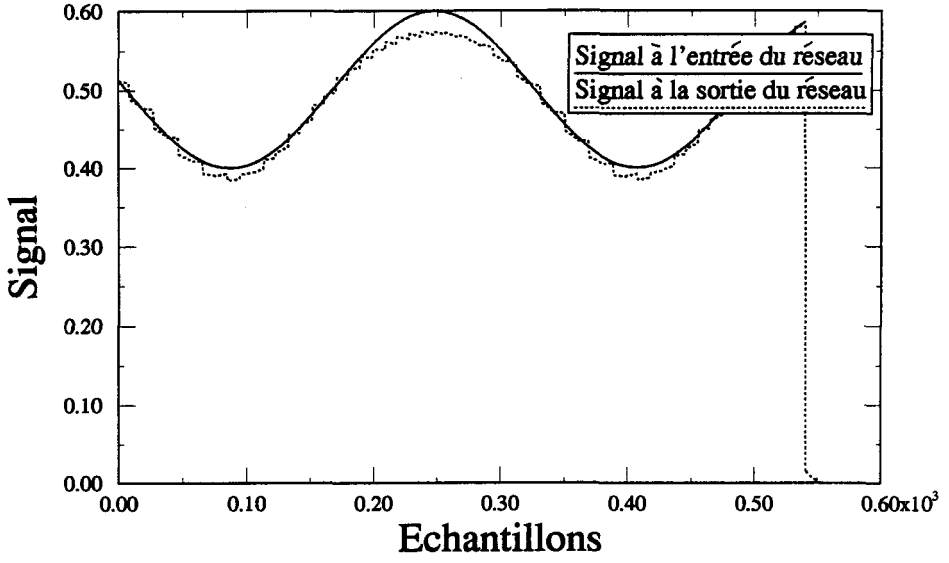

Figure 2.17 Performances du réseau de neurones pour les basses fréquences - 50 Hz - (lorsque n < p); superposition du signal à l'entrée et à la sortie du réseau.

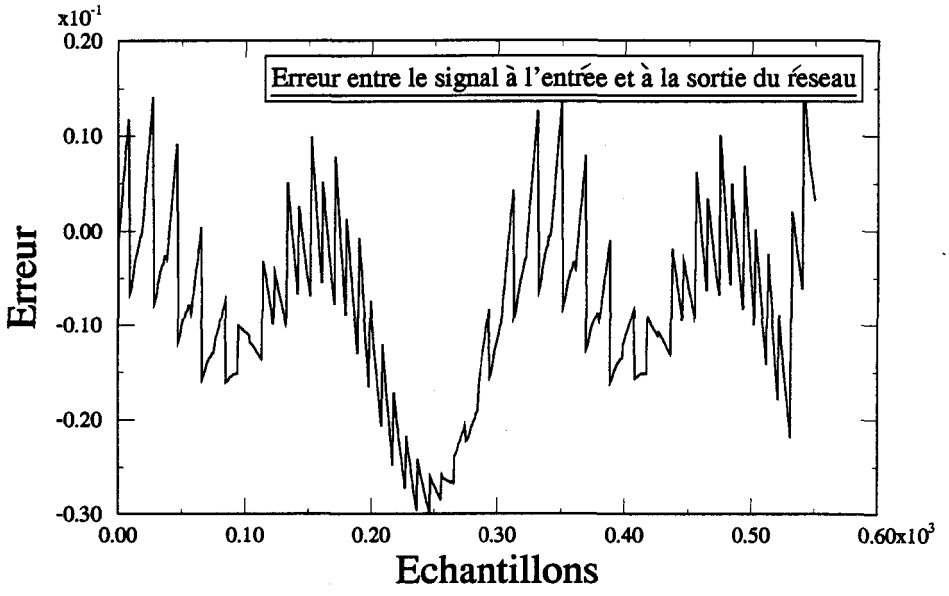

Figure 2.18 Réseau ayant appris sur des basses fréquences: erreur entre le signal à l'entrée et à la sortie du réseau.

est regénérée aisément, mais il y a dégradation au niveau de l'amplitude. Par contre, nous

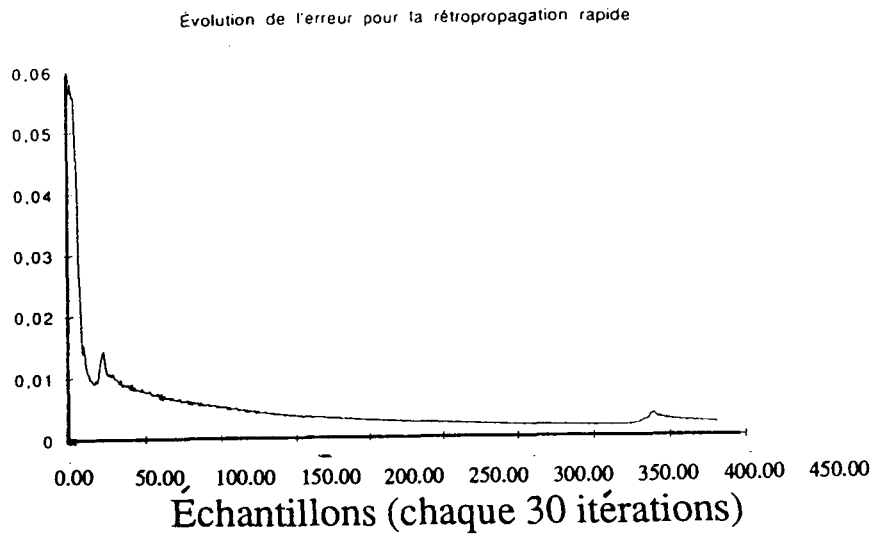

Figure 2.19 Évolution de la convergence de l'algorithme de la rétropropagation rapide vis à vis les basses fréquences.

constatons la présence de marque de discontinuité et de périodicité dans le signal d'erreur, l'erreur étant de 3 % (figure 2.18)

Même si la période est répartie sur un grand nombre d'échantillons, chaque séquence du signal est unique (17 séquences — caractérisant le contenu de toutes les fenêtres) En nous référant au travail d'Eric Baum [6,77], nous aurions besoin d'au moins 17 noeuds dans la couche cachée, alors que dans cette expérience, nous n'avons utilisé que 8 noeuds dans cette couche. L'aspect séquentiel de l'apprentissage s'effectue de telle façon que le choix des patrons à l'entrée du réseau se fait de façon contigüe, ce qui facilite la convergence, aide à la mémorisation et par conséquent à la regénération.

Il est à noter que l'aspect aléatoire de l'apprentissage, c'est-à-dire la présentation des séquences (patrons) de façon non contigüe, ralentirait la convergence et donnerait le même résultat. L'approche aléatoire de la présentation des vecteurs de données entraîne une longue durée d'apprentissage, car chaque vecteur de données est pris de façon aléatoire,

ne possédant pas de relation avec ce qui l'a précédé ni ce qui lui succédera. L'ajustement de poids est alors plus grand (Δw est plus grand, l'erreur étant trop grande). Par contre, l'approche séquentielle engendre un temps d'apprentissage moins long, car chaque vecteur de données est lié au précédent, ce qui constitue un repère pour l'apprentissage; par conséquent, l'ajustement de poids est minime (Δw est plus petit, l'erreur étant petite) à chaque itération.

2.6.4 Expérience # 4: étude de la mémorisation en fonction de l'algorithme d'apprentissage

Dans cette partie, nous étudierons l'influence de l'algorithme d'apprentissage sur la mémorisation d'un signal périodique. Nous comparerons trois algorithmes basés sur la rétropropagation: la rétropropagation standard, la rétropropagation cumulée et la rétropropagation rapide. Les raisons de ce choix seront fournies ultérieurement.

Données de l'expérience:

— fréquence de 1200 et 2400 Hz;
— amplitude de 3000;
— normalisation absolue;
— fréquence d'échantillonnage de 16 kHz;
— 342 échantillons (soit 171 * 2 = 342);
— chaque période contient 13.33 échantillons pour la première fréquence et 6.66 pour la deuxième fréquence;
— réseau à propagation-avant;
— 19 noeuds dans les couches d'entrée et de sortie;
— 8 noeuds dans la couche cachée; le choix de ce paramètre est basé sur le nombre de séquences à apprendre soit 171/19 = 8 séquences ;
— algorithme d'apprentissage: la rétropropagation standard, rapide et cumulée;

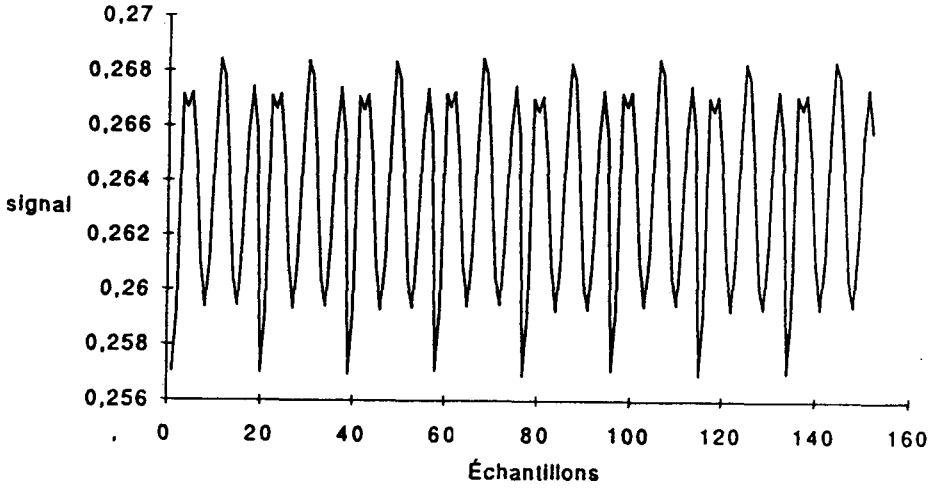

Figure 2.20 Illustration des performances de l'algorithme de la rétropropagation standard en terme de mémorisation pour un signal sinusoïdal de fréquence de 1200 Hz (f_e=16 kHz); pour un nombre d'itérations (fixe) de 15 000.

— nombre de noeuds à l'entrée de réseau de neurones est supérieur à celui des échantillons par période (introduction des décalages artificiels);

— une stratégie consistant à fixer un nombre d'itérations à 15 000 et voir lequel des algorithmes d'apprentissage convergera le plus vite: mémorisation et regénération de l'information;

2.6.4.1 Résultats de l'expérience # 4

Effectivement, cette expérience a confirmé l'importance du type d'apprentissage en fonction de la mémorisation d'un réseau. Les résultats des expériences sont illustrés par les graphiques 2.20 à 2.25. Nous avons jugé préférable d'illustrer la mémorisation des algorithmes d'apprentissages (R. standard, cumulée et rapide) par le tracé des signaux (signal à l'entré et à la sortie du réseau); cependant l'analyse de la convergence de l'erreur est étudié dans la section 2.2.1.2 (figure 2.2 à 2.4)

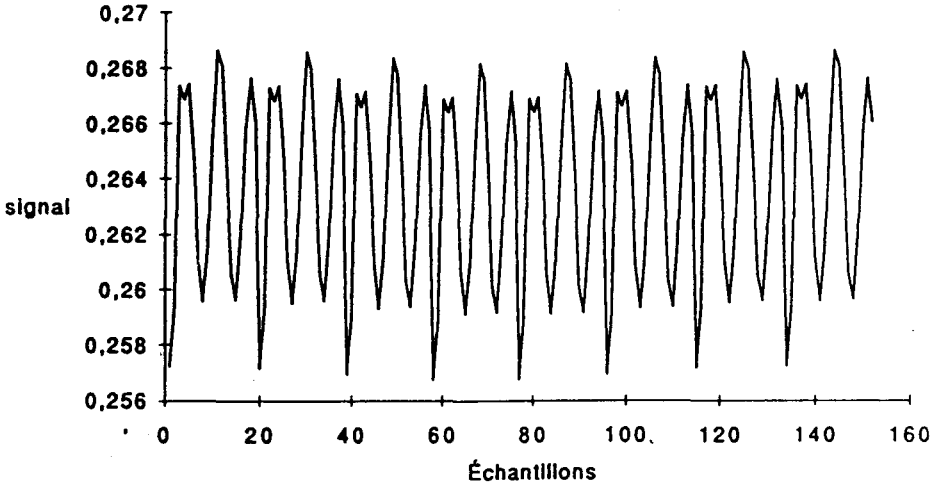

Figure 2.21 Illustration des performances de l'algorithme de la rétropropagation standard en terme de mémorisation pour un signal sinusoïdal de fréquence de 2400 Hz (f_e=16 kHz); pour un nombre d'itérations (fixe) de 15 000.

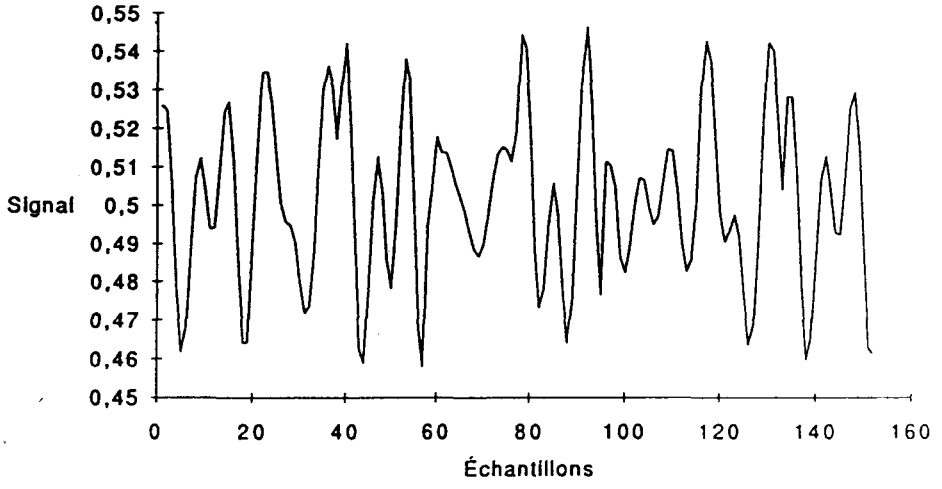

Figure 2.22 Illustration des performances de l'algorithme de la rétropropagation cumulée en terme de mémorisation pour un signal sinusoïdal de fréquence de 1200 Hz (f_e=16 kHz); pour un nombre d'itérations (fixe) de 15 000.

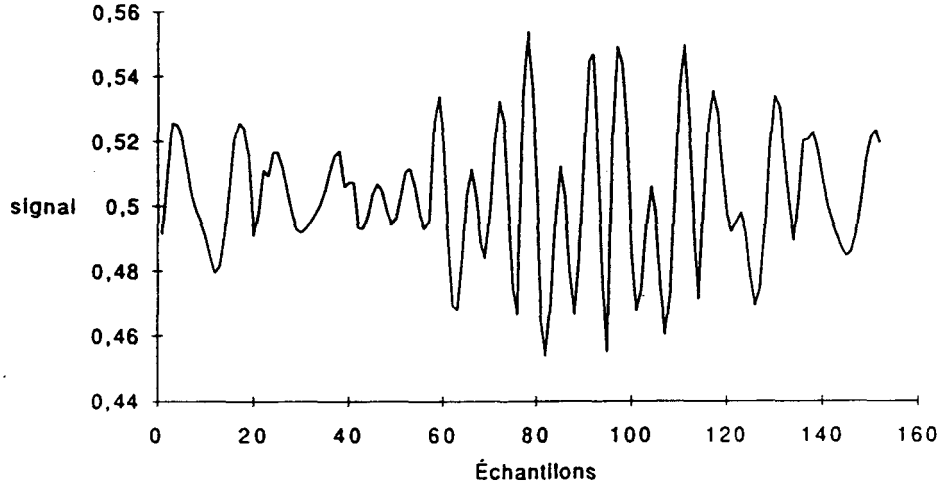

Figure 2.23 Illustration des performances de l'algorithme de la rétropropagation cumulée en terme de mémorisation pour un signal sinusoïdal de fréquence de 2400 Hz (f_e=16 kHz); pour un nombre d'itérations (fixe) de 15 000.

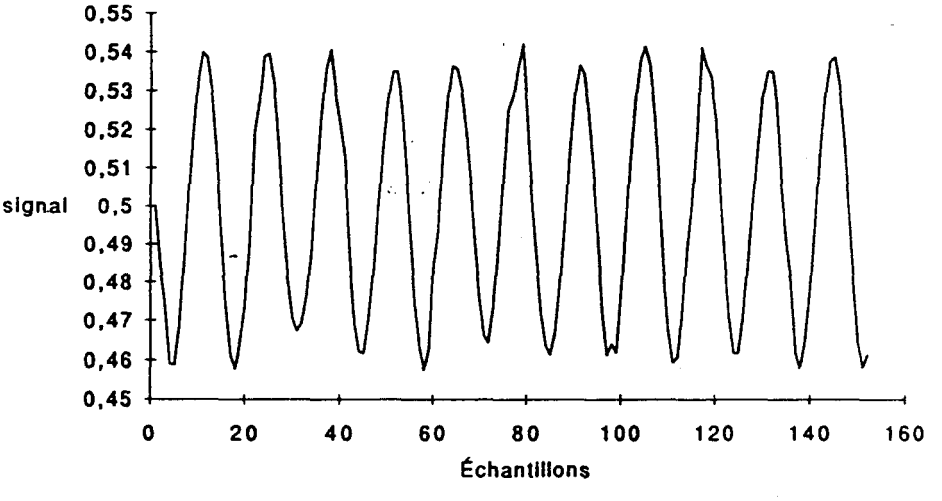

Figure 2.24 Illustration des performances de l'algorithme de la rétropropagation rapide en terme de mémorisation pour un signal sinusoïdal de fréquence de 1200 Hz (f_e=16 kHz); pour un nombre d'itérations (fixe) de 15 000.

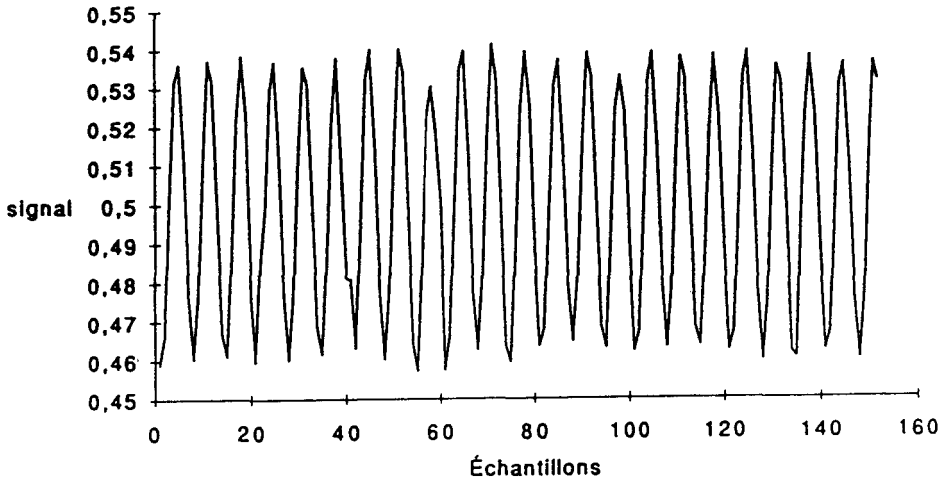

Figure 2.25 Illustration des performances de l'algorithme de la rétropropagation rapide en terme de mémorisation pour un signal sinusoïdal de fréquence de 2400 Hz (f_e=16 kHz); pour un nombre d'itérations (fixe) de 15 000.

2.6.4.2 Interprétation des résultats

Le but de cette expérience est de mettre en évidence l'importance de l'algorithme d'apprentissage sur la mémorisation. Nous avons illustré les performances des algorithmes de la rétropropagation standard (figures 2.20 et 2.21), R. cumulée (2.22 et 2.23) et R. rapide (2.24 et 2.25). L'algorithme de la rétropropagation rapide converge plus rapidement, malgré un faible nombre d'itérations (15000) (figures 2.24 et 2.25). Ce qui n'est pas le cas pour la rétropropagation standard ((figures 2.20 et 2.21)) et est encore pire pour la rétropropagation cumulée.

Enfin, la mémorisation d'un réseau de neurones est liée au choix judicieux de plusieurs paramètres, tels que le nombre de noeuds dans la couche d'entrée et dans la couche cachée, le nombre d'échantillons par période, la relation pouvant exister entre le nombre de noeuds à l'entrée du réseau et celui des échantillons par période, la normalisation des données, l'algorithme d'apprentissage, la convergence et la stabilité de l'apprentissage.

2.7 Étude de la fonction de transfert: fonction linéaire — fonction non-linéaire

Chaque neurone (noeud) de la couche cachée comprend une fonction de transfert, pouvant être linéaire ou non. Cette fonction donne au réseau de neurones sa caractéristique non-linéaire. Pour analyser son influence sur un réseau, nous allons nous pencher sur trois types de fonctions: une fonction linéaire (sommateur), une fonction non-linéaire surjective — chaque élément de l'ensemble de départ possède au moins une image dans l'ensemble d'arrivée — (sinus), et une fonction non-linéaire bijective (la sigmoïde).

Données de l'expérience:

— fréquence de 1200 Hz;

— amplitude de 3000;

— normalisation absolue;

— fréquence d'échantillonnage de 16 kHz;

— 171 échantillons (fichier d'apprentissage);

— chaque période contient 13.33 échantillons;

— réseau à propagation-avant;

— 19 noeuds dans les couches d'entrée et de sortie;

— 8 noeuds dans la couche cachée — le choix de ce paramètre est basé sur le nombre de séquences à apprendre soit (171/19 = 8 séquences) ;

— algorithme d'apprentissage: la rétropropagation rapide;

— nombre de noeuds à l'entrée de réseau de neurones est supérieur à celui des échantillons par période (introduction des décalages artificiels);

— la stratégie consiste à étudier l'effet de la fonction de transfert à la sortie d'un neurone sur la convergence de l'algorithme d'apprentissage.

2.7.1 Étude d'une fonction linéaire: sommateur

Nous avons utilisé un réseau de type propagation-avant, fonctionnant avec l'algorithme de la rétropropagation rapide, ayant 19 noeuds dans les couches d'entrée et de sortie et 8 noeuds dans la couche cachée.

Les résultats obtenus ne sont pas concluants, car l'apprentissage ne converge pas; par conséquent le réseau n'est pas en mesure de mémoriser l'information.

D'un point de vue mathématique, il ressort clairement que la sortie de chaque noeud dans la couche cachée est composée de la somme pondérée des données provenant de la couche d'entrée. La même procédure se répète, mais sur un niveau plus loin. Le fait d'avoir des valeurs plus grandes à la sortie de la couche cachée rend la convergence difficile.

Nous avons effectué une simulation avec une seule séquence de données. Le nombre d'itérations est très élevé (de l'ordre de 20 millions d'itérations). Les résultats de cette expérience révèlent effectivement que le réseau a réussi à mémoriser l'information. Cependant l'erreur demeure élevée. En conclusion, nous pouvons dire que le fait de n'avoir que des unités linéaires dans un réseau de neurones a pour conséquence d'allonger le temps d'apprentissage.

2.7.2 Étude d'une fonction non-linéaire — une sinusoïde: sinus

Les conditions expérimentales sont identiques à ce qui a été avancé précédemment, le seul paramètre ayant changé est la fonction de transfert — non-linéaire.

Les résultats sont meilleurs que ceux de l'expérience précédente: l'algorithme d'apprentissage a tendance à bien converger (évolution de l'erreur est instable (voir figure 2.27)) et par conséquent, le réseau est capable de mémoriser l'information. Au niveau de la généralisation, le principal problème est causé par la périodicité (indésirable) de la fonction sinus. Cette périodicité peut entraîner une confusion dans la discrimination des patrons.

La présente analyse diffère de la précédente en un point très important, au niveau de la deuxième couche (couche cachée): la fonction non-linéaire. Cette fonction, la sinusoïde

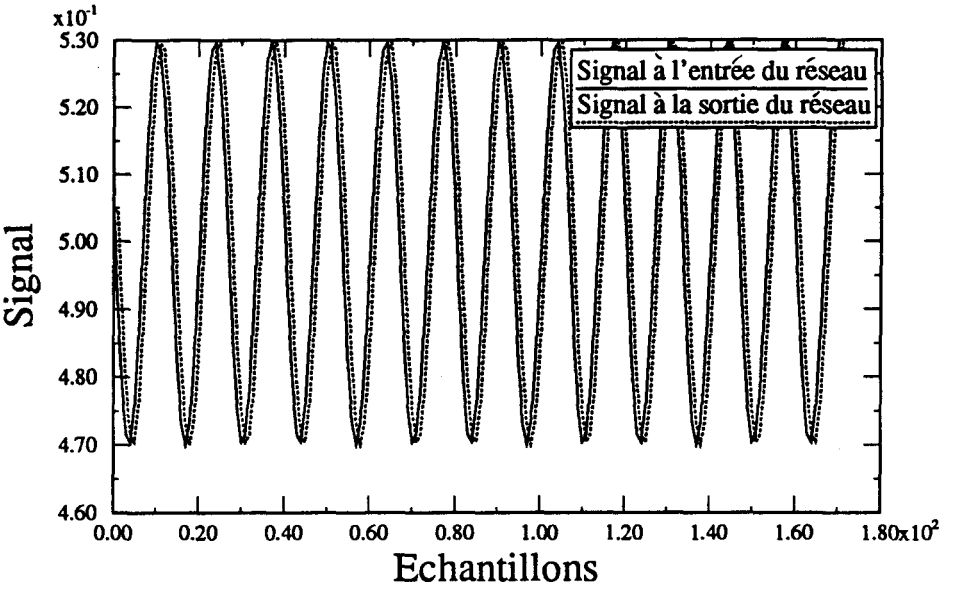

Figure 2.26 Performances du réseau de neurones utilisant des fonctions **sinusoïdales** (sinus) comme fonction de transfert.; superposition du signal à l'entrée et à la sortie du réseau (f=1200 Hz, f_e= 16 kHz).

possède un intervalle d'application de R (ensemble source) vers l'intervalle [-1,1] (ensemble d'arrivée). Donc, elle réalise en quelque sorte une normalisation des données dans cet intervalle. C'est grâce à elle que le réseau est en mesure de mémoriser l'information.

La normalisation réalisée au niveau de la deuxième couche (normalisation non-linéaire) diffère de celle effectuée lors de la phase du pré-traitement [0.1 , 0.9] qui est linéaire. L'ordre de grandeur est différent dans les deux cas. La normalisation du pré-traitement effectue une conversion entre 0.1 et 0.9; la normalisation au niveau de la couche cachée, par contre, effectue une conversion entre –1 et 1. L'écart de l'erreur peut être corrigé avec une augmentation du nombre d'itérations.

En résumé, la périodicité influence directement la discrimination et par conséquent la mémorisation. Une conséquence de ceci se manifeste sur la convergence de l'algorithme d'apprentissage — elle peut semer la confusion — comme nous l'avons illustré dans les graphiques 1.5 (loi normale - sinus), 1.6 (sigmoïde) et 2.27.

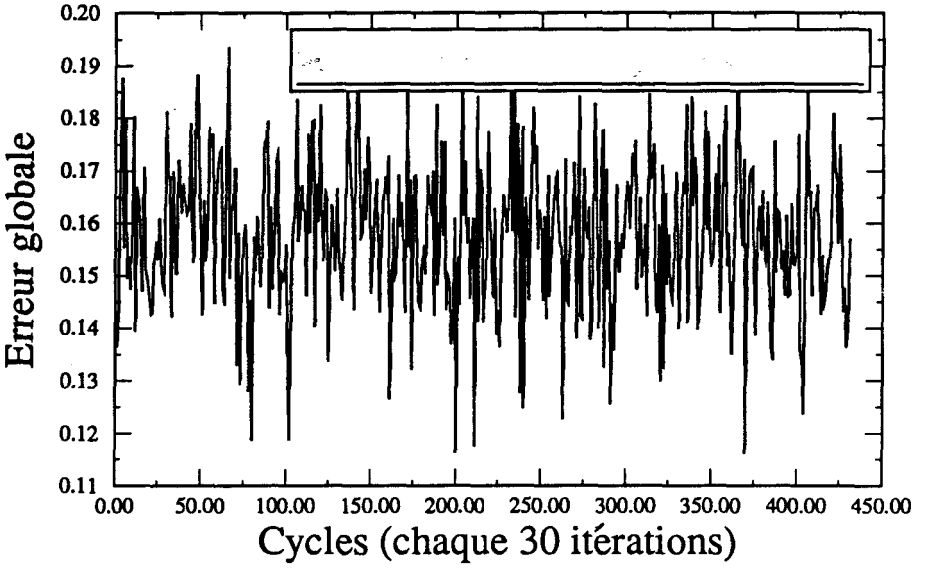

Figure 2.27 Évolution de l'erreur lors de l'apprentissage de la fonction sinus (après 135 000 itérations).

2.7.2.1 Interprétation des résultats

Nous constatons très clairement que lorsqu'on utilise la fonction sinus comme fonction de transfert, l'algorithme d'apprentissage a beaucoup de difficultés à converger. L'évolution de l'erreur est instable comme illustré dans le graphique 2.27.

Dans un réseau, lorsqu'on superpose le signal à l'entrée et celui à la sortie, on observe un léger décalage (figure 2.26), par comparaison à la figure 2.29). L'erreur atteint facilement les 10 %.

L'erreur s'explique en partie par le fait que la fonction sinus n'est pas bijective (n'est pas injective), mais surjective (chaque élément de l'ensemble de départ R n'a pas une image et une seule dans l'ensemble d'arrivée, mais plusieurs (deux)). Ce qui génère une confusion et gêne la convergence de l'algorithme. Chaque entrée du réseau doit avoir une et une seule configuration à la sortie et non pas plusieurs.

2.7.3 Étude d'une fonction non-linéaire — une sigmoïde

Cette fonction est différente des précédentes, car elle est non-linéaire et apériodique. Sa

L'abscisse de la donnée d'une sinusoïde de période T	L'ordonnée de la donnée calculé par un sinus	L'ordonnée de la donnée *calculé par une sigmoïde*
T/8	sinus(T/8)	sigmoïde(T/8)
3.T/8	sinus(3.T/8) = sinus(T/8)	sigmoïde(3.T/8)
0	0	1/2

Tableau 2.2 Comparaison entre deux fonctions de transfert non linéaires: le sinus et la sigmoïde..

forme lui permet de discriminer plusieurs états sans générer une confusion (à chaque élément de l'ensemble de départ est associé un seul élément et un seul de l'ensemble d'arrivée). De fait, cette fonction effectue une normalisation au niveau de la couche cachée, ce qui aide partiellement l'algorithme à converger et par conséquent, le réseau à mémoriser.

La principale force de cette fonction réside dans sa capacité de discriminer plusieurs états. Par exemple, si on prend une sinusoïde de fréquence élevée, alors pour chaque donnée espacée de nT/4, le sinus donnera la même sortie; la sigmoïde par contre donnera une valeur différente comme illustré au tableau 2.2.

La fonction non-linéaire a été étudiée de façon détaillée dans le début de ce chapitre, où nous démontrons l'importance de la sigmoïde dans un réseau de neurones.

En résumé, nous pouvons dire que le choix de la fonction de transfert limite fortement la capacité du réseau de neurones à mémoriser et à regénérer l'information. Il faut tenir compte de deux caractéristiques importantes: la non-linéarité et la non-périodicité (caractère monotone de la fonction).

2.7.3.1 Interprétation des résultats

Les résultats obtenus montrent l'importance du choix de la fonction de transfert. Le choix de la sigmoïde comme fonction de transfert a aidé à stabiliser l'évolution de l'erreur pour mener à terme la convergence (figure 2.29). L'erreur entre le signal de sortie et le signal d'entrée est de 0.005 % (figure 2.30).

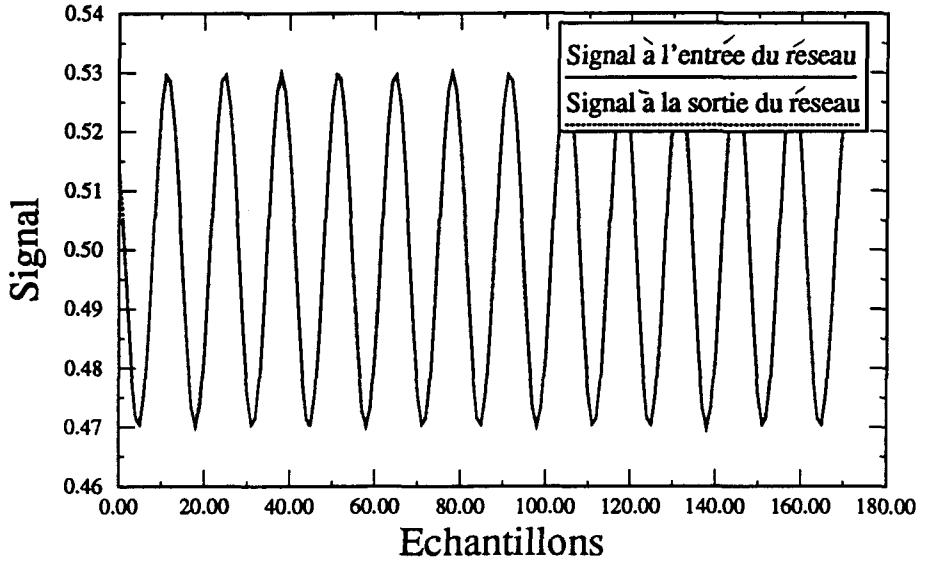

Figure 2.28 Illustration des performances du réseau de neurones utilisant des fonctions hyperboliques (sigmoïde) comme fonction de transfert.

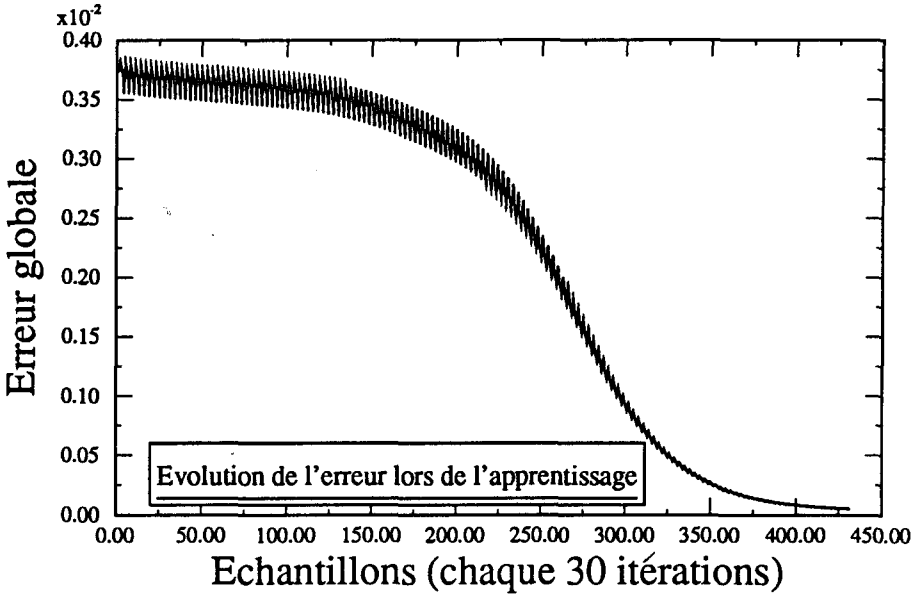

Figure 2.29 Illustration de la convergence du réseau utilisant la sigmoïde comme fonction de transfert (bonne convergence).

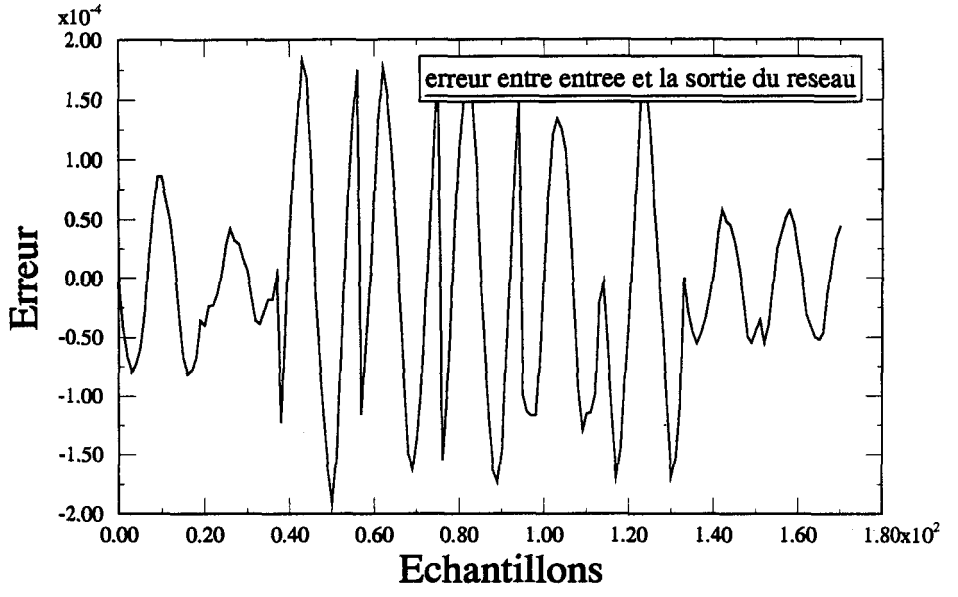

Figure 2.30 Erreur entre le signal à l'entrée et à la sortie du réseau.

2.8 Étude de la généralisation d'un réseau de neurones

À la section précédente, nous avons vu qu'un réseau de neurones est en mesure de mémoriser un signal périodique et de le reproduire à sa sortie. Il réussit donc à apprendre les deux paramètres essentiels dans le signal: l'amplitude et la fréquence . Notre objectif est de savoir si le réseau de neurones est capable de généraliser au niveau de l'amplitude et de la fréquence, d'apprendre sur un signal périodique puis d'interpoler et d'extrapoler en se basant sur l'information apprise. Nous voulons également vérifier les capacités de généralisation d'un réseau de neurones.

2.8.1 Généralisation d'amplitude

Cette partie portera sur l'étude de la généralisation d'amplitude. Nous essayerons de voir quels paramètres influencent cette caractéristique et de découvrir s'il est possible de les contrôler.

Nous débuterons cette étude par le choix des hypothèses et des données de l'expérience. Nous allons effectuer 2 expériences pour étudier la généralisation d'amplitude du réseau de neurones.

2.8.1.1 Expérience #1: limite de la généralisation en amplitude - apprentissage sur un seul niveau d'amplitude

Données de l'expérience:

— données d'apprentissage : signal sinusoïdal périodique de fréquence f et d'amplitude A=3000;
— 171 échantillons;
— fréquence d'échantillonnage de 16 kHz;
— fréquence du signal de 1200 Hz;
— réseau multi-couches à propagation-avant (feed-forward);
— algorithme d'apprentissage: propagation rapide;
— normalisation absolue c'est-à-dire comprise entre +/– 40000;
— nombre d'itérations est de 20000 (1052 cycles).
— la simulation est réalisée sur une station Sparc;

2.8.1.1.2 Résultats

Les résultats de cette simulation sont représentés aux figures 2.31, 2.32 et 2.33. Nous avons superposé l'entrée et la sortie du réseau sur le même graphique et placé l'erreur sur un autre graphique. La réponse du réseau avec des niveaux d'amplitudes variant de 10 à 40000 est représentée sur les graphiques cités ci-dessus. Les résultats obtenus sont positifs.

Il ressort clairement que la fréquence n'est nullement affectée par la variation d'amplitude.

La figure 2.32 montre très bien que l'interpolation se fait sur une fenêtre (un patron ou une séquence du signal), et non sur le signal au complet. Cependant, ce problème disparaît lors de l'augmentation des niveaux d'amplitude. Des expériences connexes ont

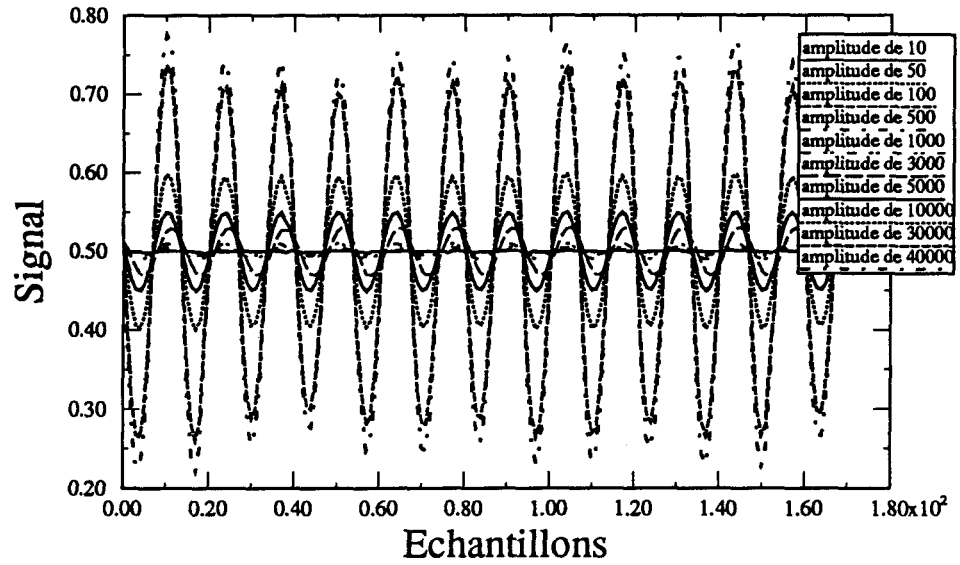

Figure 2.31 Illustration des performances du réseau ayant appris sur un niveau d'amplitude de 3000 et une fréquence de 1200 Hz: regénération de plusieurs niveaux d'amplitude: 10, 50, 100, 500, 1000, 3000, 5000, 10000, 30000 et 40000.

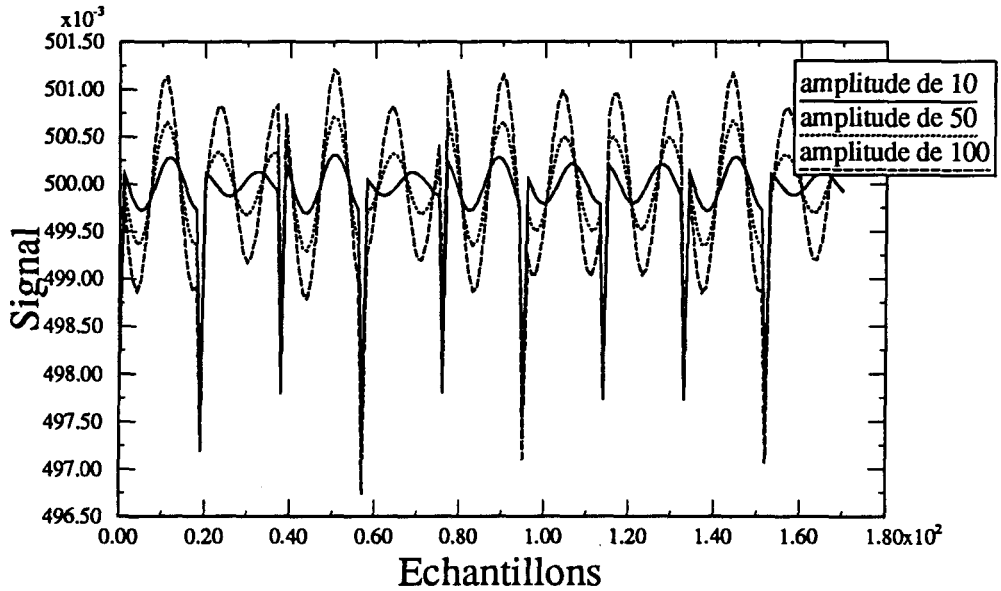

Figure 2.32 Illustration des performances du réseau ayant appris sur un niveau d'amplitude de 3000 et une fréquence de 1200 Hz: regénération de plusieurs niveaux d'amplitude: 10, 50, 100 et 500; apparition des discontinuités lorsque les niveaux sont peu élevés ; illustration de l'effet de décalage des périodes.

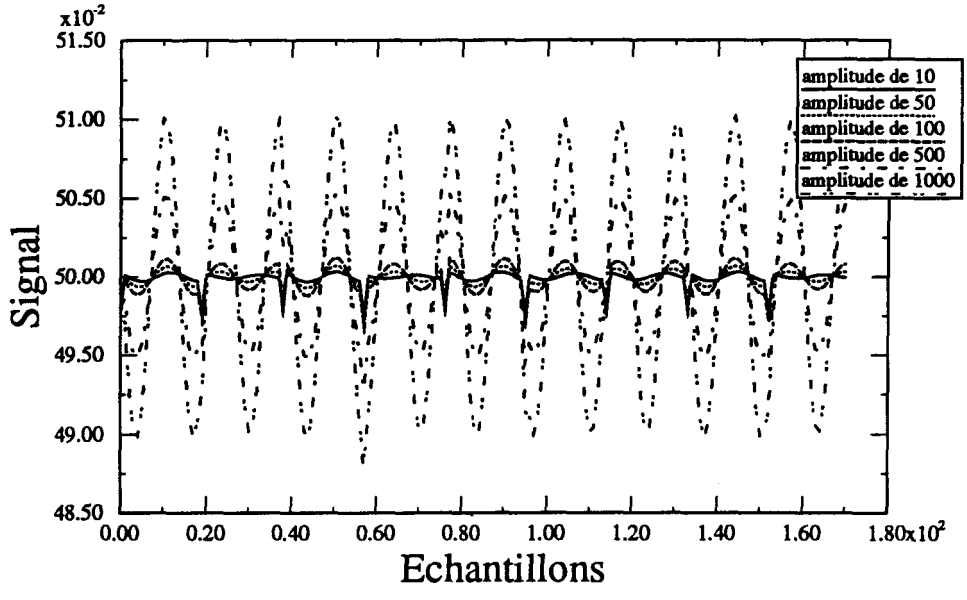

Figure 2.33 Illustration des performances du réseau ayant appris sur un niveau d'amplitude de 3000 et une fréquence de 1200 Hz: regénération de plusieurs niveaux d'amplitude: 10, 50, 100, 500 et 1000

été effectuées sur les fréquences, le même constat est relevé (pour ne pas trop encombrer le manuscrit avec trop de figures, nous avons supprimer les figures de cette partie).

2.8.1.1.3 Interprétation des résultats

Les résultats de cette expérience démontrent que le signal appris est regénéré sans aucun problème. Par contre, l'erreur augmente au fur et à mesure qu'on s'éloigne de l'amplitude apprise. Cette dernière est de 10 % tel qu'illustré dans les graphiques (2.31, 2.32 et 2.33) pour les faibles amplitudes. Nous constatons que lorsque l'amplitude et la fréquence sont basses, la démarcation de la fin de la fenêtre est très visible. Nous remarquons bien les différents décalages. Cependant cette démarcation disparaît au fur et à mesure que l'amplitude augmente.

En général, nous pouvons considérer la généralisation fiable sur une plage de +/- 2000 de l'amplitude apprise.

Les causes d'erreurs peuvent être liées à la normalisation d'une part (lors de la phase de prétraitement, nous normalisons linéairement entre 0.1 et 0.9; cependant la normalisation effectuée par la couche cachée se fait entre 0 et 1 de façon non-linéaire, à cause de la sigmoïde), à l'instabilité de l'évolution de l'erreur et à la convergence de l'algorithme lors de l'apprentissage d'autre part.

2.8.1.2 Expérience # 2: limite de la généralisation d'amplitude - apprentissage sur plusieurs niveaux d'amplitude

Les conditions expérimentales sont identiques à celles de l'expérience précédente. Seulement deux paramètres varient dans celle-ci: le nombre de signaux à apprendre et le niveau d'amplitude dans ces derniers. Les 10 signaux à apprendre ont une fréquence commune mais 10 niveaux d'amplitude différents (10, 50, 100, 500, 1000, 3000, 5000, 10000, 30000 et 40000). Nous avons mené cette expérience en vue de lever le doute sur la généralisation du réseau au niveau de l'amplitude.

2.8.1.2.1 Résultats

Interprétation des résultats

Les résultats obtenus par cette expérience démontrent que le signal appris est bien regénéré. Par contre, on remarque deux constatations importantes:

— l'amélioration de la regénération des signaux ayant un niveau d'amplitude élevé (10000, 30000 et 40000) (figure 2.35).
— la dégradation de la regénération des signaux ayant un niveau d'amplitude faible (10,50,100 et 500) (figure 2.34).

Le fait d'apprendre sur des niveaux d'amplitude variables a favorisé l'apprentissage des grands niveaux aux dépens des bas niveaux. Cette information capitale sera utilisée ultérieurement.

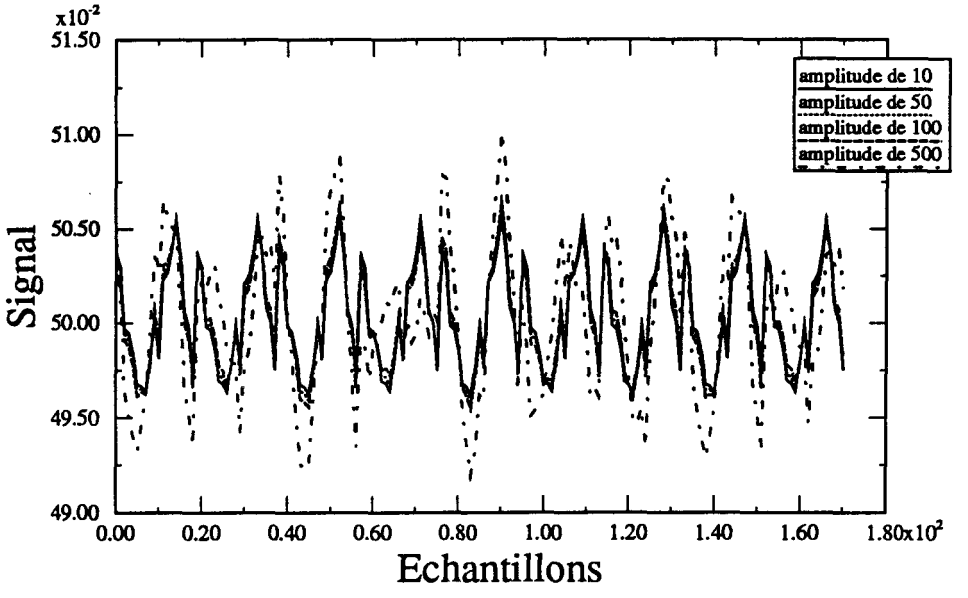

Figure 2.34 Performances du réseau ayant appris sur un niveau d'amplitude variable et une fréquence de 1200 Hz: regénération de plusieurs niveaux d'amplitude: 10, 50, 100 et 500; apparition des discontinuités lorsque les niveaux sont peu élevés; illustration de l'effet du décalage des périodes.

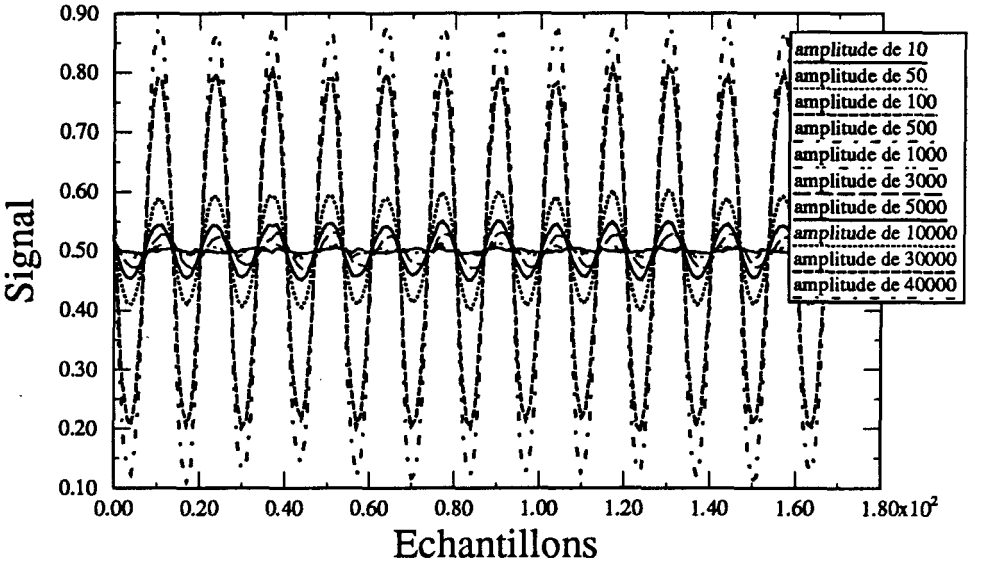

Figure 2.35 Performances du réseau ayant appris sur un niveau d'amplitude variable et une fréquence de 1200 Hz: regénération de plusieurs niveaux d'amplitude: 10, 50, 100, 500, 1000, 3000, 5000, 10000, 30000 et 40000.

2.8.2 Généralisation en fréquence

Dès lors que nous avons vérifié l'aptitude d'un réseau de neurones à généraliser au niveau des amplitudes sur une plage limitée, nous tenterons de refaire l'expérience en tenant compte des fréquences.

2.8.2.1 Expérience # 1: généralisation au niveau des fréquences — apprentissage sur deux fréquences f1 = 1200 Hz et f2 = 2400 Hz

Données de l'expérience:

— données d'apprentissage: 2 signaux sinusoïdaux périodiques de fréquences f1 et f2 et d'amplitude commune A;
— 342 échantillons;
— fréquence d'échantillonnage de 16 kHz;
— réseau à propagation-avant;
— nombre de noeuds à l'entrée et à la sortie du réseau est de 19; celui dans la couche cachée (20) est tributaire du nombre de séquences présentées (342/19 = ~ 20);
— algorithme d'apprentissage: la rétropropagation rapide;
— normalisation absolue (+/- 40000);
— simulations sur une station Sparc 2.

2.8.2.1.1 Résultats

Cette simulation est faite sur deux signaux périodiques de même amplitude mais de fréquence différente (deux fréquences : l'une est le double de l'autre). Elle nous a permis de regénérer avec succès les deux fréquences apprises lors de l'apprentissage. De plus, lorsque nous avons fourni un signal n'ayant jamais été appris, le réseau nous a donné une sortie en fonction des fréquences (phase apprentissage) qu'il avait apprise. Sa qualité de regénération dépend de la proximité de la fréquence apprise (voir les figures 2.36 à 2.51). Dans cette partie, nous avons pris le soin de parcourir une plage de fréquences allant

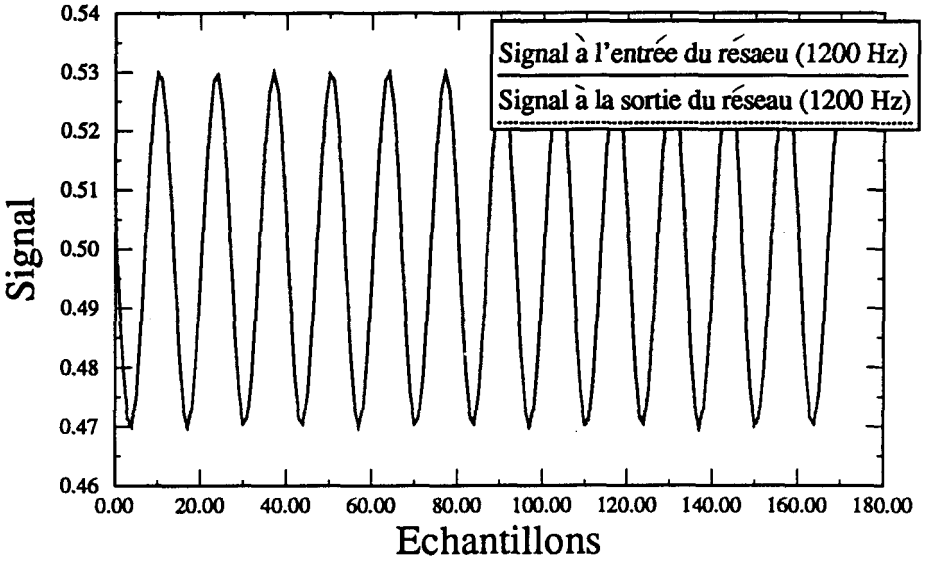

Figure 2.36 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence apprise lors de l'apprentissage (1200 Hz).

de 1200 Hz à 2700 Hz par intervalle de 100 Hz. Nous avons veillé à ne pas avoir de chevauchements de signaux de fréquences différentes dans les fenêtres.

2.8.2.1.2 Interprétation des résultats

Cette expérience démontre bien qu'un réseau ayant appris sur deux signaux périodiques est en mesure de généraliser au niveau des fréquences intermédiaires. Ceci constitue un atout qui peut être exploité dans le champ de la parole.

La précision du signal regénéré est d'autant meilleure que la fréquence du signal de rappel est proche de la fréquence apprise. Cette constatation nous oblige à restreindre la plage de généralisation à celle donnant le meilleur résultat.

Après l'analyse des résultats, nous proposerons une plage de +/- 100 Hz aux alentours de la fréquence apprise. La précision est d'autant meilleure que la fréquence est grande. Ceci est bien illustré sur les figures précédentes (2.36 à 2.51).

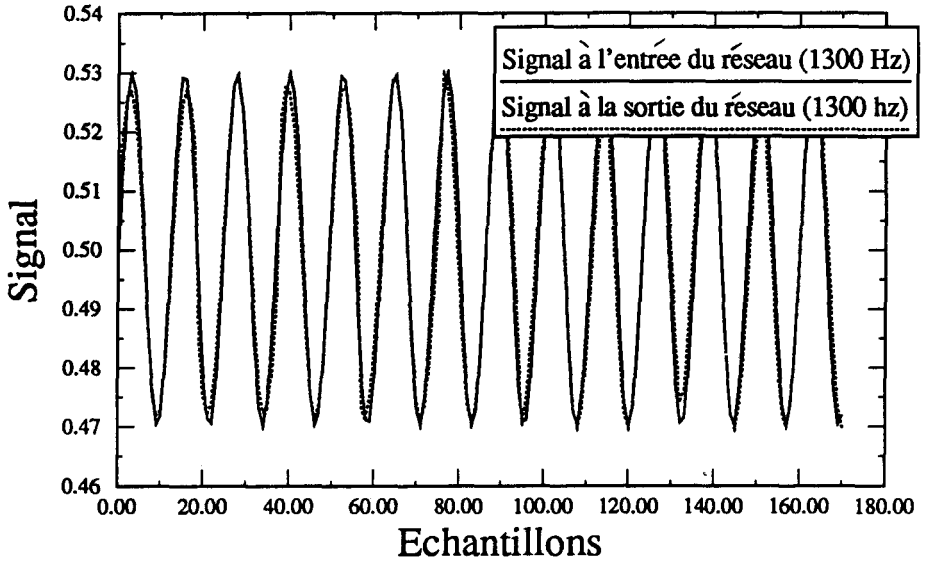

Figure 2.37 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (1300 Hz).

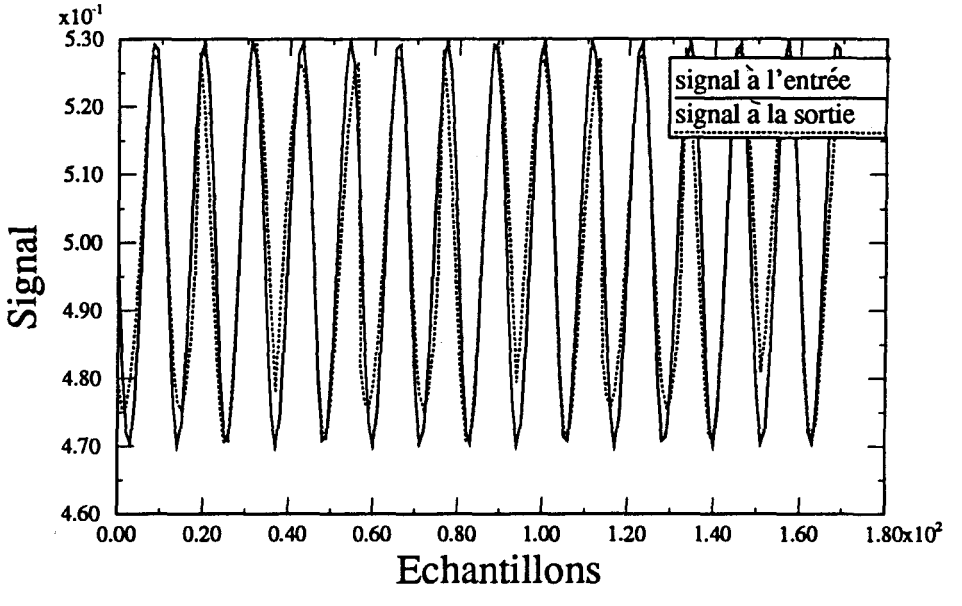

Figure 2.38 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (1400 Hz).

Nous sommes curieux de voir comment le réseau de neurones arrive à généraliser en

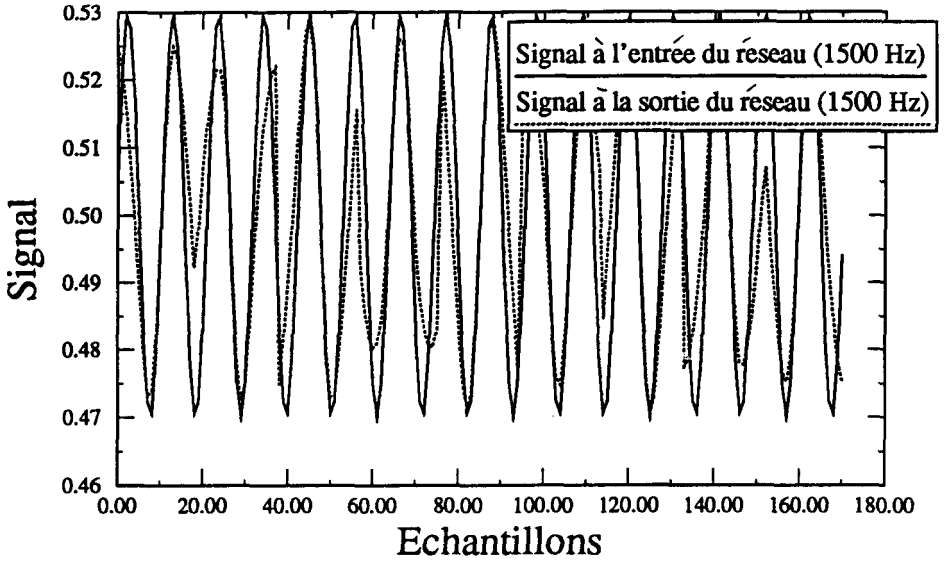

Figure 2.39 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (1500 Hz).

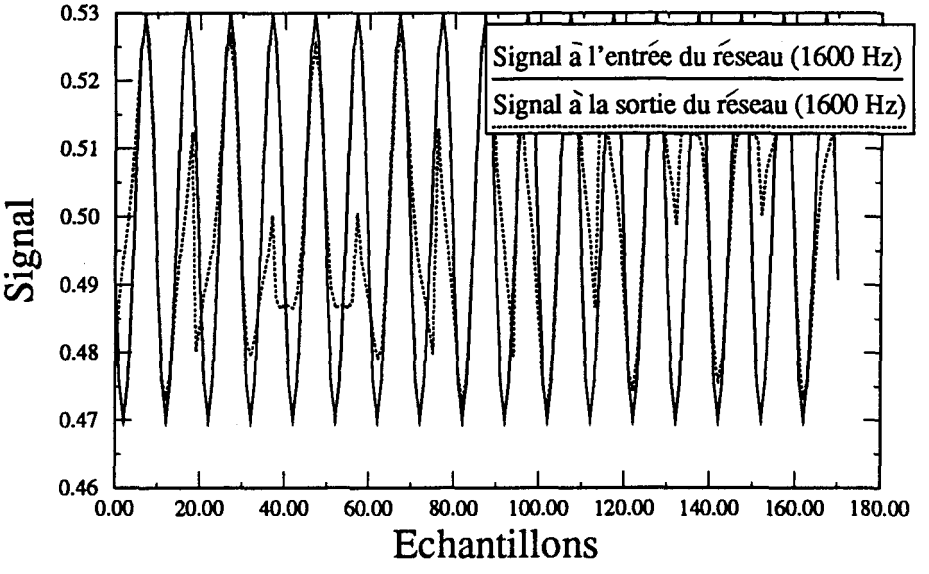

Figure 2.40 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (1600 Hz).

fréquence. Une inteprétation plausible, est que le réseau interpole entre les valeurs apprises

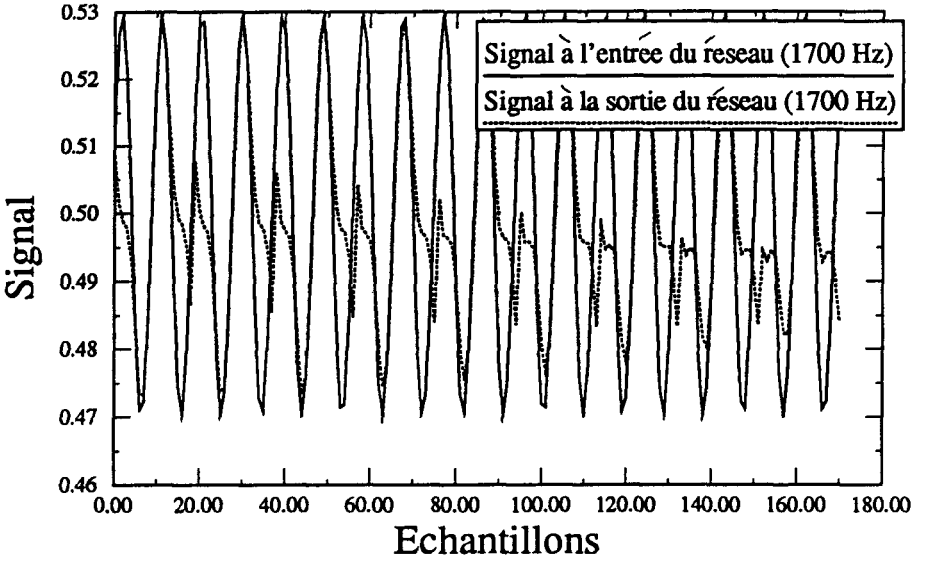

Figure 2.41 Illustration des performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (1700 Hz).

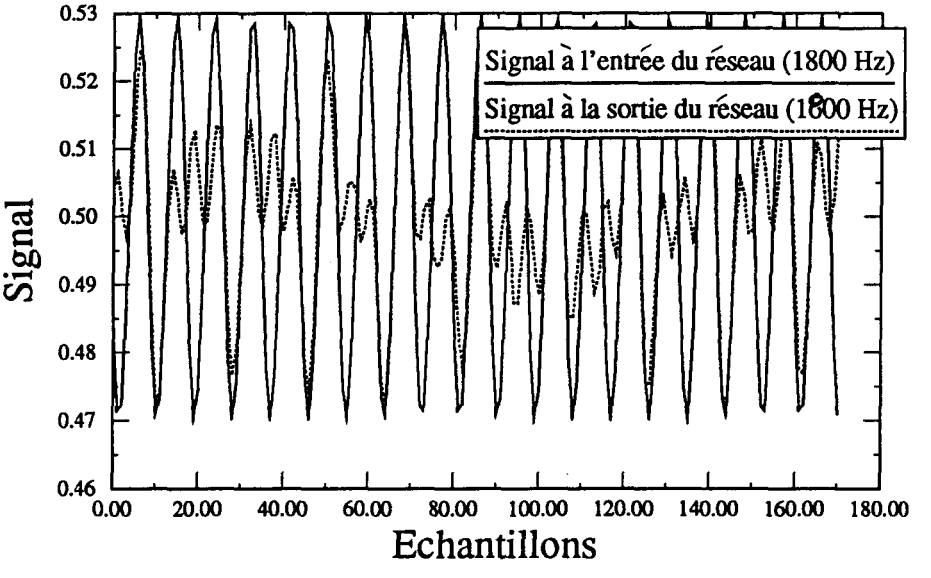

Figure 2.42 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (1800 Hz).

lors de l'apprentissage et l'interpolation est d'autant meilleure que les deux fréquences sont

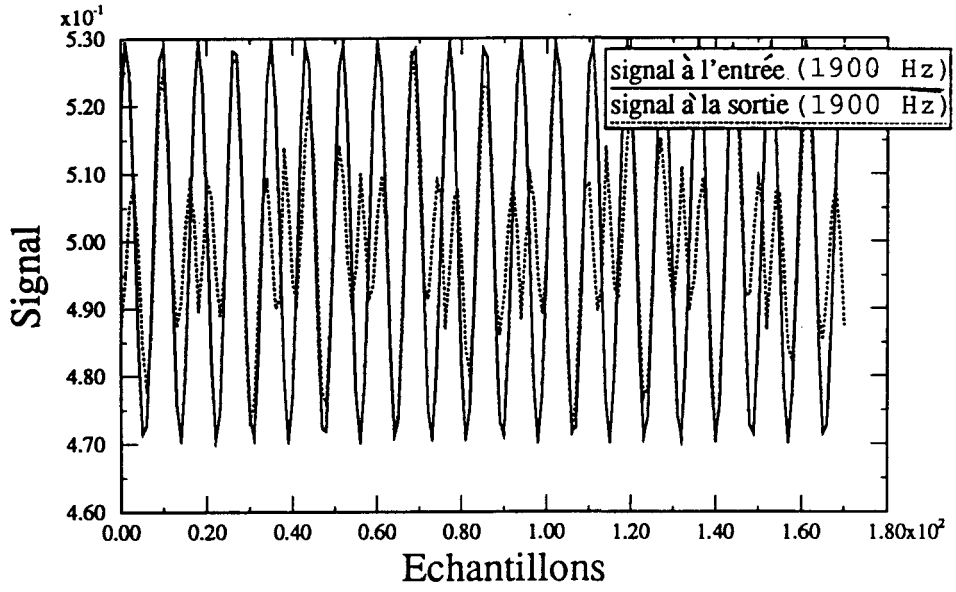

Figure 2.43 Illustration des performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (1900 Hz).

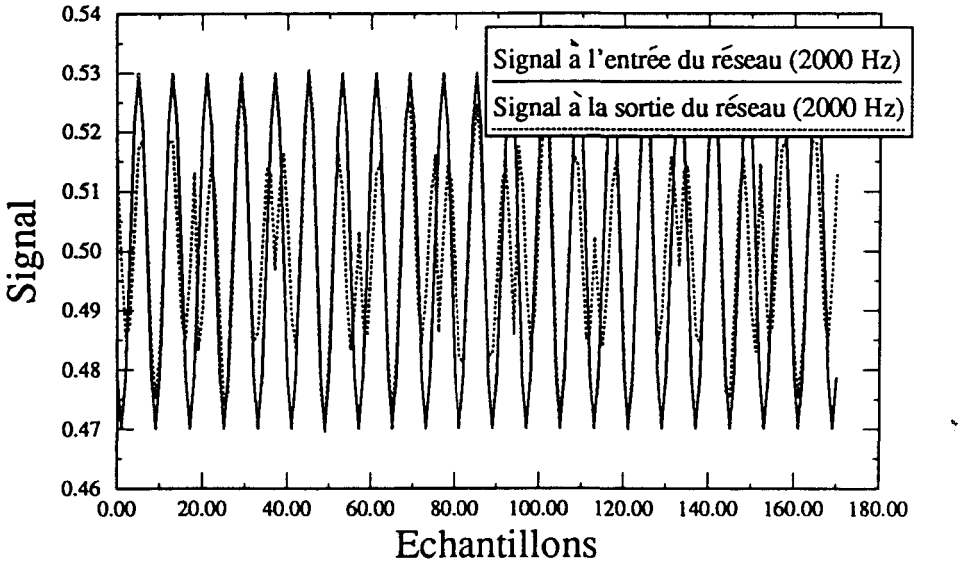

Figure 2.44 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (2000 Hz).

rapprochées (fréquence présentée et fréquence apprise).

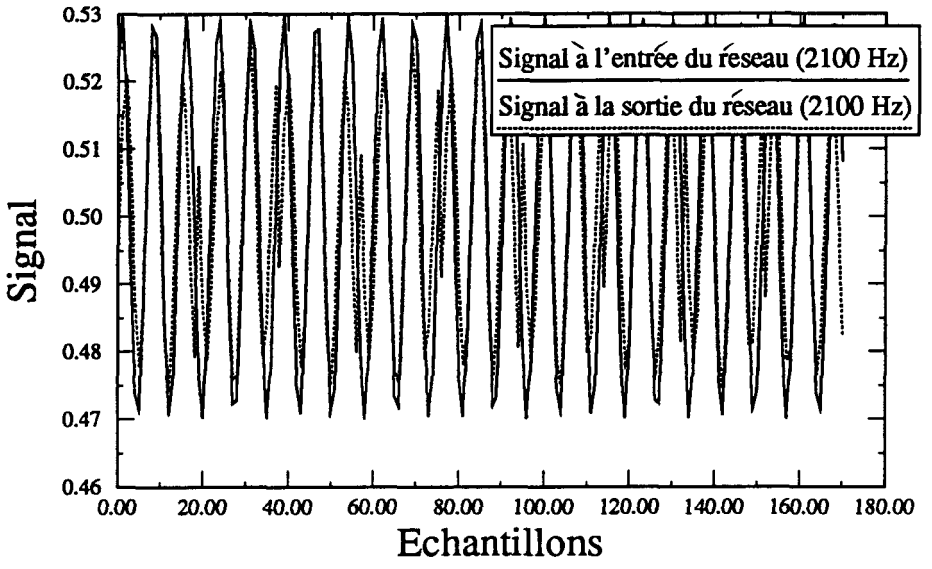

Figure 2.45 Illustration des performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (2100 Hz).

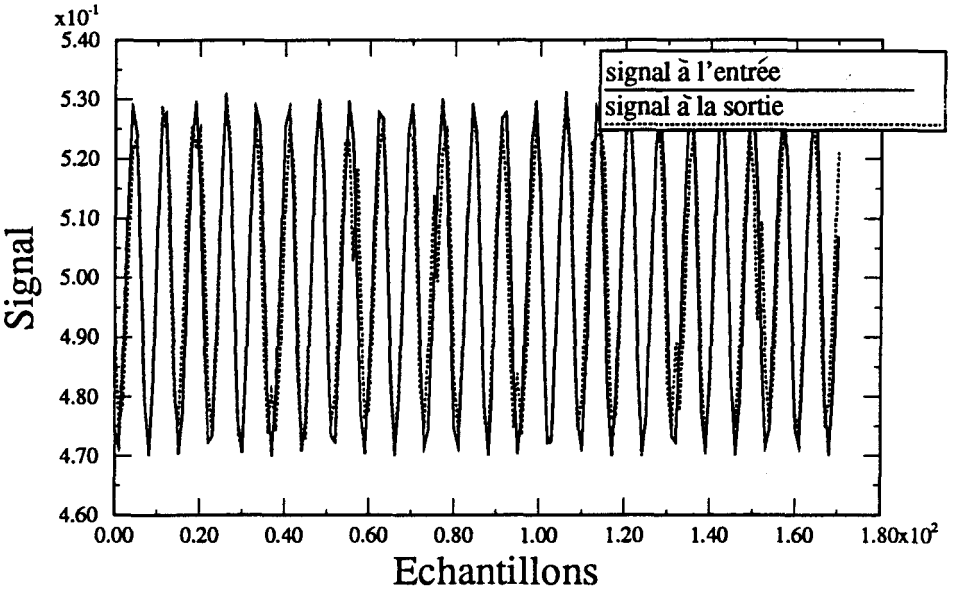

Figure 2.46 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (2200 Hz).

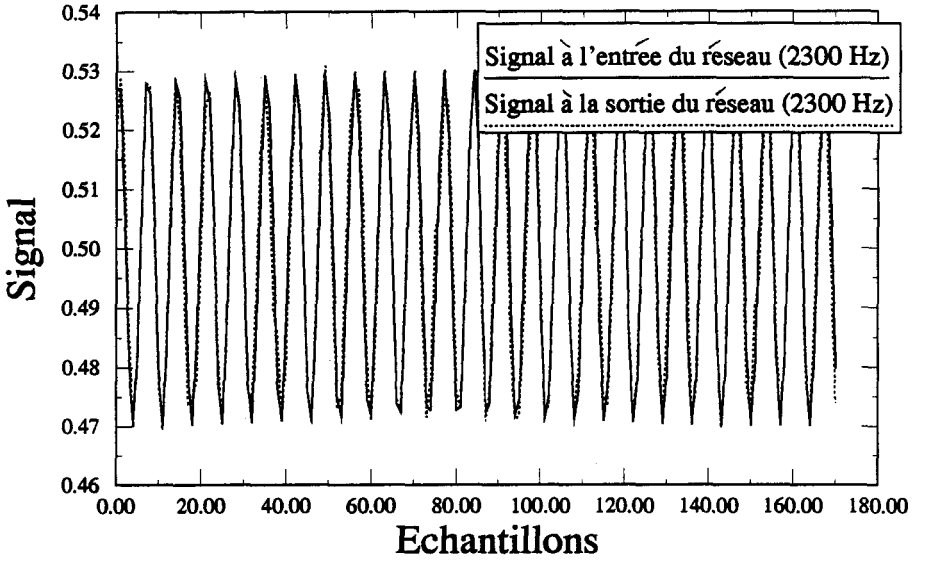

Figure 2.47 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (2300 Hz).

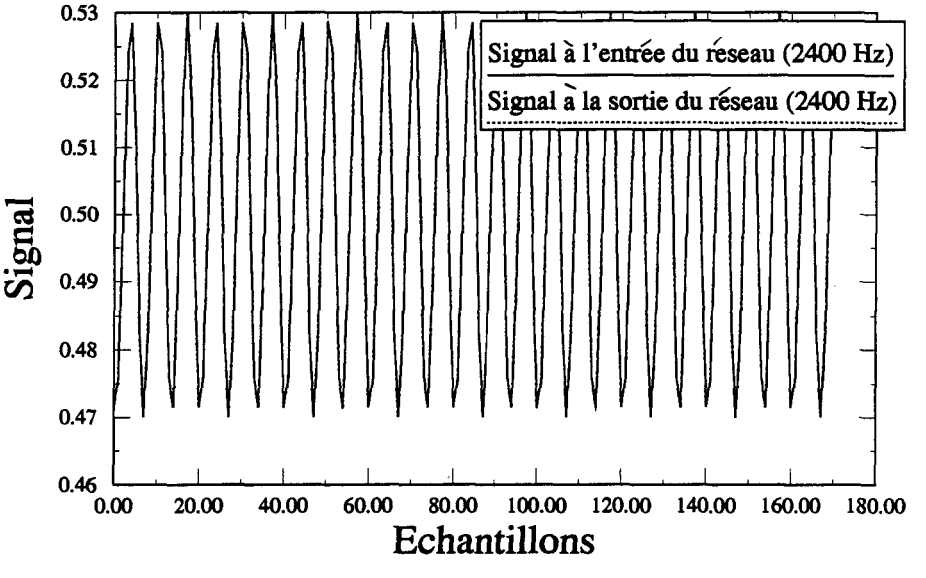

Figure 2.48 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence apprise lors de l'apprentissage (2400 Hz).

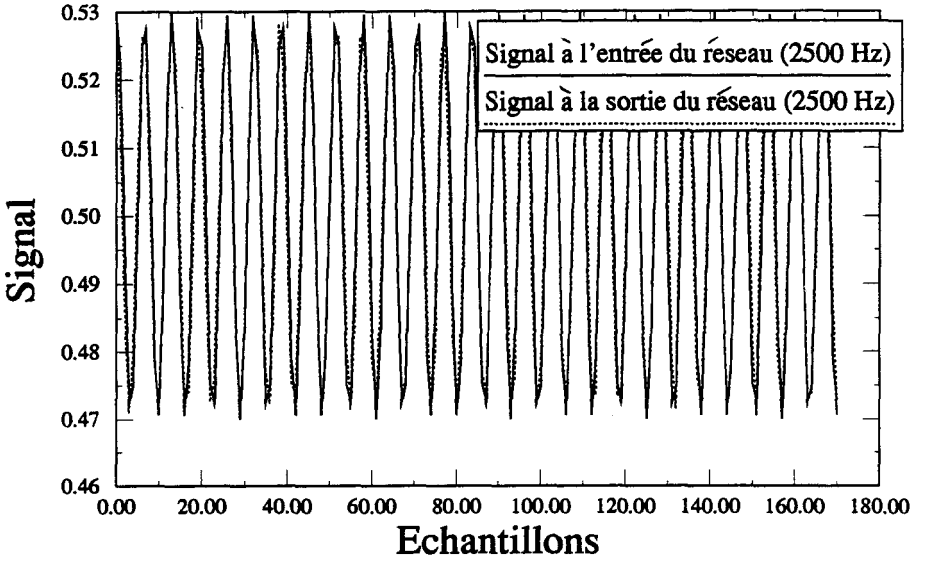

Figure 2.49 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (2500 Hz).

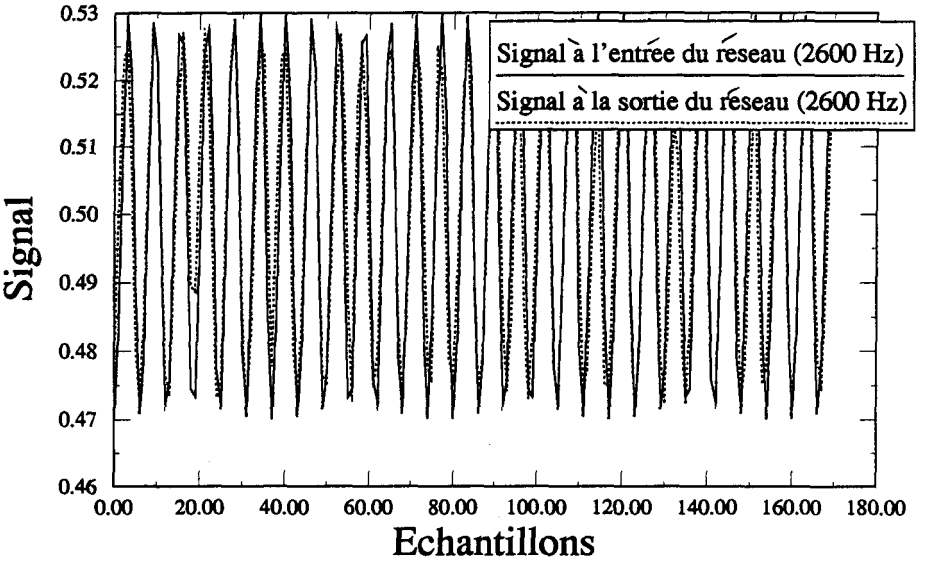

Figure 2.50 Illustration des performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (2600 Hz).

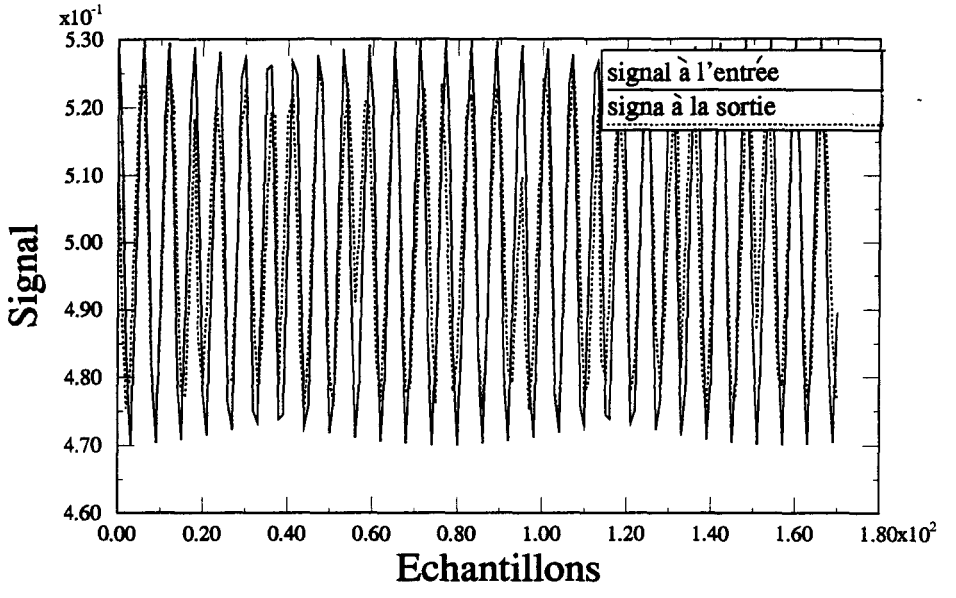

Figure 2.51 Performances du réseau quant à la généralisation des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage (2700 Hz).

En guise de conclusion, nous dirons qu'un réseau est capable de généraliser au niveau des fréquences dans une plage spécifique (+/- 100) Hz de la fréquence apprise.

Récapitulation

Nous aimerions préciser que les expériences effectuées dans cette partie ont porté sur l'étude d'un paramètre à la fois, soit l'étude de la variation de l'amplitude (fréquence fixe), soit celle de la variation de la fréquence (amplitude fixe).

Le réseau de neurones proposé est en mesure de généraliser au niveau de l'amplitude: +/- 2000 aux alentours de l'amplitude apprise. Il est également capable de généraliser au niveau des fréquences: +/- 100 Hz aux alentours de la fréquence apprise . Donc, ces deux points sont capitaux pour l'application de l'approche neurale au filtrage d'un signal dynamique (les paramètres varient dans le temps) ou d'un signal périodique sinusoïdal (les paramètres sont fixes).

2.9 Analyse et discussion de l'algorithme d'apprentissage

Dans ce chapitre, nous avons essayé de dériver et de mettre l'emphase sur les méthodes d'apprentissage et surtout sur l'algorithme de la rétropropagation (qui n'est qu'une généralisation de l'algorithme de LMS "Least Mean Square"). Nous avons exposé à l'annexe A les méthodes de correction de l'erreur, celles n'utilisant pas le gradient (αLMS) et celles utilisant le gradient (μ LMS, rétro-propagation...). Ces méthodes étaient appliquées, dans un premier temps, à des réseaux dont les noeuds n'ont que des fonctions linéaires (pas sigmoïde) à leurs sorties. Nous avons vu leurs performances et surtout leurs convergences quasi-absolues (forme parabolique de l'évolution de l'erreur) vers un minimum global. Nous avons vu l'alternative des méthodes utilisant le gradient pour enfin aboutir à l'algorithme de la rétropropagation. Nous avons vu, à travers les différents travaux, les avantages et les inconvénients de cet algorithme. Nous avons constaté que plusieurs travaux sont quasi-unanimes pour dire que la rétropropagation non-regroupée est deux fois plus rapide et tend plus à atteindre le minimum global que la rétropropagation regroupée (rétropropagation-cumulée) qui, on le sait, fait une bonne approximation du gradient de l'erreur. Nous avons exploré et analysé les travaux d'Halbert White [87, 90, 92], qui ont bien illustré le parallèle avec l'approche statistique de Robbin et Monro (1951) pour expliquer le fonctionnement de la rétropropagation.

A travers nos lectures, nous avons pu analyser l'algorithme basé sur l'approche statistique de James Ting-Holo (1991) [84], qui semble être très performant sur des données statiques, mais il faudra le vérifier sur des données de la parole (des signaux dynamiques, puis en voir la généralisation. Nous avons passé en revue également certains travaux de Y. Bengio (1991) montrant que l'accélération de la convergence peut entraîner des problèmes de généralisation du réseau.

Nous avons constaté que le "TSLA" proposé par Williams et Zipper (1989) était en mesure de stocker des signaux périodiques et de les reconnaître, puis de les regénérer indéfiniment. Nous avons réalisé la même expérience, avec un réseau de type propagation-

avant (feed-forward) ayant 3 couches dont une cachée. De notre côté, nous avons utilisé la rétropropagation rapide (standard). Les résultats sont concluants, malgré la différence des conditions expérimentales.

Nous avons constaté que plusieurs auteurs prétendent que leurs algorithmes sont les plus rapides et les plus performants. Il serait certainement très intéressant de pouvoir les comparer sur les mêmes données i.e sur un signal temporel dynamique, afin de vérifier leurs performances.

Pour résumer ce chapitre, nous retiendrons des articles consultés que l'algorithme de la rétropropagation demeure un choix prometteur pour notre recherche. Plusieurs travaux le confirment, entre autres ceux de Y. Bengio (1991) et Halbert White (1990). Cependant, la méthode de la rétropropagation non-regroupée demeure intéressante malgré le problème des minima locaux. Il y a toujours une possibilité de régler ce dernier, notamment en faisant plusieurs essais d'apprentissage avec des poids initiaux différents, et en prenant le meilleur de ceux-ci (du point de vue de la convergence). Les contraintes de temps et d'argent ont évidemment certaines répercussions.

En ce qui concerne l'algorithme d'apprentissage, nous tenons à attirer l'attention sur le fait que les deux algorithmes (R. standard et R. rapide) peuvent s'appliquer aux deux sortes de structures de réseaux (récursifs ou non) moyennant un léger changement dans la programmation de l'algorithme. Le compromis à faire se trouve au niveau de la durée de l'apprentissage, de la convergence et de la généralisation.

En ce qui concerne l'algorithme d'apprentissage, notre choix s'est porté sur la rétropropagation non-regroupée i.e rétropropagation rapide [22]. Ce choix promet un temps de convergence deux fois plus rapide que la rétropropagation regroupée (rétropropagation cumulée). Pourquoi cet algorithme et non un algorithme basé sur une approche statistique tel que celui proposé par James Ting Holo [84] ? En voici les raisons:

— l'algorithme statistique est évalué à l'aide d'un réseau multi-couches basé sur le

perceptron, avec une fonction discrète. Afin de pouvoir appliquer ces résultats à notre cas, il aurait fallu démontrer les performances sur le genre de signal que nous allons traiter;

— cet algorithme (statistique) utilise une architecture réduite (porte logique);

— cet algorithme a déjà démontré ses performances en ce qui concerne la convergence. Cependant, le problème demeure au niveau de la généralisation.

2.10 Analyse et discussion

Dans ce chapitre, nous avons vu les résultats des simulations et des expériences effectuées sur les possibilités de filtrage d'un réseau de neurones. Pour débuter l'étude des principaux paramètres qui régissent le fonctionnement d'un réseau de neurones, nous avons analysé d'abord l'importance de la normalisation sur les performances du réseau, plus particulièrement en ce qui concerne la mémorisation, la généralisation au niveau de l'amplitude et la généralisation au niveau des fréquences. Nous avons constaté qu'une normalisation absolue (valeurs limites: maximale et minimale fixes) donne de meilleurs résultats en ce qui a trait à la mémorisation et la généralisation d'amplitude.

Ensuite, nous avons défini et étudié les paramètres qui peuvent influencer la mémorisation du réseau. Les principaux paramètres en question sont la relation qui peut exister entre le nombre d'échantillons par période et le nombre de noeuds dans la couche d'entrée ($n = k.p$, où k est un réel positif ; cas où $n > p$ et cas où $n < p$), le nombre de noeuds à l'entrée du réseau, le nombre de noeuds dans la couche cachée, le nombre de noeuds dans la couche de sortie, le type de fonction de transfert et la convergence de l'algorithme d'apprentissage.

Nous avons constaté que si le nombre de noeuds à l'entrée est un multiple du nombre d'échantillons par période du signal, alors la sortie du réseau peut largement être influencée. Nous avons pu établir une relation qui aiderait beaucoup l'apprentissage des patrons périodiques. Cette relation se manifeste sous forme d'introduction de décalage artificiel

des vecteurs de données. Ces décalages aident le réseau de neurones à mieux caractériser le signal périodique, pour éventuellement le mémoriser. Pour mieux caractériser un patron périodique dans un réseau, il faut effectuer l'apprentissage au moins sur le carré de la période du signal (du point de vue du nombre de points). Cette remarque est plus facilement applicable aux hautes fréquences (f > 850 Hz) qu'aux basses fréquences. La raison de ce phénomène est la suivante: plus la fréquence est basse, plus le nombre de points par période (ce qui dépend, bien sûr de la fréquence d'échantillonnage) est élevé et, par conséquent, le carré de ce nombre est élevé (p^2 échantillons).

Plusieurs chercheurs ayant étudié les réseaux de neurones ont démontré la relation qui existe entre le nombre de patrons (vecteurs) à apprendre et le nombre de noeuds se trouvant dans la couche cachée. Tous reconnaissent que ces deux paramètres doivent être égaux. Les expériences effectuées dans le cadre de ce travail vont dans le même sens que ce qui a été dit précédemment. La seule différence est liée à la complexité du signal à apprendre: plus le signal à apprendre est facile (propriétés constantes dans le temps), moins nous aurons besoin de noeuds dans la couche cachée; par contre, si le signal est complexe, nous aurons besoin d'un nombre de noeuds équivalent au nombre de patrons appris. Le surplus de noeuds dans cette couche aura tendance à générer un bruit, qui même si parfois bénéfique, pourrait être destructeur dans d'autres applications.

En ce qui concerne le nombre de noeuds dans la couche de sortie, nous pouvons dire d'après nos expériences que pour effectuer le filtrage, il ne faut pas perdre l'information utile, car le fait de réduire le nombre de noeuds dans la couche de sortie peut générer un phénomène d'oubli indésirable. Par contre, un surplus de noeuds dans cette couche, en prenant comme référence le nombre de noeuds dans la couche d'entrée, aurait l'effet contraire c'est-à-dire la génération d'un surplus d'information qui se manifeste sous forme de bruit. Alors, idéalement, le nombre de noeuds dans cette couche devrait être équivalent à celui dans la couche d'entrée.

Nous avons aussi vu , que la fonction de transfert, située à la sortie du neurone (couche

cachée — couche de sortie) a une grande influence sur la convergence et par conséquent sur la mémorisation. Le fait de choisir une fonction linéaire (sommateur) a pour effet de compliquer l'apprentissage et de rendre presque impossible la mémorisation. Par contre, le choix d'une fonction non-linéaire et non-injective (pour un élément de l'ensemble de départ, on aurait au moins deux éléments dans l'ensemble d'arrivée), entraînerait une évolution instable de l'erreur, ce qui peut affecter la mémorisation de l'information. Alors, les résultats de cette partie nous montrent que le choix de la fonction sigmoïde est effectivement le meilleur, car cette fonction est injective et assure une plus grande stabilité (erreur) de la convergence, et par conséquent de la mémorisation de l'information. Les expériences ont très bien démontré cette affirmation.

Pour terminer cette partie concernant les paramètres qui influencent la mémorisation, nous pouvons dire qu'une bonne convergence de l'algorithme d'apprentissage est équivalente à une bonne mémorisation, ce qui ne veut pas dire nécessairement une bonne généralisation. Quand on parle de mémorisation, on fait référence à la mémorisation des paramètres essentiels dans le signal, et non à la mémorisation des patrons du signal — réseau associatif.

Maintenant, nous allons discuter des paramètres qui peuvent infuencer la généralisation. Le traitement d'un signal dynamique (signal dont les propriétés varient dans le temps) nécessite un système capable de mémoriser un certain nombre de relations importantes afin de pouvoir regénérer n'importe quel patron à la sortie du réseau de neurones, périodique ou non. Mais pour assumer cette fonction, nous avons besoin d'un repère fixe, auquel nous nous référons pour mesurer le niveau d'amplitude. Ce qui est vrai pour le niveau d'amplitude peut aussi l'être pour la fréquence. En nous basant sur les travaux antérieurs et sur les expériences effectuées dans le cadre de ce travail, expériences qui démontrent que le réseau de type propagation-avant est en mesure d'apprendre les relations intrinsèques incarnées dans les données apprises, nous souhaitons concevoir un réseau de neurones, à même d'effectuer deux choses: la mémorisation et la généralisation.

Les expériences effectuées dans le présent travail ont montré que le réseau est capable de généraliser au niveau des amplitudes et des fréquences. La généralisation du niveau d'amplitude est liée aux données d'apprentissage. L'apprentissage effectué sur des signaux sinusoïdaux d'un seul niveau, a montré que le réseau est en mesure de généraliser l'amplitude, mais dans une plage restreinte à +/- 2000 du niveau appris, ce qui est très concluant. Toutefois, notre objectif idéal est d'avoir une généralisation qui peut se faire sur une plage de niveau assez large. Pour ce faire nous avons gardé la même architecture, fonctionnant sous le même algorithme. Seules ont été modifiées les données d'apprentissage. Nous avons fait varier le niveau d'amplitude de 50 à 40000. Les résultats obtenus sont meilleurs que les précédents. Cependant, ils favorisent les haut niveaux au détriment des bas niveaux. De plus, cette performance pour les hauts niveaux affecte sérieusement la fréquence des bas niveaux.

En ce qui concerne la généralisation des fréquences, les performances sont intéressantes. Le réseau ayant appris deux types de fréquences distinctes est en mesure de généraliser autour de ces deux fréquences. La généralisation est d'autant meilleure que la fréquence présentée à l'entrée du réseau est proche de celle apprise. Cependant, la généralisation demeure tout de même assez bonne sur une plage de +/- 100 Hz.

Récapitulons: un réseau de neurones multi-couches à propagation-avant, fonctionnant sous l'algorithme de la rétropropagation rapide, est susceptible de mémoriser de l'information et de généraliser au niveau des amplitudes et des fréquences. La mémorisation est liée à certains paramètres, dont le principal est la convergence de l'algorithme d'apprentissage.

CHAPITRE 3

CONCEPTION DE FILTRES: EXPÉRIENCES ET RÉSULTATS

Dans ce chapitre, nous exposerons les résultats des expériences et des simulations effectuées sur les capacités d'un réseau de neurones formels à filtrer un signal. Nous élaborerons nos hypothèses de départ. Nous utiliserons les résultats des simulations concernant la mémorisation et la généralisation pour le filtrage d'un signal pur.

Nous aimerions préciser que, au début de chaque simulation, nous nous assurons que les connexions synaptiques - poids - sont générées de façon aléatoire de telle sorte que les conditions initiales seront différentes pour chaque cas.

3.1.1 Application directe de la généralisation d'amplitude et de fréquence: conception de filtre

Comme un réseau de neurones semble en mesure de généraliser au niveau de l'amplitude et au niveau des fréquences, nous diposons des outils de base pour la conception de systèmes de filtrage d'un signal dynamique (filtres passe-bas, passe-haut, passe-bande et coupe-bande). Dans cette section, nous allons vérifier si notre modèle peut filtrer un signal bruité avec un rapport signal/bruit de -10 dB, ce qui est important. Il est à noter que nous avons testé l'aptitude du réseau pour divers signaux d'entrée (signal nul, signal périodique); son comportement semble cohérent.

3.1.1.1 Expérience # 1: conception d'un filtre passe-bas

Cette expérience fait l'étude du passe-bas basé sur l'approche neurale.

Fréquence	Nombre d'échantil-lons par période	Carré du nombre d'échantillons par période	Nombre d'échantil-lons proposés par période	Carré du nombre d'échantillons proposé	Numéro du signal
50	320	102400	323	104329	1
250	64	4096	76	5776	2
450	35	124	38	1444	3
650	24	606	38	1444	4
850	19	232	19	361	5
1050	15	164	19	361	6
1250	13	164	19	361	7
1450	11	122	19	361	8
1650	10	94	19	361	9
1850	9	74	19	361	10
2050	8	60	19	361	11
Total	560	110983	589	1054159	****

Tableau 3.1 Nombre d'échantillons établis dans le fichier d'apprentissage.

Données d'apprentissage:

— données d'apprentissage : 10 signaux sinusoïdaux périodiques d'amplitude de 3000 et de fréquence variant de 50 Hz à 2050 Hz par intervalle de 200 Hz;

— 589 échantillons;

— amplitude de 3000;

— fréquence d'échantillonnage de 16 kHz;

— normalisation absolue (+/- 40000);

— algorithme d'apprentissage: la rétropropagation rapide;

— une architecture de réseau de type propagation-avant, avec 19 noeuds dans les couches d'entrée et de sortie et de 20 noeuds dans la couche cachée;

— une simulation sur une station sparc 2;

— Approximation: nous faisons une approximation pour les quatres premiers signaux; la contrainte imposée dans l'apprentissage (l'apprentissage sur p^2 échantillons) n'est pas respectée. Cela a pour conséquence de ramener au minimum le nombre d'échantillons contenus dans le fichier d'apprentissage.

3.1.1.1.1 Résultats

Les résultats obtenus concordent bien avec nos prévisions. Le problème réside au niveau de la fréquence de coupure (nous sommes conscients que nous faisons référence à des systèmes linéaires). Le réseau filtre bien le signal sur la plage spécifiée. Le réseau est testé sur des signaux bruités et non-bruités. Les résultats sont illustrés dans les figures 3.1 à 3.18. Nous allons illustrer les performances du modèle du filtre passe-bas sur trois catégories de signaux : la première catégorie de signal a une fréquence (2000 Hz) contenue dans la plage couverte par le filtre (figures 3.1 à 3.7 illustrent les performances du filtre sur un signal composé de 400 Hz et 1600 Hz), la seconde catégorie de signal est composée de deux fréquences (2000 Hz et 4000 Hz) dont une seule se trouve dans la plage permise (figure 3.8) et la troisième catégorie de signal a une fréquence se trouvant complètement à l'extérieur de la plage permise (2 kHz) (figures 3.11 à 3.14). Les figures 3.15 à 3.18 illustrent les performances du filtre sur des données de parole (non-bruitée et bruitée avec S/N =-10 dB)

Remarque

Les résultats présentés dans la figure 3.12 sont uniquement liés à l'apprentissage du réseau et ne correspondent nullement à un patron relié au signal nul. Il n'y aucune relation apparente entre cette sortie et la réponse du réseau à un signal nul.

3.1.1.1.2 Interprétation des résultats

Pour tester la capacité du réseau à ne laisser passer que les basses fréquences ($f < 2.05$ kHz) sans introduire de dégradation majeure, il faut effectuer des tests sur un signal

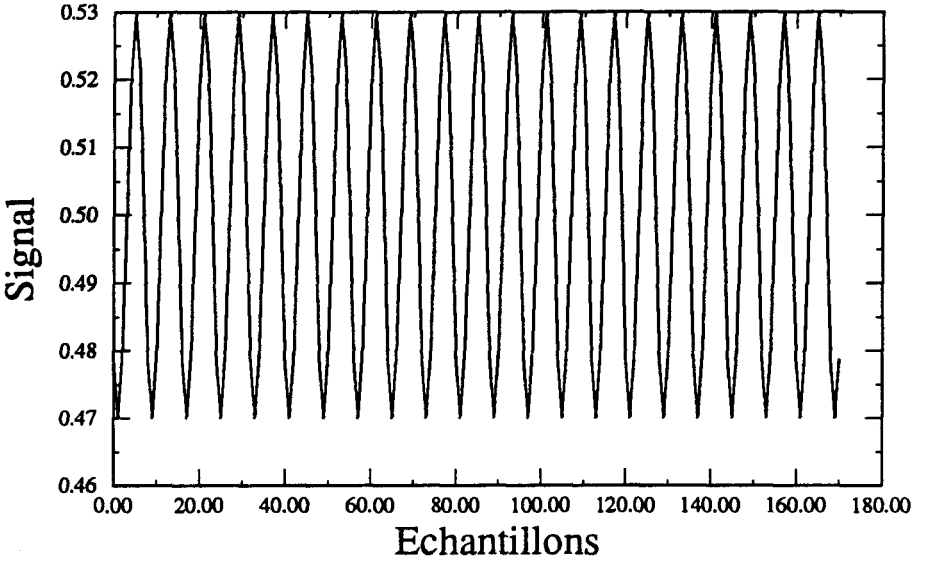

Figure 3.1 Filtrage des basses fréquences (f_c =2.05 kHz). Signal fourni à l'entrée du réseau; fréquence non-apprise lors de l'apprentissage, mais présente dans la plage permise.

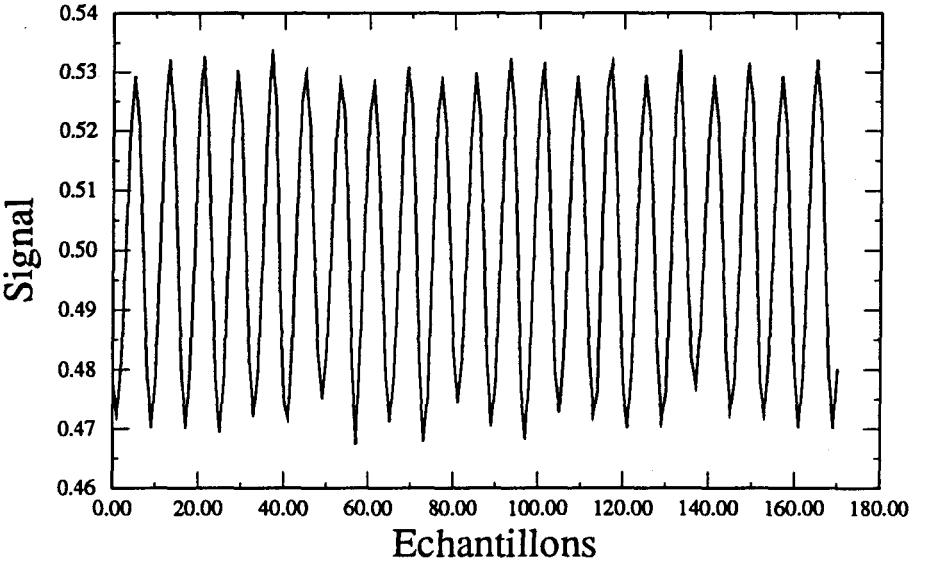

Figure 3.2 Filtrage des basses fréquences (f_c =2.05 kHz). Signal obtenu à la sortie du réseau (filtre); fréquence non-apprise lors de l'apprentissage, mais présente dans la plage permise.

pur (sinusoïdal) d'une fréquence de 2000 Hz et d'une amplitude de 3000. Les résultats

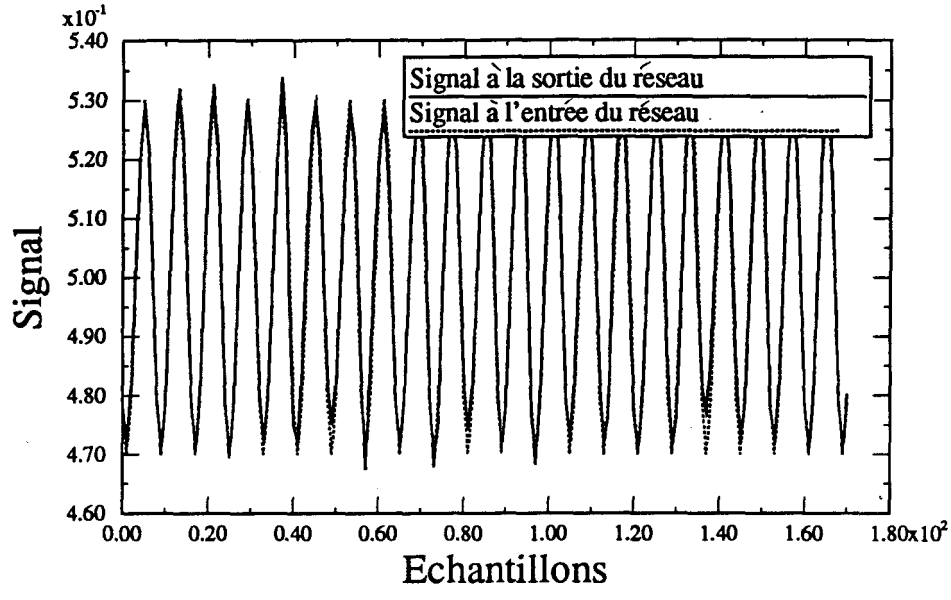

Figure 3.3 Filtrage des fréquences. Superposition des signaux à l'entrée et à la sortie du réseau; fréquence non-apprise lors de l'apprentissage 2000 Hz, mais présente dans la plage permise.

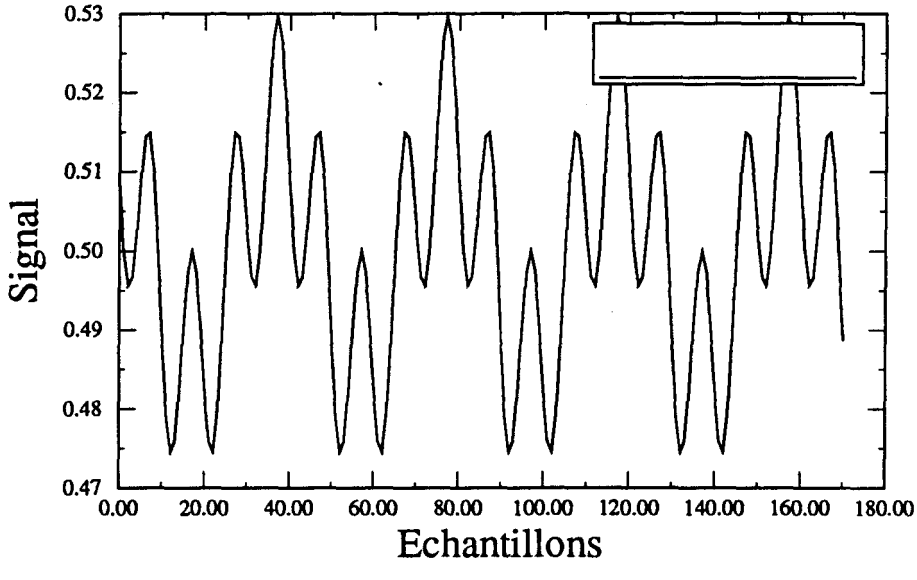

Figure 3.4 Signal à l'entrée du réseau de neurones; composé du produit de 2 signaux sinusoïdaux de fréquences respectives de 600 Hz et 1000 Hz.

obtenus nous montrent que le réseau est en mesure de regénérer le signal sans introduire

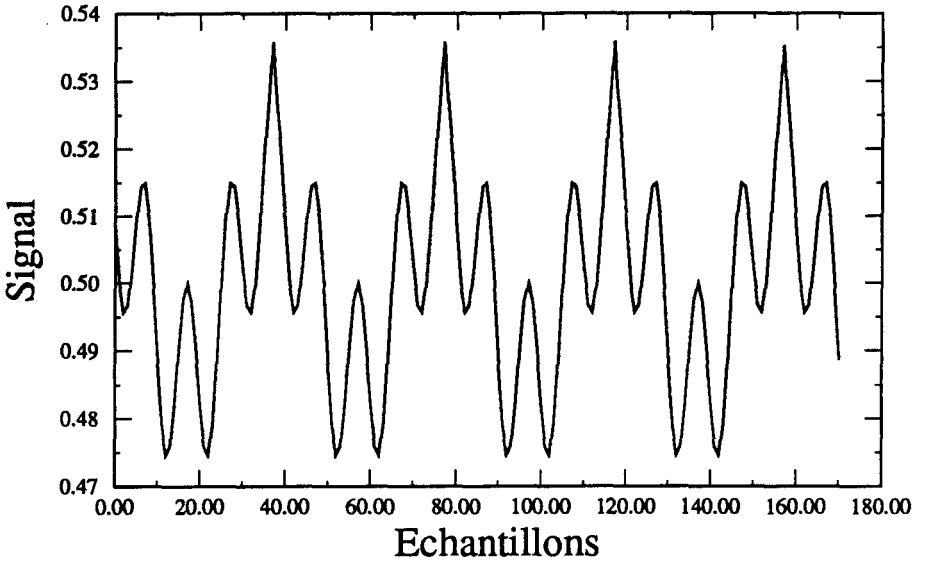

Figure 3.5 Signal à la sortie du réseau de neurones; c'est le produit de 2 signaux sinusoïdaux de fréquences respectives de 600 Hz et 1000 Hz.

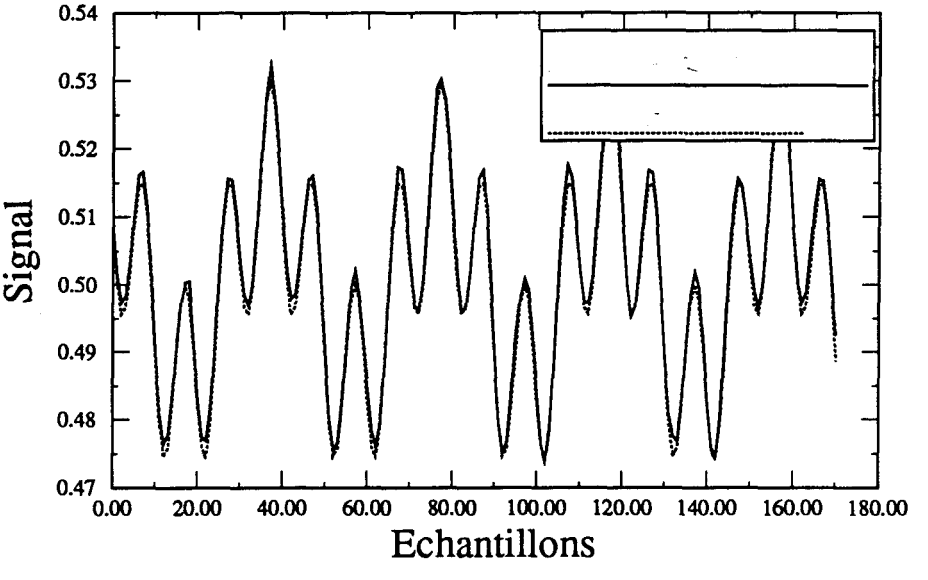

Figure 3.6 Superposition des signaux à l'entrée et à la sortie du réseau de neurones; (produit de 2 signaux sinusoïdaux de fréquences respectives de 600 Hz et 1000 Hz).

de dégradation au niveau de la périodicité du signal. Nous aurons néanmoins une légère

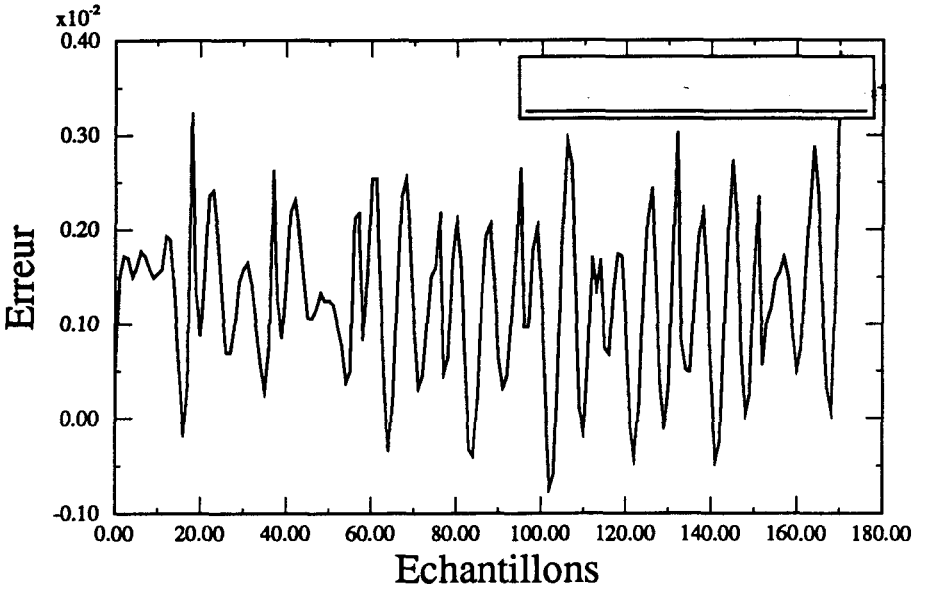

Figure 3.7 Erreur entre les signaux à l'entrée et à la sortie du réseau de neurones; (produit de 2 signaux sinusoïdaux de fréquences respectives de 600 Hz et 1000 Hz).

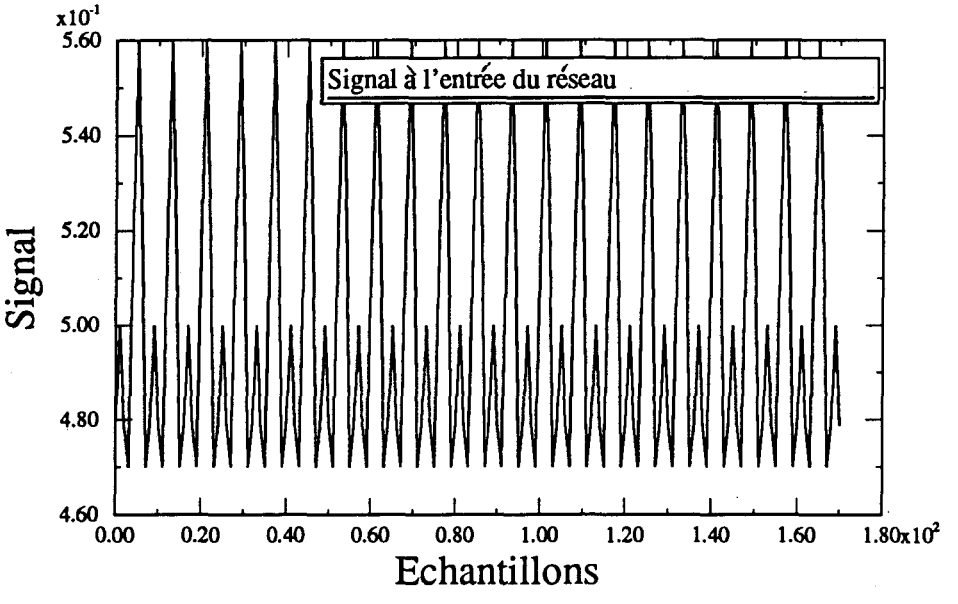

Figure 3.8 Filtrage des basses fréquences (f_c =2.05 kHz). Signal composé de 2 raies spectrales de fréquences respectives de 2000 et 4000 Hz, fournies à l'entrée du réseau (filtre); fréquence non-apprise lors de l'apprentissage.

modification au niveau de l'amplitude. L'information spectrale est intacte. Ce cas fera

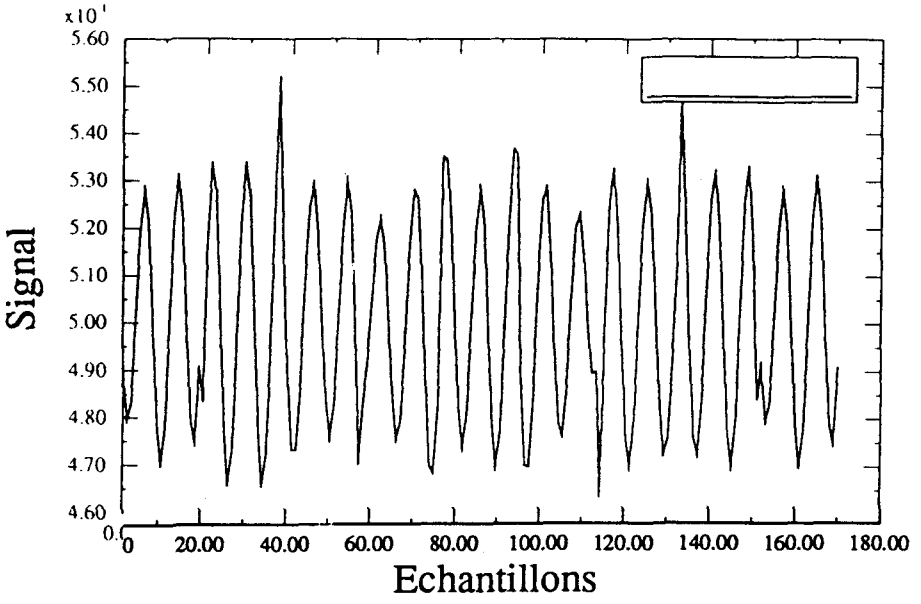

Figure 3.9 Filtrage des basses fréquences (f_c =2.05 kHz). Signal composé d'une seule raie spectrale de fréquence 2000 Hz, regénéré à la sortie du réseau (filtre); fréquence non-apprise lors de l'apprentissage.

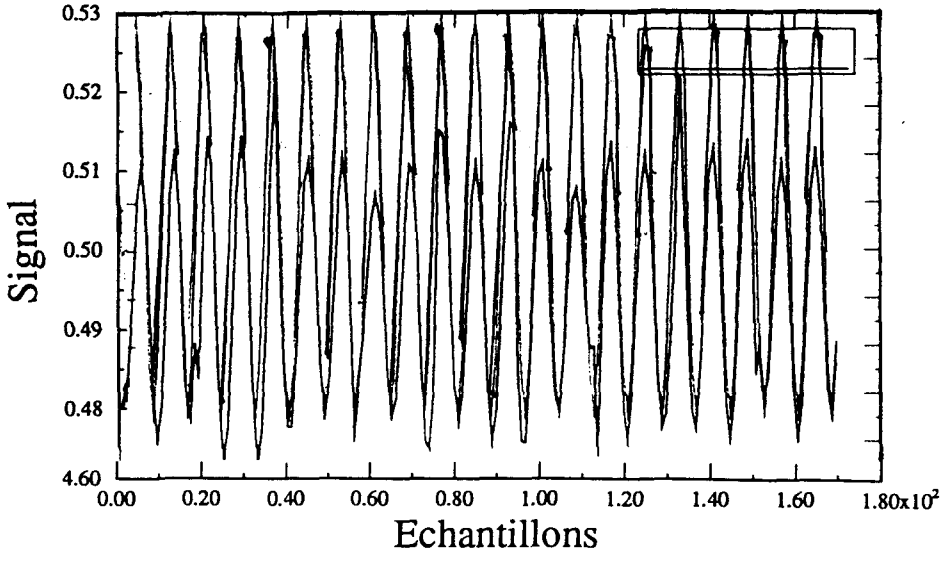

Figure 3.10 Superposition du signal regénéré par le réseau et un signal ayant une fréquence de 2000 Hz et une amplitude de 3000.

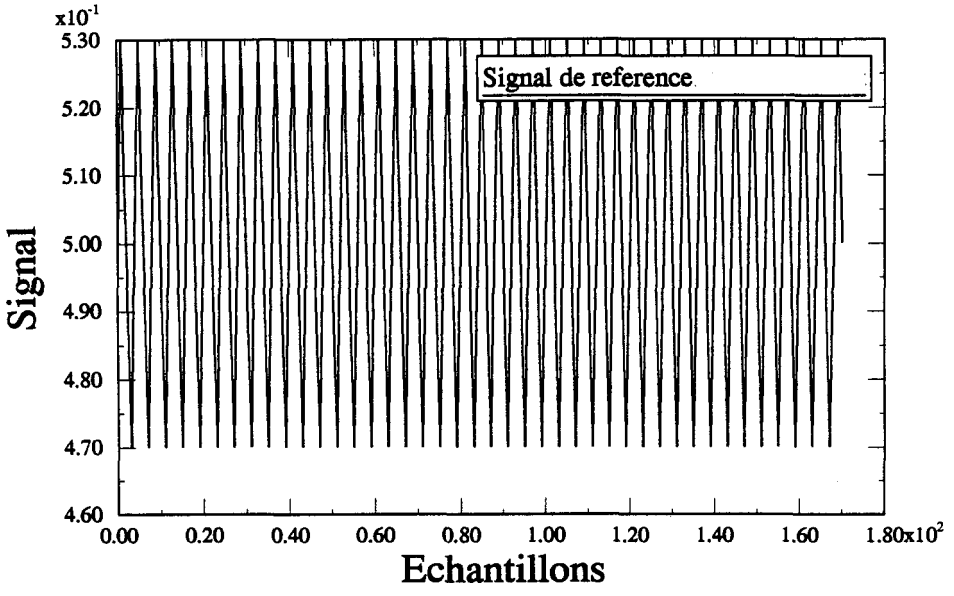

Figure 3.11 Filtrage des basses fréquences (f_c = 2.05 kHz). Signal fourni à l'entrée du réseau; la fréquence (4000 Hz) n'est pas apprise lors de l'apprentissage et se trouve complètement à l'extérieur de la plage des fréquences couverte par le filtre.

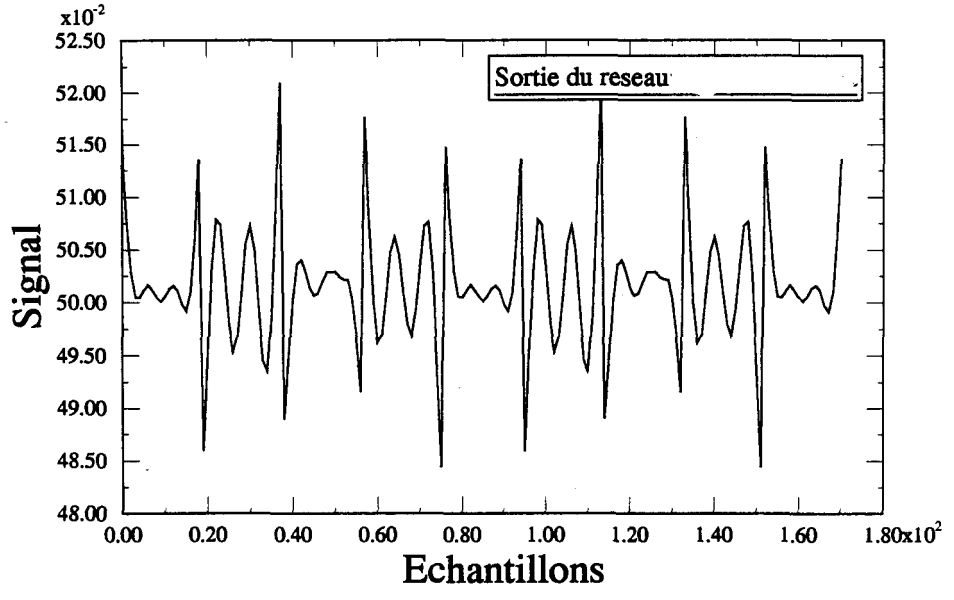

Figure 3.12 Filtrage des basses fréquences (f_c =2.05 kHz). Signal regénéré à la sortie du réseau (complètement aléatoire, mais il y a présence de démarcation des fenêtres)

l'objet d'une comparaison aux performances du modèle classique (passe-bas).

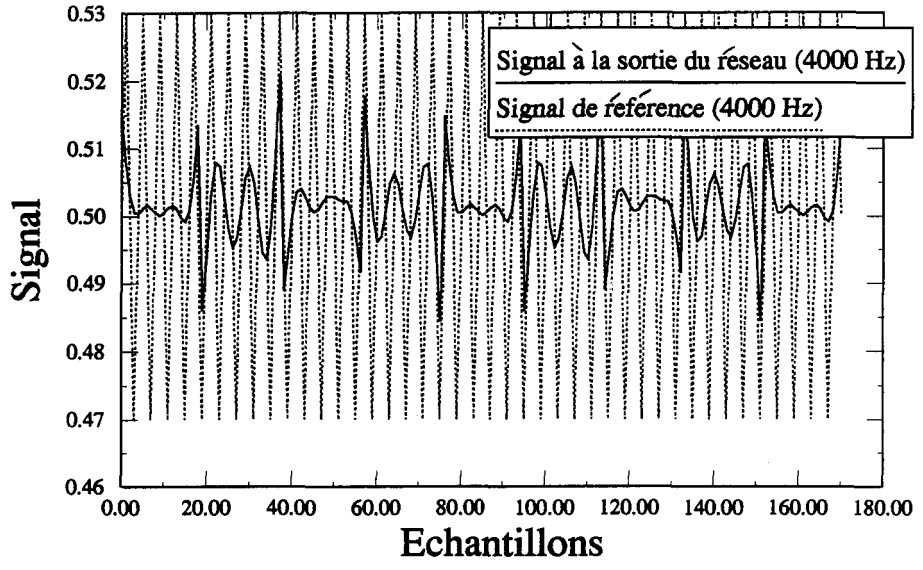

Figure 3.13 Filtrage des basses fréquences (f_c =2.05 kHz). Superposition des deux signaux (f=4000 Hz) à l'entrée et à la sortie du réseau; fréquence (4000 Hz) non-apprise lors de l'apprentissage et ne se trouve pas dans la plage permise.

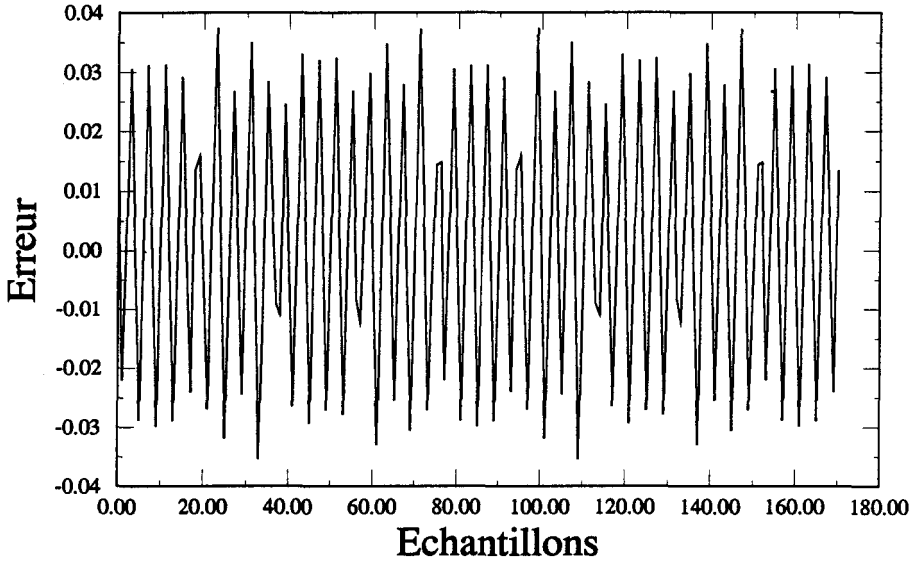

Figure 3.14 Filtrage des basses fréquences (f_c = 2.05 kHz). Erreur entre le signal à l'entrée et celui à la sortie.

Nous avons également testé ce réseau avec un signal composé d'une fréquence qui se trouve dans la plage couverte par le réseau et d'une autre se trouvant à l'extérieur de

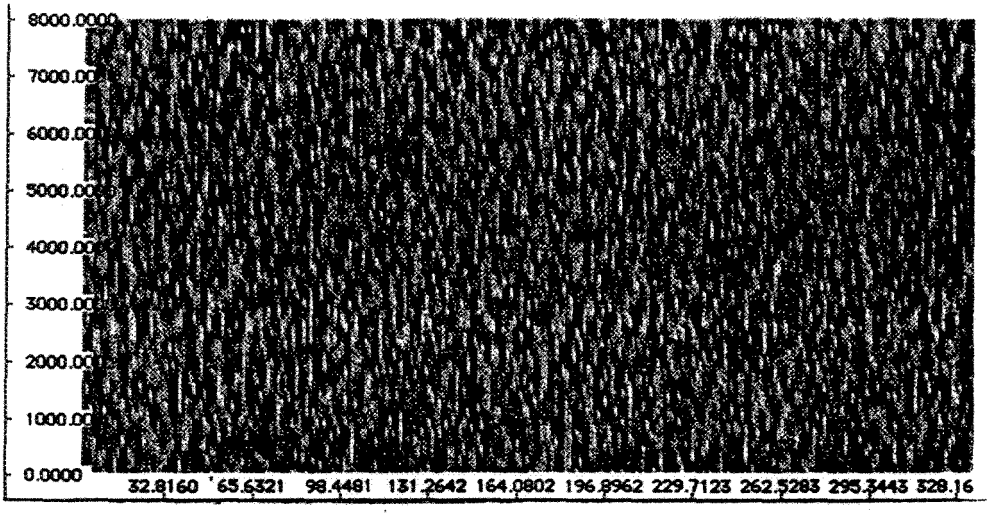

Figure 3.15 Spectrogramme d'un signal de parole bruitée "la bise" avec un rapport signal à bruit de —10 dB, présenté à l'entrée du réseau. Illustration des performances du filtre réducteur de bruit (passe-bas).

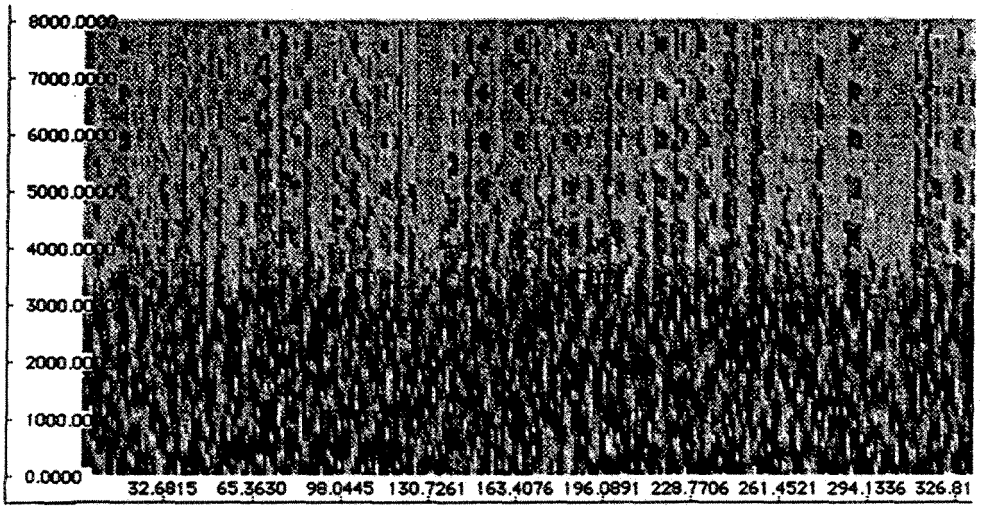

Figure 3.16 Spectrogramme d'un signal de parole bruitée, avec un rapport signal à bruit de —10 dB, obtenu à la sortie du réseau. Illustration des performances du filtre réducteur de bruit (passe-bas).

celui-ci. Le résultat obtenu est intéressant: le réseau a effectivement rejeté la fréquence

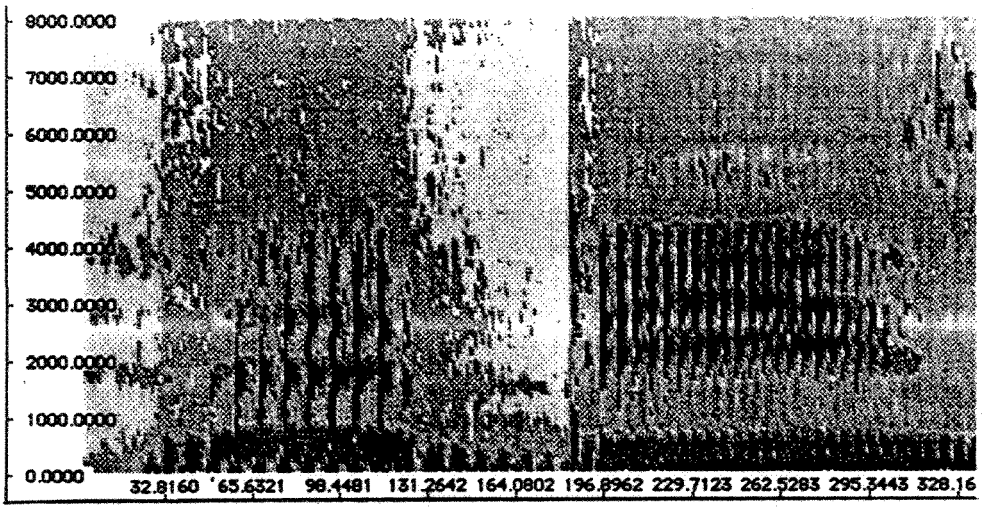

Figure 3.17 Spectrogramme d'un signal de parole non-bruitée "la bise", présenté à l'entrée du réseau. Illustration des performances du filtre (f_c=2.05 Khz) réducteur de bruit (passe-bas).

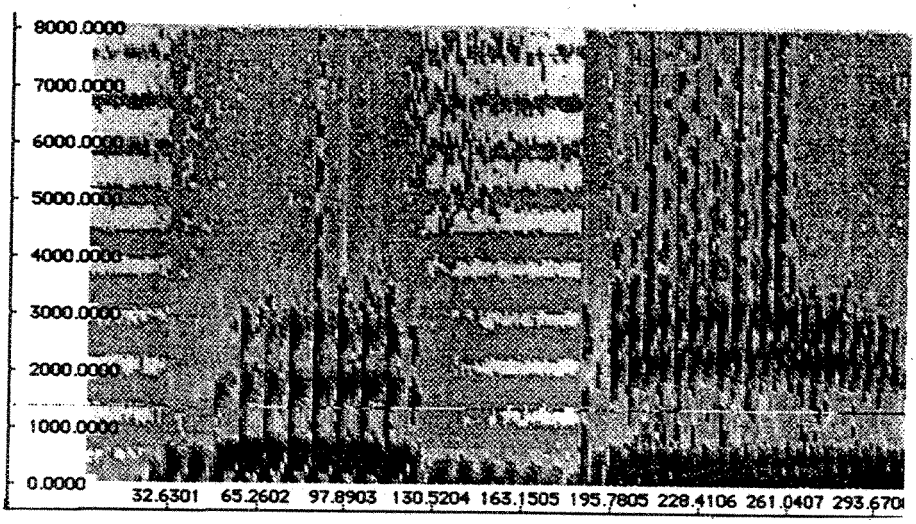

Figure 3.18 Spectrogramme d'un signal de parole non-bruitée, obtenu à la sortie du réseau. Illustration des performances du filtre (f_c=2.05 Khz) réducteur de bruit (passe-bas).

qui se trouve à l'extérieur de la plage permise. L'erreur entre le signal de référence et le

signal regénéré à la sortie du réseau varie entre 3% et 5%, ce qui est acceptable.

Le modèle filtre bien le signal bruité avec un rapport signal/bruit de -10 dB. Il laisse passer l'information dans la plage spécifiée et coupe au niveau des hautes fréquences (f > 2050 Hz). Toutefois, le modèle génère de l'énergie dans la plage extérieure (jusqu'à 3000 Hz); le dépassement (de la fréquence de coupure) peut atteindre 1000 Hz tel qu'illustré dans la figure 3.18. Cette génération d'énergie peut être causée par une discontinuité au niveau du signal se manifestant sous-forme d'un spectre large bande.

Il est important de noter que la reproduction par photocopie peut donner l'impression qu'il y a plus d'énergie à l'extérieure de cette plage (figures avec spectrogramme seulement).

L'audition du signal bruité filtré illustre une nette amélioration relative par rapport au signal de référence (signal bruité original). La comparaison avec les filtres classiques est illustrée plus loin dans le chapitre.

3.1.1.2 Expérience # 2: filtre passe-bande

Cette expérience fait l'étude du passe-bande basé sur l'approche neurale.

<u>données de l'expérience:</u>

— données d'apprentissage : 11 signaux sinusoïdaux périodiques d'amplitude de 3000 et de fréquence variant de 3000 Hz à 5000 Hz par pas de 200 Hz;
— nombre d'échantillons est décrit dans le tableau 3.2.
— algorithme d'apprentissage: la rétropropagation rapide;
— architecture de réseau de type propagation-avant, avec 19 noeuds dans les couches d'entrée et de sortie et de 20 noeuds dans la couche cachée;
— amplitude de 3000;
— fréquence d'échantillonnage de 16000 Hz;

Fréquence	Nombre d'échantil-lons par période	Carré du nombre d'échantillons par période	Nombre d'échantillons proposés par période	Carré du nombre d'échantillons proposé	Numéro du signal
3000	5.33	28.44	38	1444	1
3250	4.92	24.23	38	1444	2
3450	4.64	21.50	38	1444	3
3650	4.38	19.21	38	1444	4
3850	4.15	17.27	19	361	5
4050	3.95	15.60	19	361	6
4250	3.76	14.17	19	361	7
4450	3.59	12.92	19	361	8
4650	3.44	11.83	19	361	9
4850	3.29	10.88	19	361	10
5050	3.16	10.038	19	361	11
Total	44.61	186.48	285	8303	****

Tableau 3.2 Nombre d'échantillons établis dans le fichier d'apprentissage.

— normalisation absolue;

— simulation sur une station Sparc 2.

3.1.1.2.1 Résultats

Le modèle filtre bien le signal selon les spécifications imposées. La fréquence de coupure imposée (5050 Hz) est légèrement dépassée. La même remarque est valable en ce qui concerne la fréquence de coupure basse. Dans cette partie, nous avons repris la même méthode utilisée pour le passe-bas.

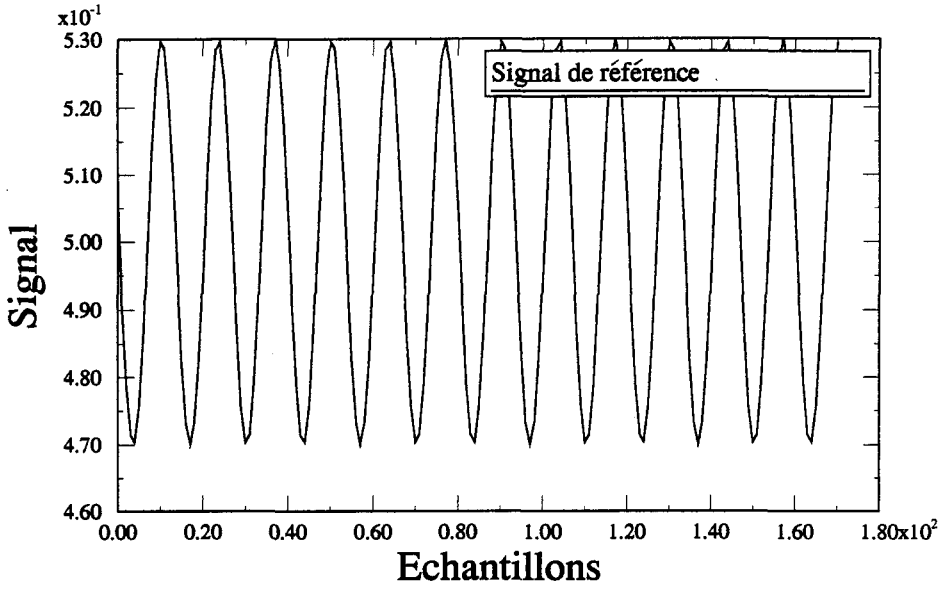

Figure 3.19 Filtre passe-bande (f_{cb} = 3.05 kHz et f_{ch} = 5.05 kHz). Signal fourni à l'entrée du réseau; la fréquence (1200 Hz) n'est pas apprise lors de l'apprentissage et ne se trouve pas dans la plage permise.

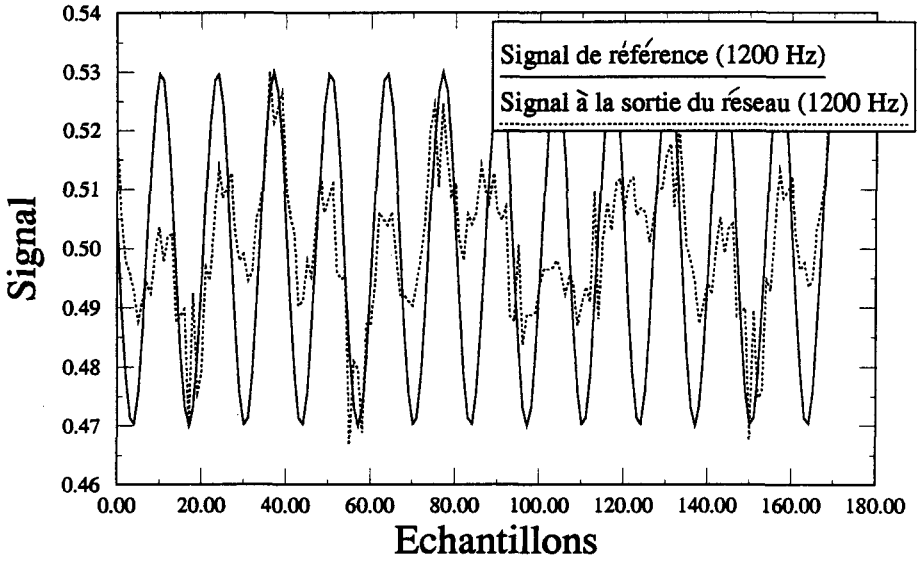

Figure 3.20 Filtre passe-bande (f_{cb} = 3.05 kHz et f_{ch} = 5.05 kHz). Superposition des signaux fournis à l'entrée et à la sortie du réseau; la fréquence (1200 Hz) n'est pas apprise lors de l'apprentissage et ne se trouve pas dans la plage permise.

Les figures 3.19, 3.20 et 3.21 illustrent les performances du réseau vis à vis d'une

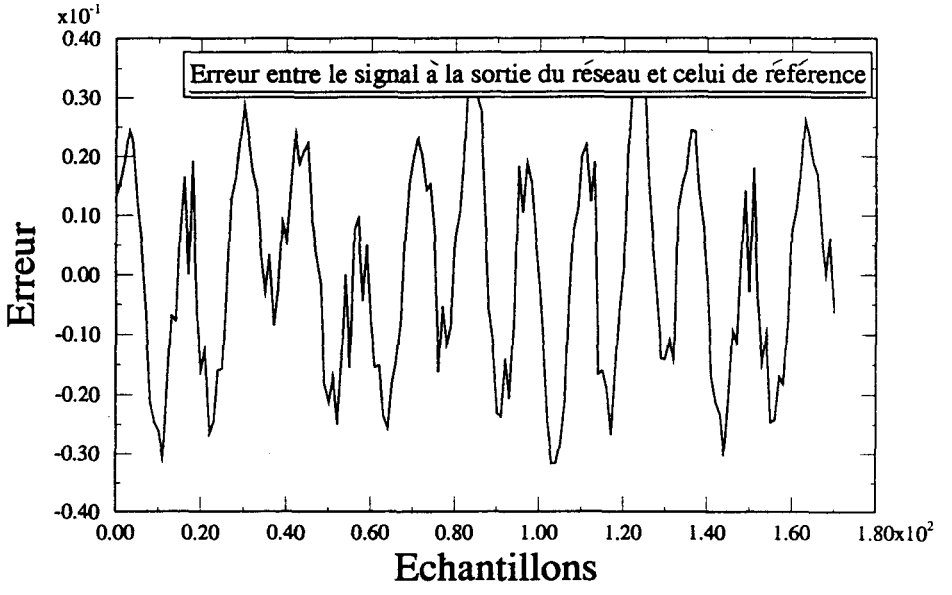

Figure 3.21 Erreur entre le signal fourni à l'entrée et à la sortie du réseau.

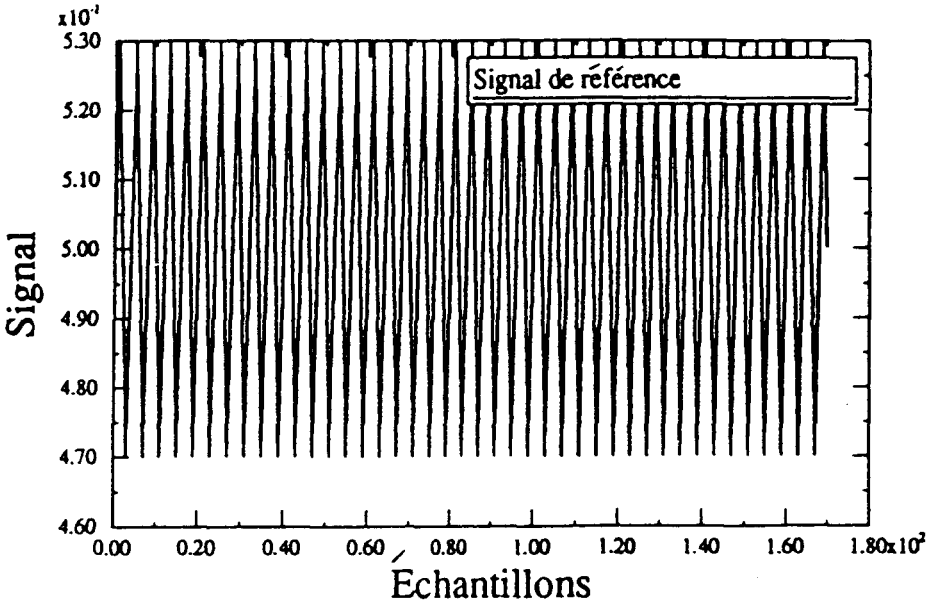

Figure 3.22 Filtre passe-bande (f_{cb} =3.05 kHz et f_{ch} =5.05 kHz)). Signal fourni à l'entrée du réseau; fréquence apprise lors de l'apprentissage (4000 Hz.) et se trouve dans la plage permise.

fréquence (1200 Hz) se trouvant à l'extérieure de la plage permise; les signaux regénérés sont dégradés. Les figures 3.22 et 3.23 illustrent les performances du réseau vis à vis

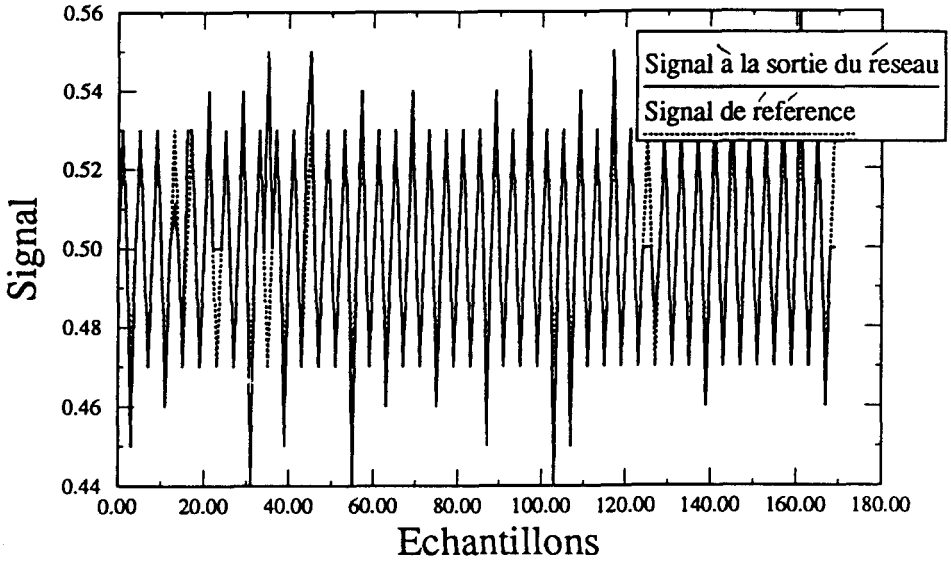

Figure 3.23 Filtrage des basse-fréquences (f_{cb} =3.05 khz et f_{ch} =5.05). Signal regénéré à la sortie du réseau; fréquence apprise lors de l'apprentissage (4000 Hz.)

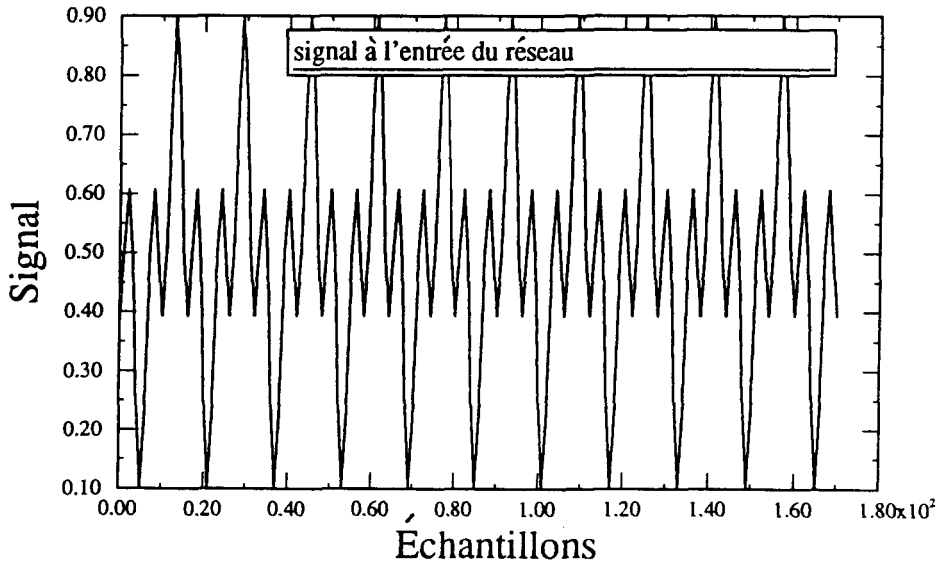

Figure 3.24 Filtre passe-bande; (f_{cb} = 3 kHz et f_{ch} = 5.05 kHz)). Signal fourni à l'entrée du réseau est composé de 2 raies spectrales à 1 et 3 kHz; la fréquence de 1 kHz n'est pas apprise lors de l'apprentissage (se trouve en dehors de la plage permise), par contre celle de 3 kHz se trouve dans la plage permise.

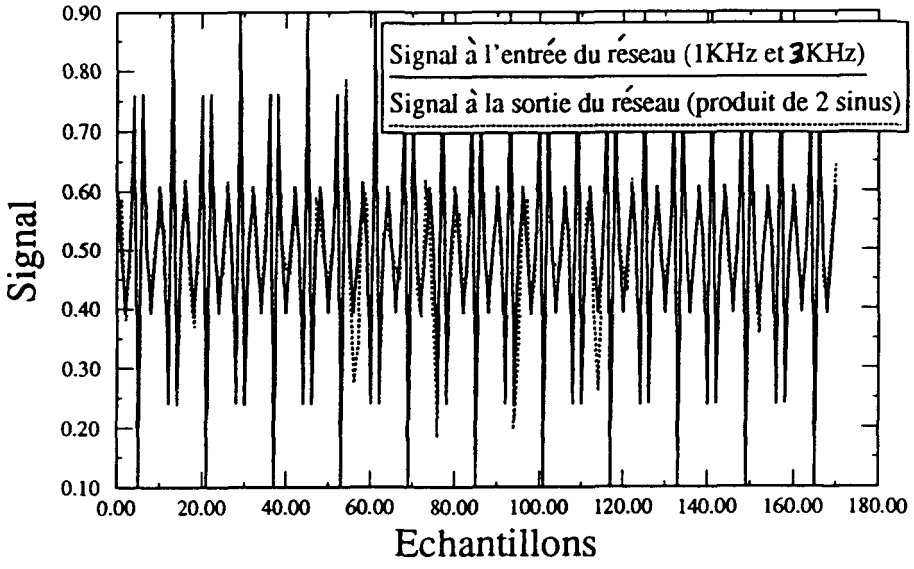

Figure 3.25 Filtrage des basses fréquences (f_{cb} = 3 kHz et f_{ch} = 5.05 kHz)). Superposition des signaux à l'entrée et à la sortie du réseau; fréquence apprise lors de l'apprentissage (3.05 kHz.).

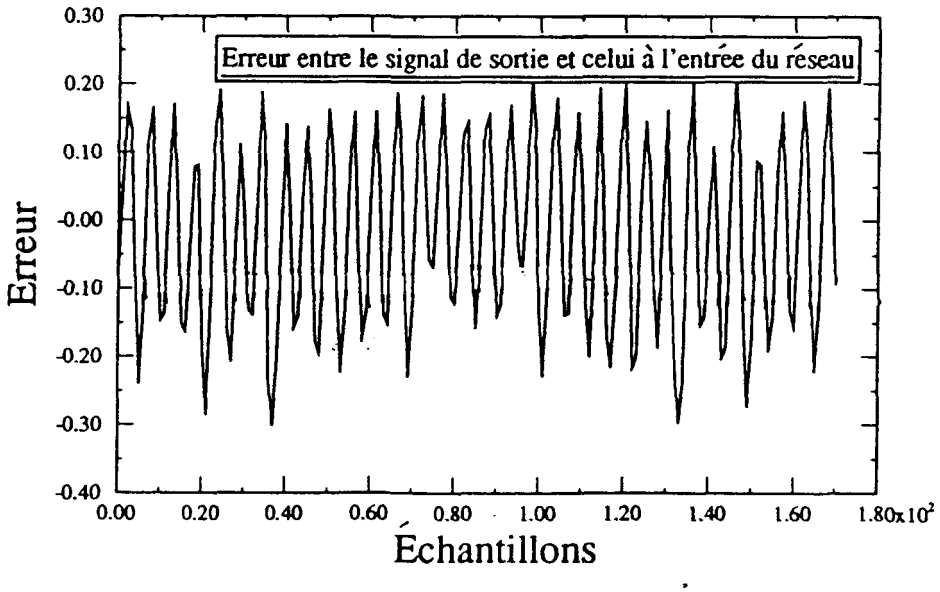

Figure 3.26 Erreur entre le signal fourni à l'entrée et à la sortie du réseau.

une fréquence (4000 Hz) se trouvant dans la plage permise; le réseau donne de bonnes performances . les figures 3.24, 3.25 et 3.26 illustrent les performances du réseau vis à

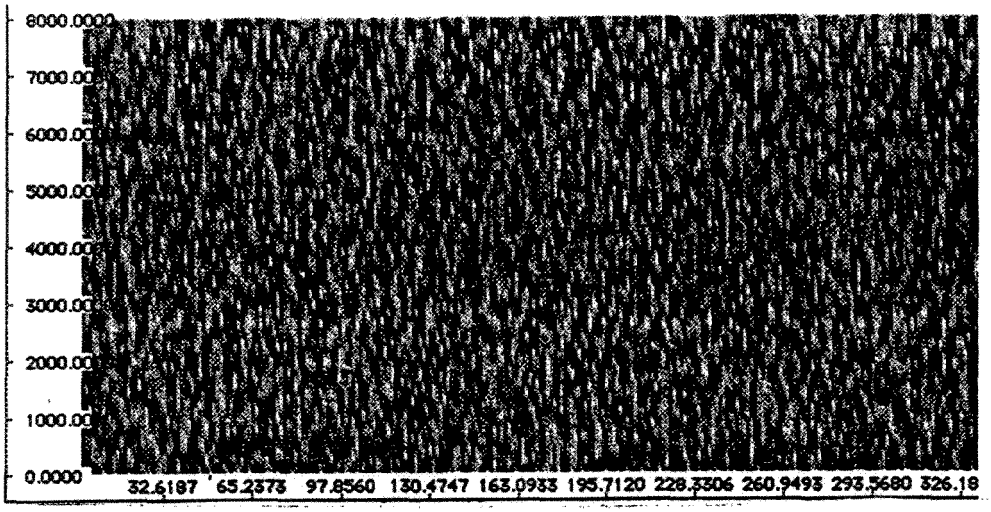

Figure 3.27 Spectrogramme d'un signal de parole bruitée — mot français "la bise" — avec un rapport signal/bruit de −10 dB.- Illustration des performances du filtre passe-bande (f_{cb}= 3.05 kHz et f_{ch}= 5.05 kHz); signal présenté à l'entrée du réseau.

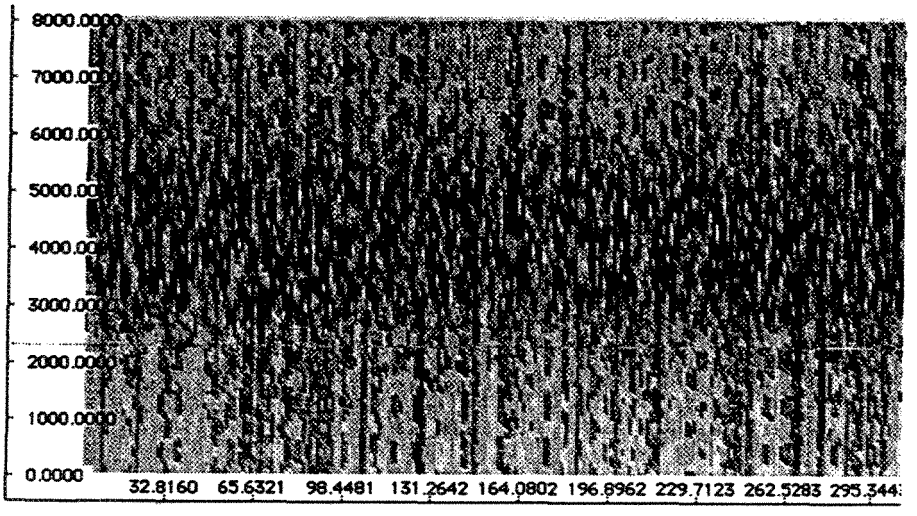

Figure 3.28 Spectrogramme d'un signal de parole bruitée — mot français "la bise" — avec un rapport signal/bruit de —10 dB. Illustration des performances du filtre passe-bande (f_{cb}= 3.05 kHz et f_{ch}= 5.05 kHz); signal obtenu à la sortie du réseau.

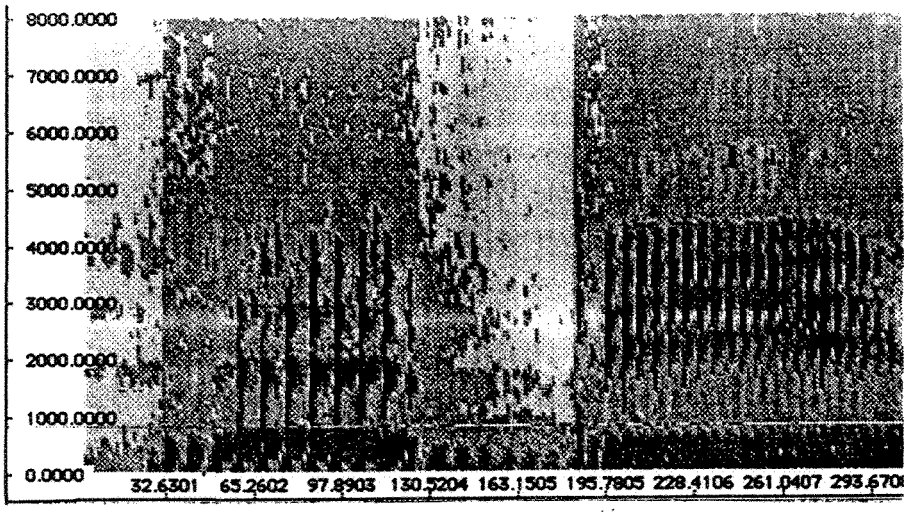

Figure 3.29 Spectrogramme d'un signal de parole non-bruitée "la bise", présenté à l'entrée du réseau. Illustration des performances du filtre réducteur de bruit (passe-bande).

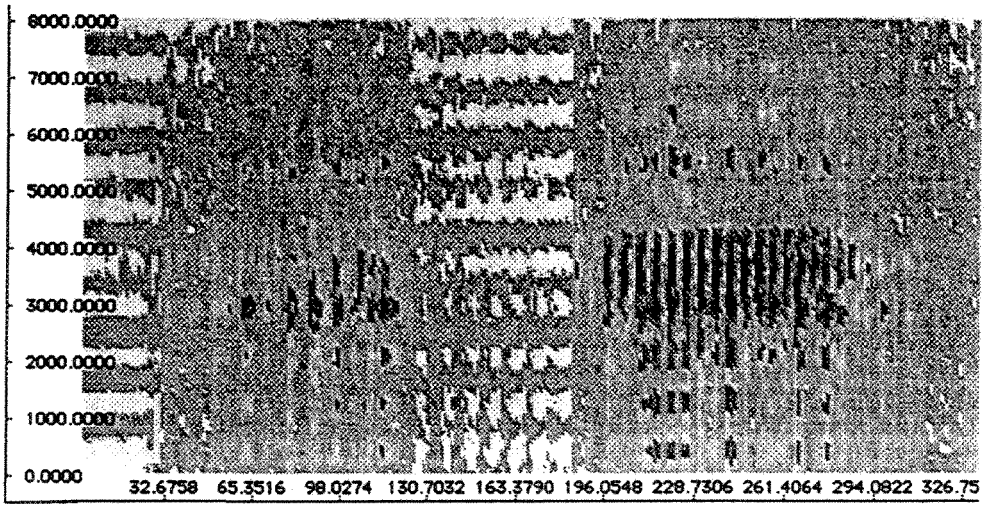

Figure 3.30 Spectrogramme d'un signal de parole non-bruitée, obtenu à la sortie du réseau. Illustration des performances du filtre réducteur de bruit (passe-bande).

vis deux fréquences (1000 et 3000 Hz) dont une se trouvant à l'extérieure de la plage

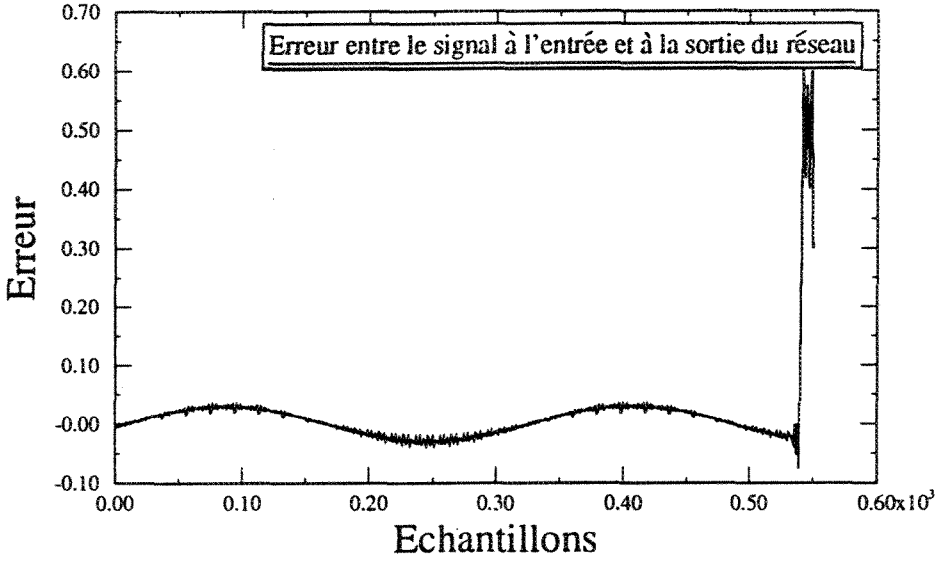

Figure 3.34 Erreur entre le signal à l'entrée et celui à la sortie du réseau.

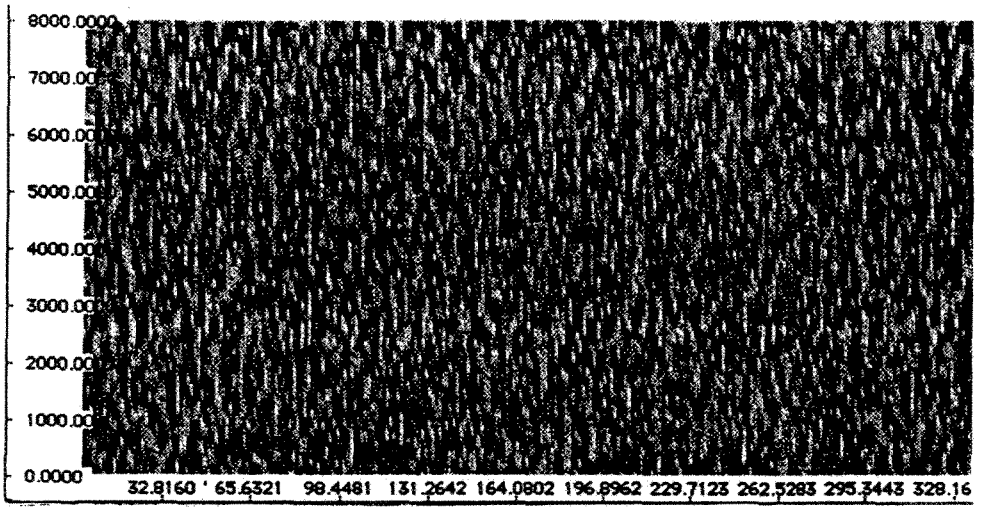

Figure 3.35 **Spectrogramme d'un signal de parole bruitée** — mot français "la bise" — avec un rapport signal/bruit de –10 dB. Illustration des performances du filtre passe-haut (f_{cb}= 5.05 kHz) basé sur l'approche neurale appliqué sur la parole propre. Signal présenté à l'entrée du réseau.

propre. Il présente un problème au niveau de la fréquence de coupure, qui est légèrement

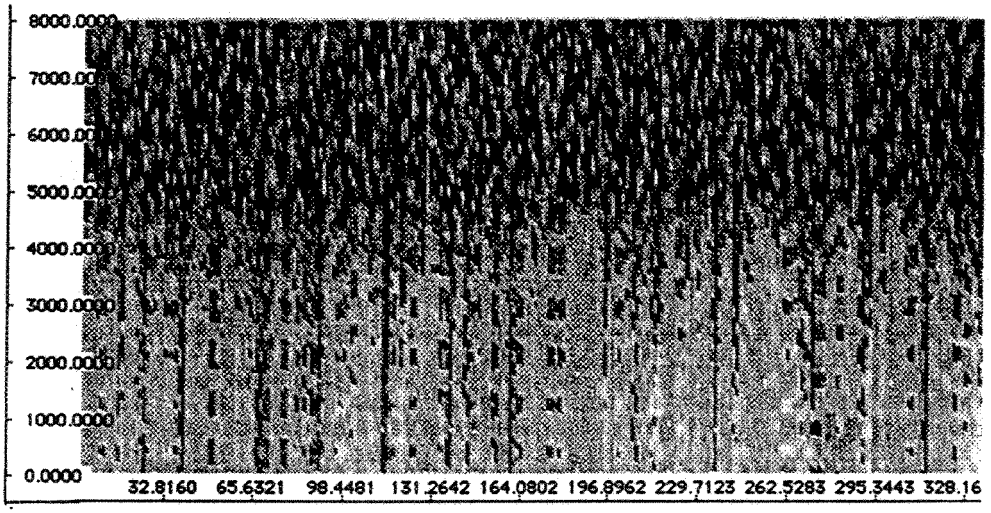

Figure 3.36 **Spectrogramme d'un signal de parole bruitée** — mot français "la bise" — avec un rapport signal/bruit de –10 dB. Illustration des performances du filtre passe-haut (f_{cb}= 5.05 kHz) basé sur l'approche neurale sur la parole propre. Signal obtenu à la sortie du réseau.

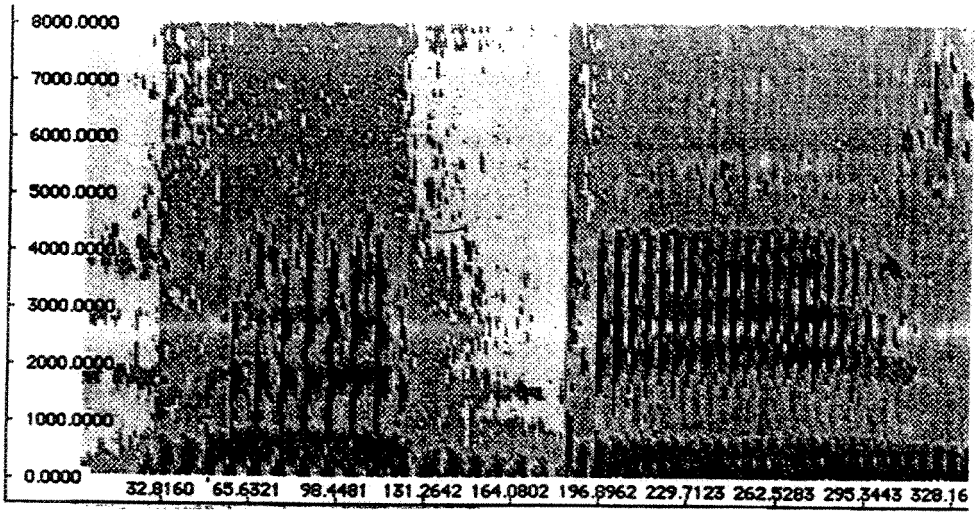

Figure 3.37 Spectrogramme d'un signal de parole non-bruitée "la bise", présenté à l'entrée du réseau. Illustration des performances du filtre réducteur de bruit (passe-haut).

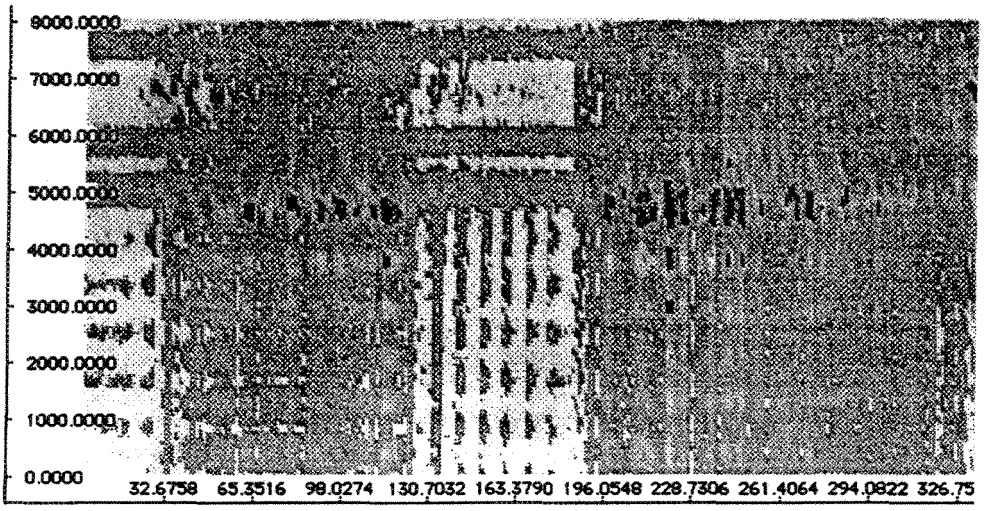

Figure 3.38 Spectrogramme d'un signal de parole non-bruitée, obtenu à la sortie du réseau. Illustration des performances du filtre réducteur de bruit (passe-haut).

déplacée. Mais, ce dernier peut être réglé en ajustant le signal à l'apprentissage. Les figures 3.35 à 3.38 présentant les spectrogrammes des signaux de parole illustrent bien cela. La taille du fichier d'apprentissage est minime comme nous l'avons démontré dans le tableau 3.3.

3.1.1.4 Expérience # 4: filtre coupe-bande

Le but de cette expérience est de tester la possibilité de translater une fréquence avec l'utilisation d'un réseau de neurones. C'est une opération très complexe.

Données de l'expérience:

— données d'apprentissage: 21 signaux sinusoïdaux périodiques d'une amplitude de 3000 et d'une fréquence variant de 50 à 5050 Hz par pas de 200 Hz; les signaux compris entre 450 Hz et 3000 Hz sont translatés à la plage des fréquences comprise entre (450+5050) Hz et (3000+5050 Hz) et ce, pour le signal à la sortie du réseau seulement — signal à l'entrée n'a subi aucun changement;

Fréquence	Nombre d'échantil-lons par période	Carré du nombre d'échantillons par période	Nombre d'échantil-lons proposés par période	Carré du nombre d'échantillons proposé	Numéro du signal
50	320	102400	323	104329	1
250	64	4096	76	5776	2
450	35.55	1264	38	1444	3
3000	5.33	28.4	38	1444	4
3200	5.00	25	19	361	5
3400	4.7	22	19	361	6
3600	4.44	19.71	19	361	7
3800	4.21	17.72	19	361	8
4000	4.00	16.00	19	361	9
4400	3.63	13.22	19	361	10
4600	3.47	12.09	19	361	11
4800	3.33	11.11	19	361	12
5000	3.20	10.24	19	361	13
Total	460.76	107935.49	665	116603	****

Tableau 3.4 Nombre d'échantillons établis dans le fichier d'apprentissage.

— nombre d'échantillons par fréquence est représenté dans le tableau 3.4.

— algorithme d'apprentissage: la rétropropagation rapide;

— architecture du réseau de type propagation-avant, avec 19 noeuds dans les couches d'entrée et de sortie et 20 noeuds dans la couche cachée;

— amplitude utilisée de 3000;

— fréquence d'échantillonnage de 16000 Hz;

— normalisation absolue;

— une simulation sur une station Sparc 2;

Remarque: Les trois premiers signaux ne respectent pas le critère établi, c-à-d faire l'apprentissage sur p^2 échantillons.

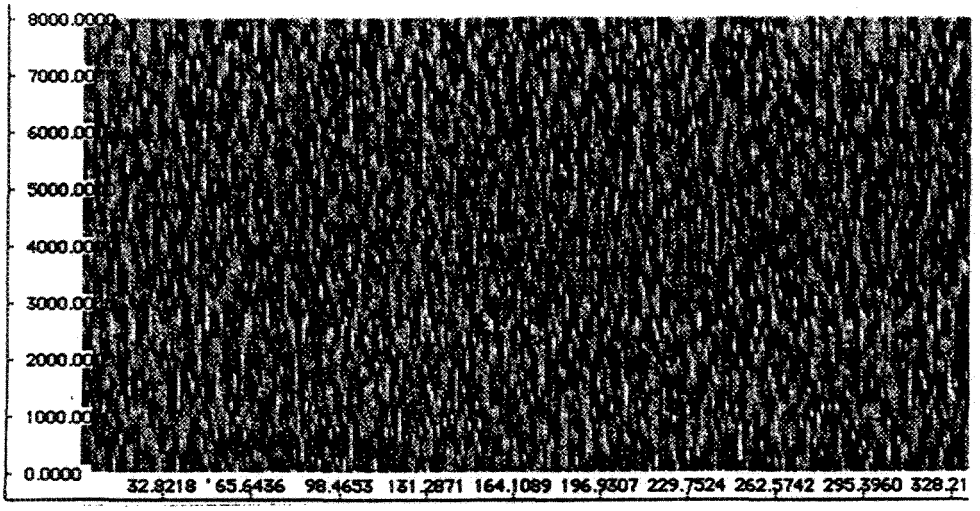

Figure 3.39 Signal à l'entrée du réseau; Signal de parole bruitée (mot prononcé est "la bise".) avec un rapport signal/bruit de —10 dB.

3.1.1.4.1 Résultats

Le modèle filtre un signal bruité avec un rapport signal/bruit de –10 dB. Il laisse passer l'énergie se trouvant dans la première bande, soit entre 50 et 450 Hz et déplace une partie de l'énergie se trouvant entre 450 et 3000 Hz à l'extérieur de cette plage, soit entre 5050 et 8000 Hz. Les performances du modèle sont encore meilleures lorqu'ils sont appliquées à la parole non-bruité, telles qu'illustré dans les graphiques 3.39 à 3.42.

3.1.1.4.2 Interprétation des résultats

Le modèle donne des résultats intéressants, en ce qui concerne le déplacement des fréquences d'une plage à une autre. Il respecte en partie l'objectif visé. Il laisse passer le signal dans la première bande, déplace celui-ci à l'extérieur de la plage permise s'il se trouve entre 450 et 3000 Hz, et laisse passer le signal s'il se situe entre 3000 et 5000 Hz. Les fréquences de coupure ne sont pas respectées (léger dépassement). Il y a de l'énergie dans la plage filtrée. Cela peut résulter d'une discontinuité dans le signal qui se manifeste par un spectre d'énergie large. La quantité d'information déplacée correspondant initiale-

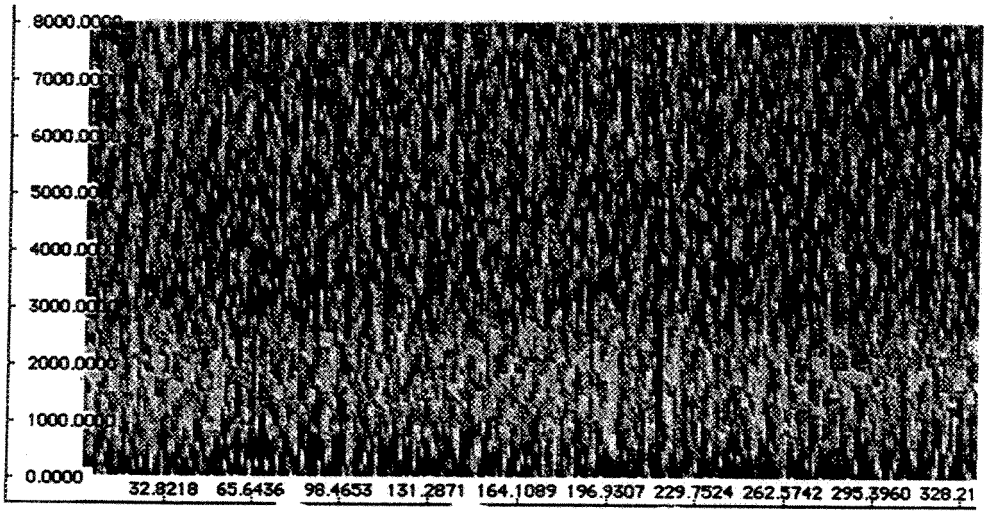

Figure 3.40 Déplacement des fréquences d'une plage (450 à 3000 Hz) à une autre (5050+(450 à 3000 Hz)): application sur de la parole bruitée (-10 dB).

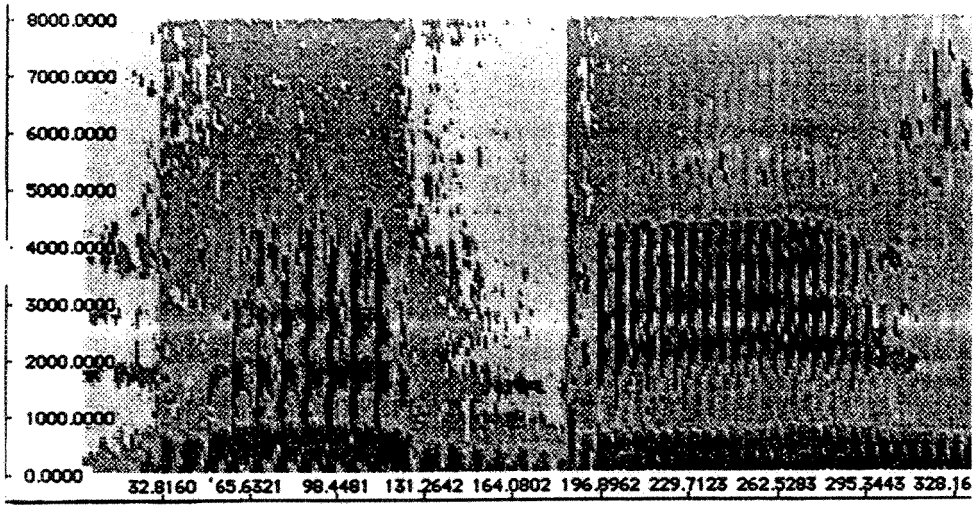

Figure 3.41 Signal à l'entrée du réseau; Signal de parole non-bruitée; mot prononcé est "la bise".

ment à la bande de 450 à 3000 Hz n'est pas intégrale, comme nous l'avons illustré dans la figure 3.41 et 3.42.

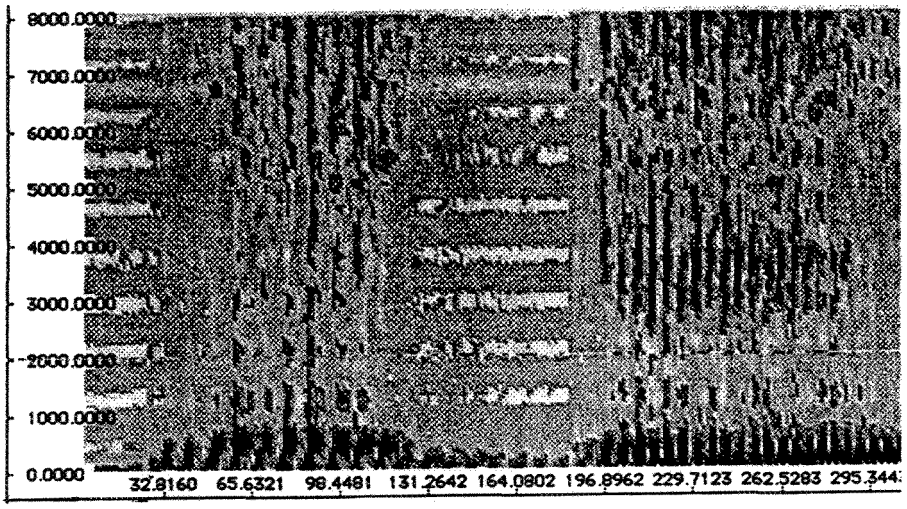

Figure 3.42 Signal regénéré à la sortie du réseau: déplacement des fréquences d'une plage (450 à 3000 Hz) à une autre (5050+(450 à 3000 Hz)): application sur de la parole non-bruitée ("la bise").

En résumé, avec la même architecture de réseau et le même algorithme d'apprentissage, nous pouvons concevoir divers filtres pouvant ressembler à des: passe-bas, passe-bande, passe-haut et coupe bande. Le seul paramètre susceptible de changer est celui du fichier d'apprentissage.

3.2 Comparaison des filtres basés sur les réseaux de neurones et ceux basés sur les méthodes classiques

3.2.1 Filtres classiques: passe-bas

Les filtres classiques sont basés sur des algorithmes mathématiques qui se prêtent bien à la programmation. Les performances du filtre dépendent directement de l'ordre du filtre (FIR et IIR). Le nombre d'opérations mathématiques (addition et multiplication) est fonction de l'ordre du filtre. Les chercheurs ont beaucoup étudié le sujet du filtrage linéaire en vue de résoudre quelques problèmes qui le caractérise (phénomène de Gibbs, etc...). Les filtres classiques ne se prêtent pas à une adaptation; une fois programmé, l'opération est rigide et

permise. Finalement, les figures 3.27 à 3.30 illustrent une application sur la parole non-bruitée et bruitée.

3.1.1.2.2 Interprétation des résultats

Les résultats obtenus sont très concluants; l'apprentissage a été très rapide (285 échantillons). Les performances du modèle sont assez interéssantes. La majeure partie de l'énergie qui passe dans le système se trouve dans la plage spécifiée. Néanmoins, il y a de larges bandes spectrales qui s'étendent sur tout le spectre et peuvent être expliquées par les raisons évoquées précédemment (expérience #1, dans l'interprétation des résultats).

L'application à la parole est très intéressante, comme nous l'avons illustré dans les figures 3.27 à 3.30.

3.1.1.3 Expérience # 3: filtre passe-haut

Cette expérience fait l'étude du passe-haut basé sur l'approche neurale.

Données d'apprentissage :

— 16 signaux sinusoïdaux périodiques d'une amplitude de 3000 et d'une fréquence variant de 5050 Hz à 8000 Hz par intervalle de 200 Hz;
— nombre d'échantillons est représenté dans le tableau 3.3.

— algorithme d'apprentissage: la rétropropagation rapide;
— architecture du réseau de type propagation-avant, avec 19 noeuds dans les couches d'entrée et de sortie et 20 noeuds dans la couche cachée;
— amplitude utilisée de 3000;
— fréquence d'échantillonnage de 16000 Hz.;
— normalisation absolue;
— simulation sur une station Sparc 2;

Fréquence	Nombre d'échantil-lons par période	Carré du nombre d'échantillons par période	Nombre d'échantil-lons proposés par période	Carré du nombre d'échantillons proposé	Numéro du signal
5050	3.16	10.038	19	361	1
5250	3.047	9.28	19	361	2
5450	2.93	8.61	19	361	3
5650	2.83	8.019	19	361	4
5850	2.735	7.48	19	361	5
6050	2.64	6.99	19	361	6
6250	2.56	6.55	19	361	7
6450	2.48	6.15	19	361	8
6650	2.40	5.78	19	361	9
6850	2.33	5.45	19	361	10
7050	2.27	5.15	19	361	11
7250	2.21	4.87	19	361	12
7450	2.14	4.61	19	361	13
7650	2.091	4.37	19	361	14
7850	2.038	4.154	19	361	15
8050	1.9875	3.95	19	361	16
Total	39.84	101.45	304	5776	****

Tableau 3.3 Nombre d'échantillons établis dans le fichier d'apprentissage.

3.1.1.3.1 Résultats

Les performances du modèle sont représentés sur la figures 3.31 pour une fréquence se trouvant dans la plage permise. Par contre, les figures 3.32 à 3.34 illustrent les performances du réseau vis à vis d'une fréquence se trouvant à l'extérieure de la plage spécifiée. Le modèle filtre bien le signal selon les contraintes du filtre imposées. La fréquence de coupure est légèrement dépassée. Il n'y a pas d'énergie dans la plage extérieure à celle spécifiée.

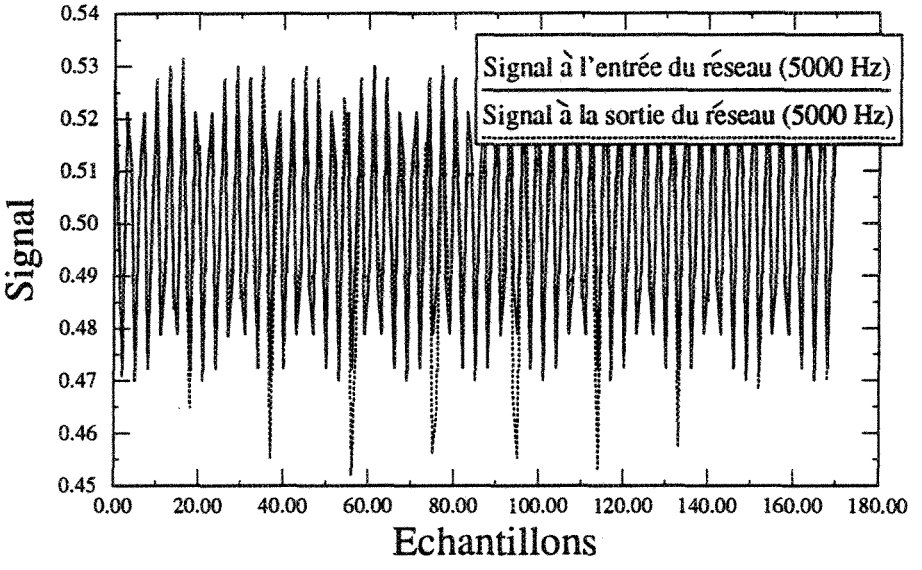

Figure 3.31 filtre passe-haut (f_{cb} = 5 kHz et f_{ch} = 8 kHz); superposition des signaux à l'entrée et à la sortie du réseau; fréquence non apprise lors de l'apprentissage (5000 Hz).

Les figures 3.35 à 3.38 illustrent bien les performances du modèle sur la parole bruitée et encore mieux sur la parole propre, comme nous le montrent les figures 3.37 à 3.38.

3.1.1.3.2 Interprétation des résultats

Les résultats obtenus par ce modèle montrent que le réseau réduit les basses fréquences au détriment des hautes fréquences. La composante basse fréquence est toujours présente mais elle est dominée largement par les hautes fréquences. La suppression des basses fréquences (figures 3.32 à 3.34) montre que le réseau génère une sorte de modulation d'amplitude visible au niveau des maximums; on en déduit que le réseau est sensible au détail concernant l'amplitude.

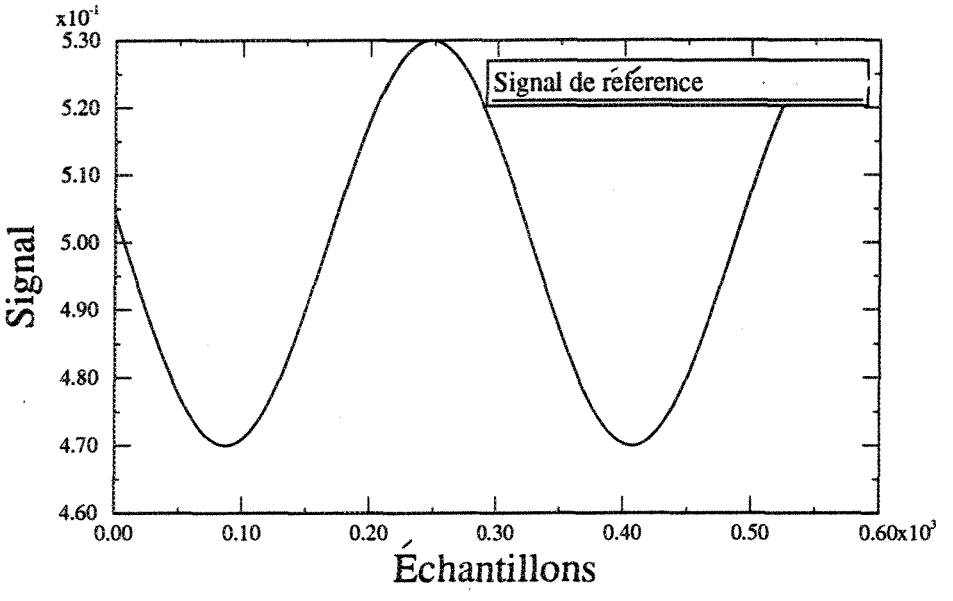

Figure 3.32 Étude du filtre passe-haut avec une fréquence (500 Hz) ne se trouvant pas dans la plage du filtre.

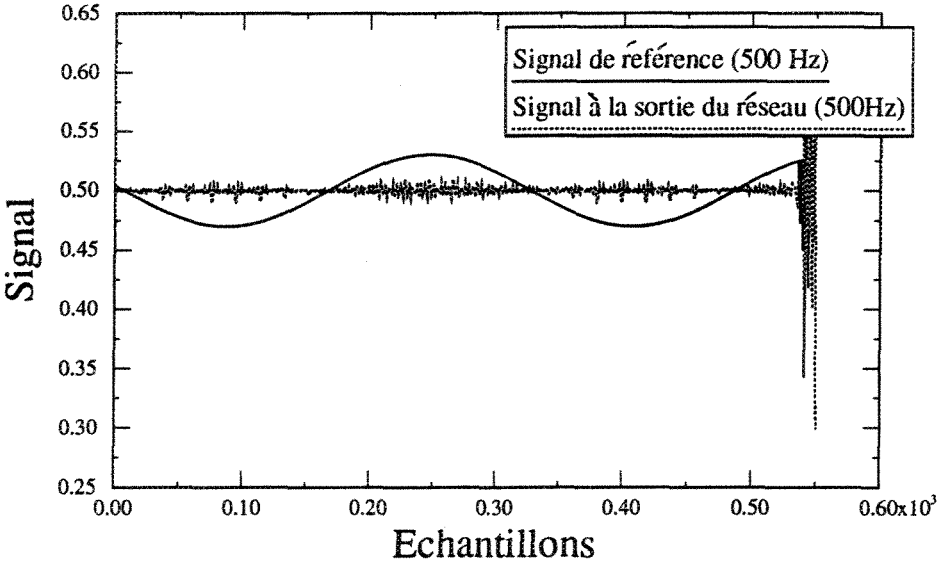

Figure 3.33 Filtrage des basses fréquences (filtre passe-haut, f_{cb} = 5 kHz et f_{ch} = 8 kHz); superposition des signaux à l'entrée et à la sortie du réseau; fréquence non apprise lors de l'apprentissage et se trouvant à l'extérieur de la plage couverte par le filtre. (500 Hz) .

Le modèle performe bien, en tant que filtre passe-haut, sur la parole bruitée et la parole

ne se prête à aucun changement sauf dans le cas des filtres adaptatifs. Le seul paramètre pouvant être modifié est celui de la fréquence de coupure, pour un ordre fixé. Le traitement est purement linéaire. Ces filtres ont l'avantage de ne laisser passer qu'une petite partie du signal en dehors de la bande autorisée. Ils ne sont nullement en mesure de faire une translation (partielle) des fréquences. Cette opération est très complexe dans le domaine fréquentiel. Chaque changement dans le fonctionnement du filtre s'accompagne par une modification de l'algorithme. De plus, compte tenu du caractère linéaire, ce type de modèle de filtre n'est pas en mesure d'ajouter de l'information dans le signal filtré.

Nous avons constaté que nous ne pouvons pas avoir explicitement la fonction de transfert du réseau, ni même la caractéristique exacte. Ceci s'explique entre autre, par le fait que les filtres conçus sont non-linéaires et nous ne pouvons y associer des caractéristiques spécifiquement liées aux filtres linéaires.

3.2.1.1 Expérience # 1: comparaison des spectrogrammes des signaux filtrés par un réseau de neurones (agissant comme un filtre passe-bas) et par un filtre classique passe-bas

Données de l'expérience:

— signal périodique ayant une amplitude de 3000 composé de 2 raies spectrales de 2 kHz et 4 kHz;
— fréquence de coupure basse (du filtre classique) de 2 kHz;
— réseau de neurones ayant appris sur un signal d'amplitude fixe et de fréquence variant de 50 Hz à 2000 Hz par intervalles de 200 Hz.

3.2.1.1.2 Résultats

Les résultats sont illustrés dans les 4 graphiques suivants (3.43 à 3.46)

Le graphique 3.43 illustre les performances du réseau de neurones ayant une fréquence maximale (extrême) de 2050 Hz (signal d'apprentissage : plage couverte de 50 à 2050 Hz) et agissant comme un filtre passe-bas sur un signal pur ;

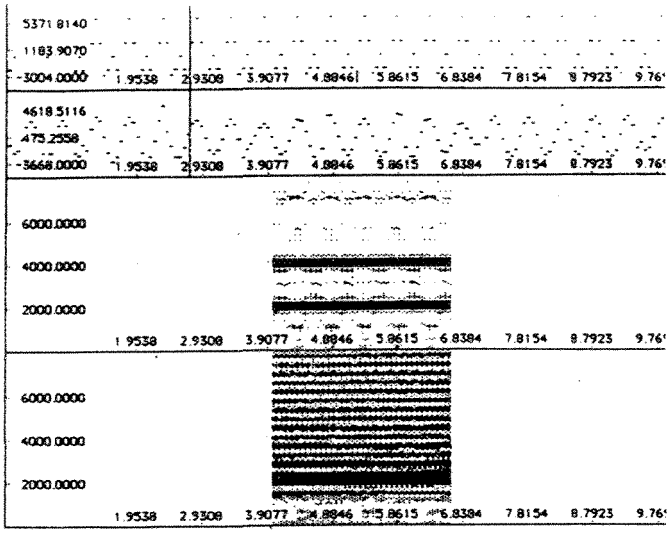

Figure 3.43 Performances du réseau: filtre passe-bas; application avec un signal pur (2 raies spectrales à 2 Khz et 4 Khz).

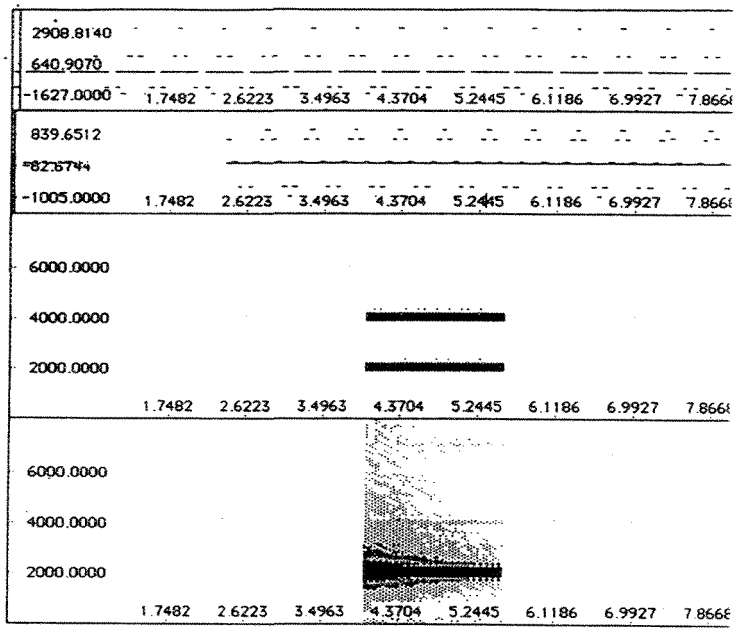

Figure 3.46 Performances du filtre passe-bas classique; application avec un signal pur (2 raies spectrales à 2 Khz et 4 Khz).

Le graphique 3.48 illustre les performances du filtre classique passe-bas ayant une fréquence de coupure de 2050 Hz et agissant sur un signal de parole.

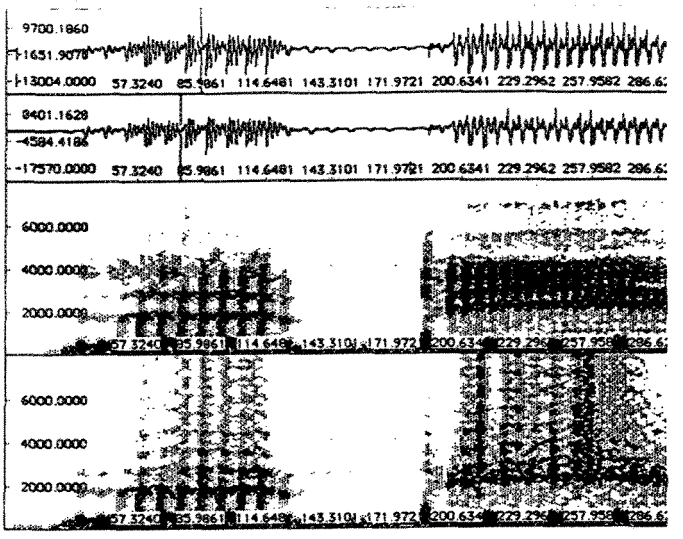

Figure 3.45 Performances du réseau: filtre passe-bas; application avec un signal de parole.

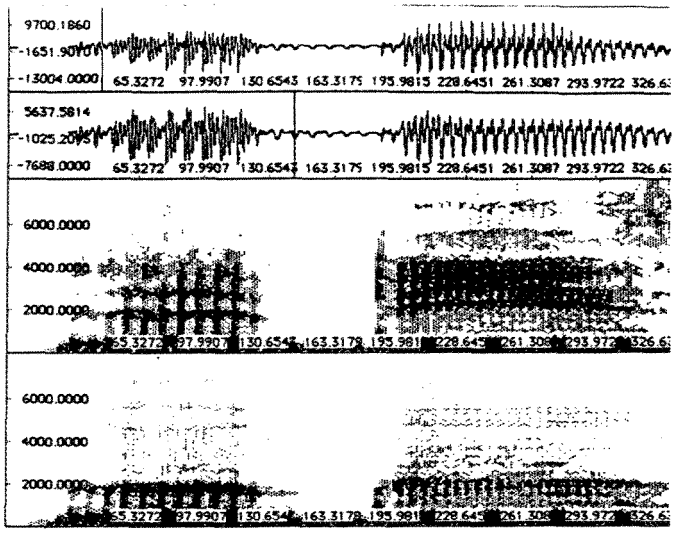

Figure 3.46 Performances du filtre classique passe-bas; application avec un signal de parole.

Le graphique 3.45 illustre les performances du réseau de neurones agissant comme un filtre passe-bas sur un signal de parole ayant une fréquence maximale (extrême) de 2050 Hz (signal d'apprentissage : plage couverte de 50 à 2050 Hz);

Le graphique 3.46 illustre les performances du filtre classique passe-bas ayant une fréquence de coupure de 2050 Hz et agissant sur un signal de parole.

3.2.1.1.3 Interprétation des résultats

Le graphique 3.43 nous montre que le réseau a coupé la fréquence de 4000 Hz, mais il a laissé passer celle de 2000 Hz. La bande spectrale aux alentours de 2000 Hz est un peu plus large à la sortie qu'à l'entrée. Il a ajouté un léger bruit dans le spectre. Nous remarquons la présence de plusieurs raies spectrales au niveau de 2500 Hz et de 3500 Hz.

Le graphique 3.44 nous montre que le filtre classique a coupé le spectre des fréquences à partir de 2000 Hz. Par contre, il a généré une dégradation au niveau spectral (ajout de l'energie entre 50 Hz et ~ 2000 Hz).

L'application sur un signal de la parole est quasiment identique lorsqu'il s'agit de tenir compte uniquement des spectres d'amplitude.

Il est à noter que la reproduction par photocopie peut donner l'impression qu'il y a plus d'énergie à l'extérieure de la plage permise. En effet, il y a un léger bruit qui se trouve être accentué par la reproduction (photocopieuse).

3.3 Filtrage d'un signal sinusoïdal bruité

Dans cette section, nous analyserons les performances d'un réseau réducteur de bruit sur un signal périodique bruité avec un rapport signal/bruit pouvant atteindre –10 dB.

3.3.1 Expérience # 1: filtrage d'un bruit blanc

Donnée de l'apprentissage:

— signal à l'entrée, périodique de fréquence 1200 Hz et d'amplitude 3000. Ce signal est bruité avec un rapport signal/ bruit de 0 dB;
— signal à la sortie, périodique de fréquence 1200 Hz et d'amplitude 3000. Ce signal n'a pas subi de changement;
— réseau à propagation-avant;

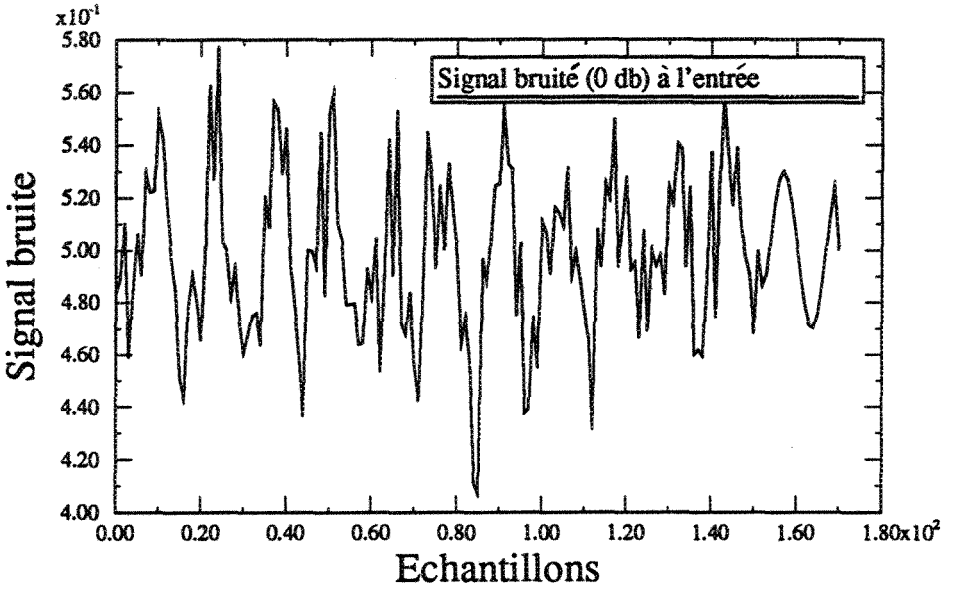

Figure 3.47 Illustration d'un signal bruité à l'entrée du réseau; celui-ci est bruité avec un bruit blanc avec un rapport signal/bruit de 0 dB.

— algorithme d'apprentissage: la rétropropagation rapide;

— 19 noeuds dans la couche d'entrée ;

— 20 noeuds dans la couche d'entrée;

— 19 noeuds dans la couche de sortie;

— nombre d'itérations de 20000 (1052 cycles);

— type de bruit: bruit Gaussien, blanc.

3.3.1.2 Résultats

Les quatres graphiques suivants représentent les performances du modèle de réseau de neurones concernant le filtrage d'un signal sinusoïdal bruité avec des rapports signaux/bruit respectivement de 0 dB et de –10 dB. Nous montrons le signal à l'entrée du réseau et à la sortie de celui-ci (figures 3.47 à 3.54).

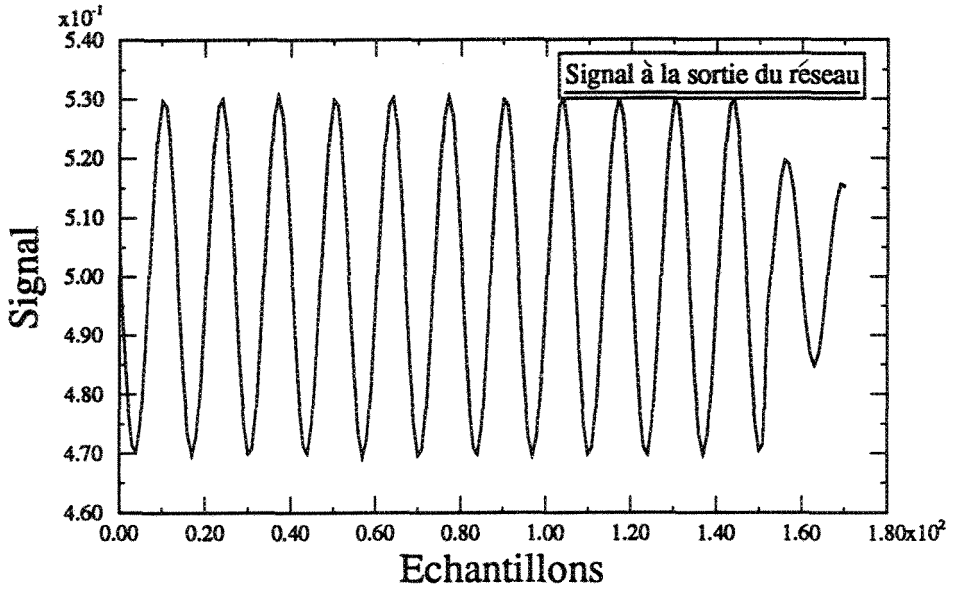

Figure 3.48 Sortie du réseau pour un signal bruité (S/N = 0 dB).

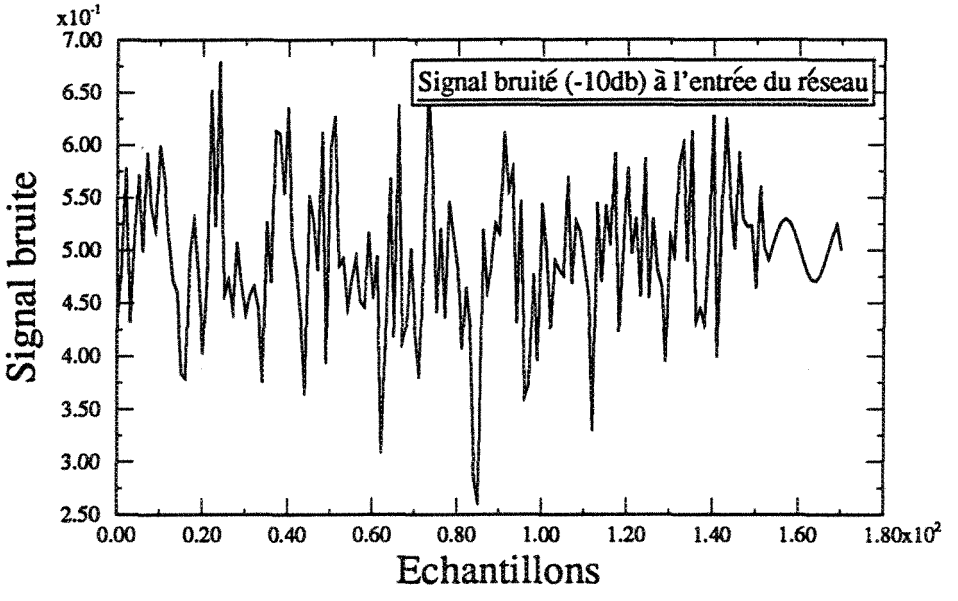

Figure 3.49 Illustration d'un signal bruité à l'entrée du réseau; signal bruité avec un même bruit gaussien qu'à l'apprentissage (rapport signal/bruit de –10 dB).

3.3.1.3 Interprétation des résultats

Les résultats obtenus démontrent les capacités du réseau à filtrer un signal sinusoïdal

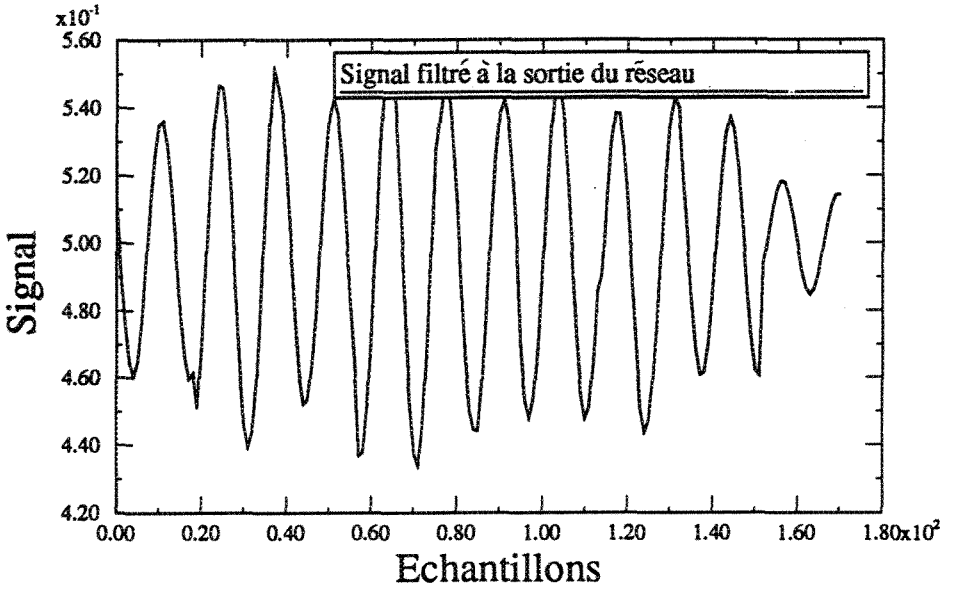

Figure 3.50 Sortie du réseau pour un signal bruité (S/N = —10 dB).

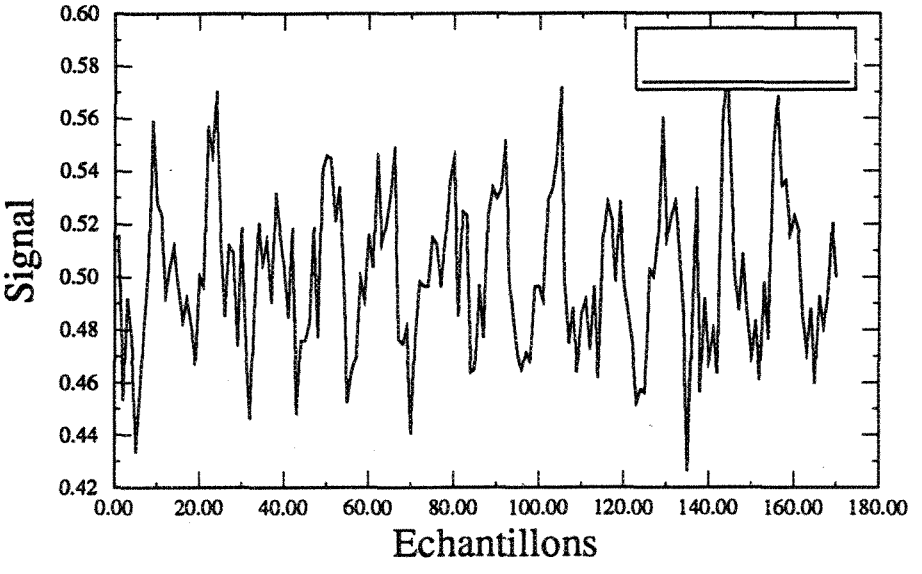

Figure 3.51 Illustration d'un signal bruité à l'entrée du réseau; celui-ci est bruité avec un bruit gaussien — non appris — (différent du précédent) avec un rapport signal/bruit de 0 dB.

périodique. Le réseau a appris sur un signal périodique bruité avec un rapport signal/bruit de 0 dB. Avec ce niveau de bruit, le réseau regénère un signal bien filtré, car l'apprentissage

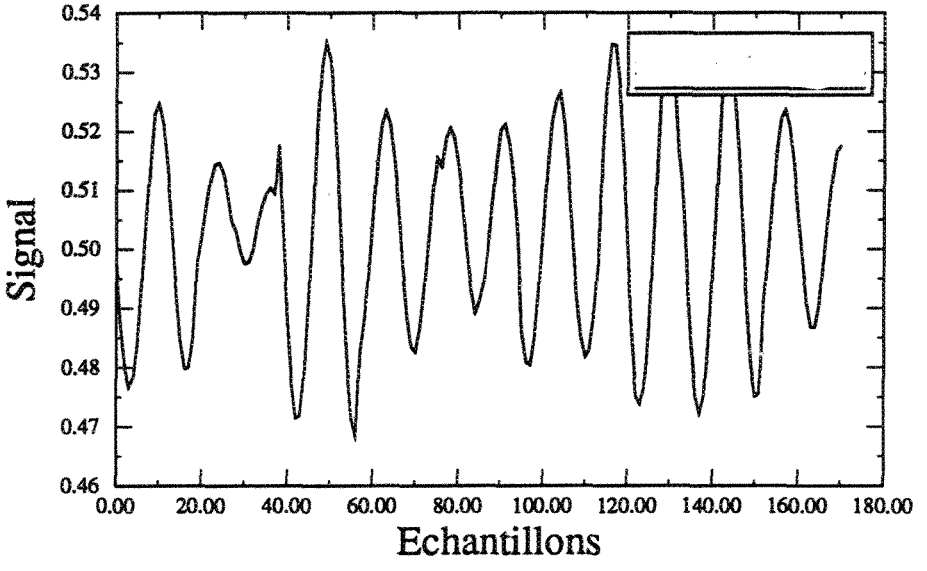

Figure 3.52 Sortie du réseau pour un signal bruité (S/N = —10 dB).

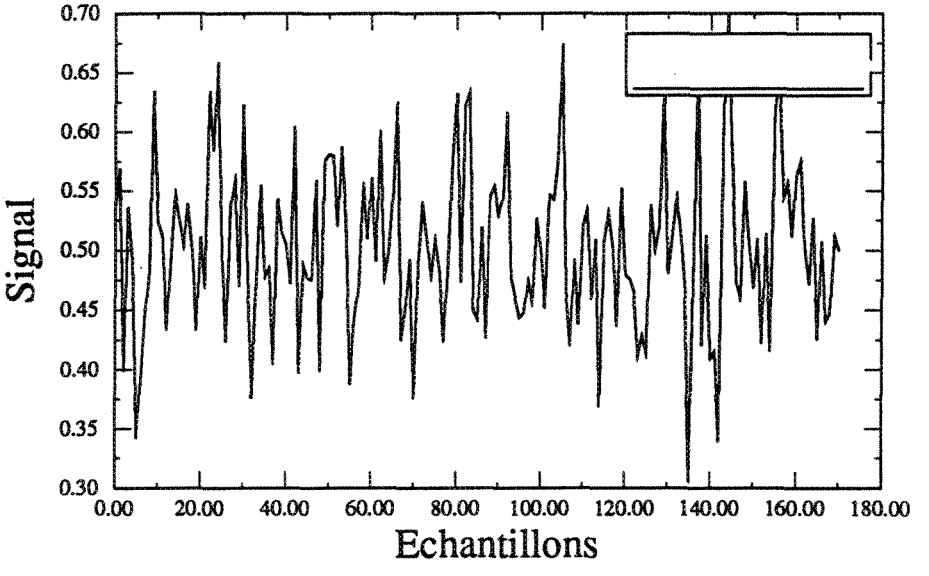

Figure 3.53 Illustration d'un signal bruité à l'entrée du réseau; celui-ci est bruité avec un *bruit blanc — non appris — (différent du précédent) avec un rapport signal/bruit de –10 dB.*

est fait sur ce dernier (figures 3.47 à 3.48). Cependant, à –10 dB, le réseau nous regénère un signal sinusoïdal déformé en amplitude (figures 3.49 à 3.50), mais ayant une fréquence

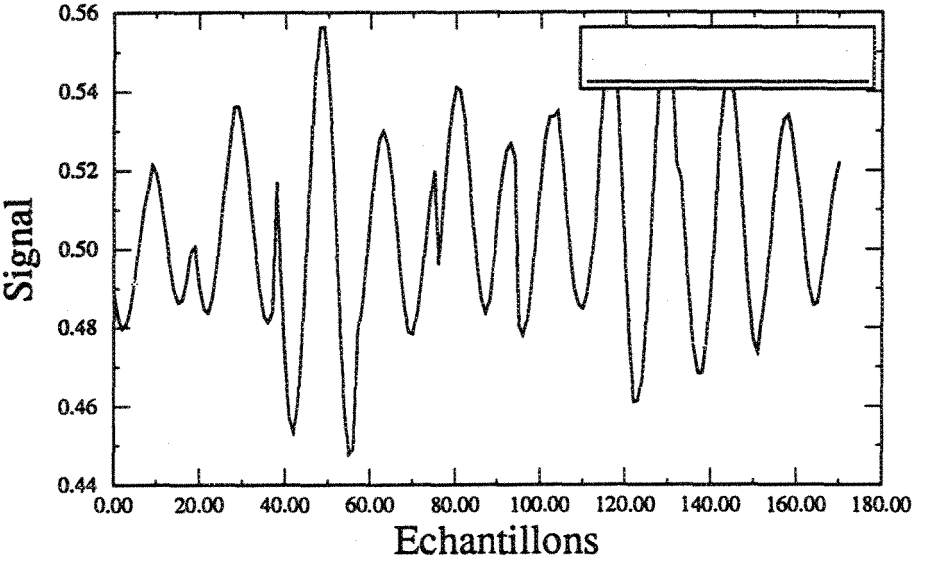

Figure 3.54 Sortie du réseau pour un signal bruité (bruit différent (-10 dB).

correcte. Les performances du réseau sont quand même intéressantes, en qui concerne le filtrage du bruit.

Nous avons testé les performances du réseau sur un autre bruit (complètement nouveau —gaussien). Les résultats du modèle sont représentés sur les figures (3.51 et 3.54). Nous constatons que le réseau a nettement plus de difficultés lorsqu'il s'agit d'un bruit différent du bruit gaussien original.

3.4 Autres approches de filtrage — apprentissage sur des données de parole —

Aux sections précédentes, nous avons vu que la conception d'un filtre passe-bas est possible. Maintenant, voyons si nous pouvons concevoir un filtre non-linéaire pour la parole bruitée. Pour effectuer ce travail , nous avons évité de réutiliser les mêmes méthodes. Nous avons dû réduire la taille des données d'apprentissage et la structure de l'architecture pour éventuellement minimiser le nombre d'opérations de calcul et le temps d'apprentissage tout en ayant une bonne convergence.

3.4.1 Expérience # 1: filtrage de la parole

Nous sommes conscients du fait que la taille des bases de données de parole se mesure en millions de bytes. Elles nécessitent un temps de traitement élevé. L'un des objectifs visé dans ce travail est de réduire la taille des données pour éventuellement minimiser le temps d'apprentissage. Pour ce faire, nous avons choisi une architecture de réseau de neurones assez réduite (49 noeuds — 780 connexions).

Cette expérience consiste à étudier le filtrage d'un signal de parole bruitée par un réseau de neurones. Les fichiers d'apprentissage sont constitués d'un signal de parole bruitée - avec un rapport signal/bruit de 0 dB- à l'entrée du réseau et d'un signal de parole propre à la sortie du réseau.

Données d'apprentissage:

— 107 756 échantillons dans le fichier d'apprentissage (portion du mot "labise");
— signal de parole bruitée avec du bruit gaussien —artificiel — à l'entrée du réseau (le rapport signal/bruit est de 0 dB);
— signal de parole propre à la sortie du réseau;
— fréquence d'échantillonnage de 16 kHz;
— algorithme d'apprentissage: la rétropropagation rapide;
— architecture composée d'une structure à propagation-avant avec 19 noeuds à l'entrée et à la sortie et 10 noeuds dans la couche cachée;
— signaux utilisés lors du rappel sont complétement différents de ceux utilisés dans l'apprentissage (locuteurs masculin et féminin); le bruit utilisé lors du rappel est différent de celui utilisé dans l'apprentissage.

3.4.1.1 Résultats

Après le passage du signal bruité (mot "la bise", S/N= 0 dB) dans le réseau, nous retrouvons un signal filtré . Le réseau coupe au niveau des hautes fréquences tel qu'illustré

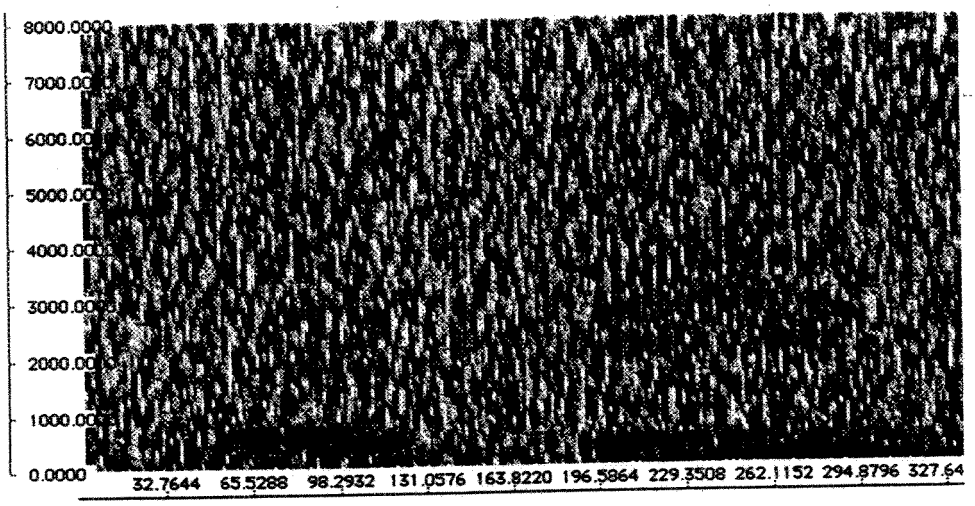

Figure 3.55 Performances du réseau: application avec un signal de parole bruitée; signal présenté (parole bruitée "la bise" avec S/B= -10 dB) à l'entrée du réseau de neurones.

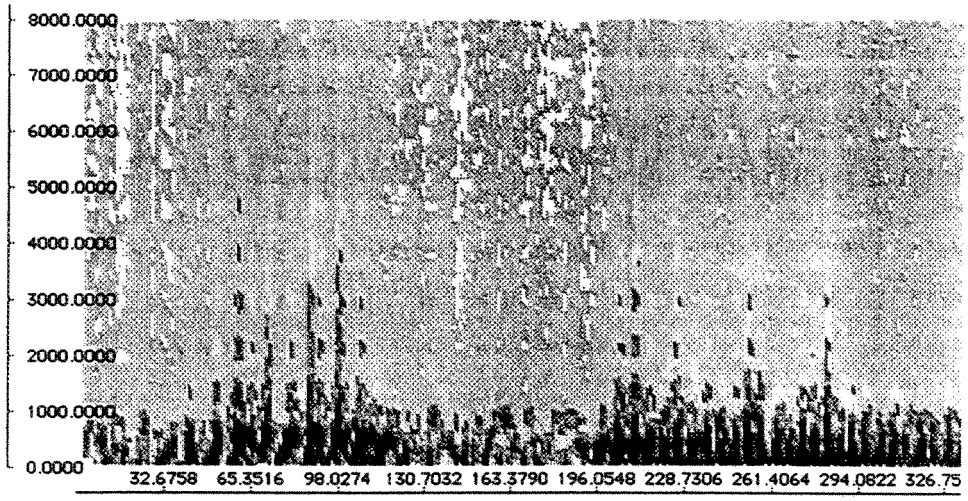

Figure 3.56 Performances du réseau: application avec un signal de parole bruitée; signal regénéré (parole non-bruitée "la bise") à la sortie du réseau de neurones.

par les figures 3.55, 3.56 et 3.57.

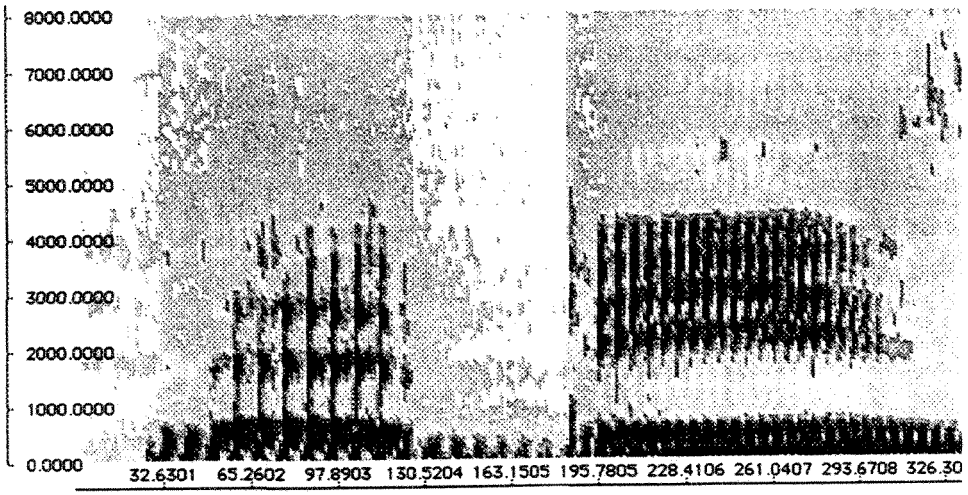

Figure 3.57 Performances du réseau: application avec un signal de parole bruitée; signal de référence (parole non-bruitée "la bise") du réseau de neurones.

3.4.1.1.1 Interprétation des résultats

Effectivement, le réseau présente de bonnes performances lors de l'audition. Nous percevons nettement le signal de parole. Cependant, tout ce que le réseau de neurones laisse passer se trouve dans la plage des basses fréquences, tandis que les hautes fréquences sont coupées. Ceci s'explique par le fait que le réseau confond le signal de parole (surtout les composantes hautes fréquences du signal) et le bruit. Cependant, il est intéressant de constater qu'il coupe le bruit des basses-fréquences et ne laisse passer que le signal de parole. Pourquoi ? En analysant les données d'apprentissage, on s'aperçoit en effet qu'une bonne partie de l'énergie du signal de parole — au niveau de l'amplitude - se situe au niveau des voyelles. De plus, la majeure partie de l'énergie se trouve effectivement au niveau du premier et du deuxième formant. Telle est la principale raison des résultats obtenus. Le réseau a réussi à apprendre cette relation intrinsèque et, bien sûr, tout ce qui se produira par la suite dépendra de ce qu'il aura appris.

Nous avons essayé de reproduire l'expérience avec d'autres données de parole bruitée (avec différents rapports signaux à bruit: –10, 0, 3, 6, et 10 dB); les résultats sont presque identiques. Ils s'apparentent beaucoup aux performances d'un filtre passe-bas ou passe-bande (classique). Dès qu'il y a apprentissage sur des données bruitées, le réseau éprouve des difficultés de convergence.

3.4.2 Expérience # 2: filtrage de la parole; apprentissage sur des portions de période des voyelles du français (/a/, /e/, /i/, /o/ et /ų/)

Dans cette expérience, l'analyse des signaux des voyelles du français a révélé qu'il s'agissait d'un signal pseudo-périodique. À la limite, nous pouvons le considérer comme périodique. Alors, en nous basant sur cette analyse, nous avons extrait trois périodes de chaque (signal d'un locuteur masculin J.D) voyelle et nous les avons reproduites afin d'avoir un seul signal respectant ainsi le critère de p^2. La démarche est effectuée pour les voyelles du français: /a/, /e/, /i/, /o/ et /ų/.

Une fois le fichier constitué, nous l'avons bruité avec un bruit gaussien — un rapport signal/bruit de 0 dB.

Données d'apprentissage:

— 790 échantillons (5 voyelles du français, même locteur);

— signal à l'entrée du réseau (composé par un signal bruité avec un rapport signal/bruit de 0 dB;

— signal à la sortie du réseau (composé par un signal propre);.

— fréquence d'échantillonnage de 16 kHz;

— algorithme d'apprentissage: la rétropropagation rapide;

— architecture à propagation-avant avec 19 noeuds à l'entrée et à la sortie et 60 noeuds dans la couche cachée;

— signaux utilisés lors du rappel sont complètement différents de ceux utilisés dans l'apprentissage (locuteurs masculin et féminin); le bruit utilisé lors du rappel est différent de celui utilisé dans l'apprentissage.

3.4.2.1 Résultats

Les résultats obtenus (avec d'autres locuteurs — masculin et féminin; même type de bruit, mais ce dernier est différent de celui utilisé dans l'apprentissage) sont identiques à ceux de l'expérience précédente où le réseau avait déjà beaucoup de difficultés à converger.

Nous pouvons dire que le fait d'introduire le bruit rend la convergence difficile. Cependant, le réseau a réussi à filtrer le signal bruité.

3.4.3 Expérience # 3: filtrage de la parole; apprentissage sur des signaux purs bruités

Dans cette expérience, les conditions expérimentales sont identiques à celles déjà présentées. Le seul paramètre modifé est le fichier d'apprentissage. Nous avons utilisé des sinusoïdes pures couvrant tout le spectre de la parole (de 50 à 5050 Hz) et les avons bruitées avec un bruit gaussien blanc. Le rapport signal/bruit est fixé à 0 dB, 3dB et 10 dB.

Les résultats obtenus s'apparentent à ceux d'un filtre passe-bas. Néanmoins, le réseau a filtré le signal et le niveau du bruit a baissé. La convergence de l'algorithme est très difficile.

Nous avons essayé d'introduire des signaux purs propres à l'entrée du fichier d'apprentissage. Il s'est avéré que ce changement entraine le passage du bruit à la sortie du réseau.

Le fait de changer le rapport signal/bruit à des répercussions négatives sur les performances du filtre.

3.4.4 Synthèse des expériences 1, 2 et 3

Nous pouvons dire que quel que soit le fichier d'apprentissage, le fait d'introduire le bruit dans le fichier à l'apprentissage cause des difficultés de convergence. Les performances sont comparables à celles du filtre passe-bas (passe-bas basé sur l'approche neurale).

L'un des avantages de cette technique est que le réseau filtre le bruit basses fréquences et ne laisse passer que le signal.

L'expérimentation illustre bien que l'apprentissage avec des données bruitées ne donne pas les résultats souhaités. Mais il y a toujours une possibilité de contourner ce problème en agissant au niveau du choix des paramètres lors de l'apprentissage. Cela a pour conséquence de restreindre l'application du filtre.

Nous avons constaté que la méthode qui consiste à apprendre sur des portions de voyelles propres a donné des résultats similaires à ceux obtenus en effectuant l'apprentissage sur des signaux purs (50 Hz à 5050 Hz). Cependant, le réseau ayant appris sur des portions de voyelles génère une dégradation au niveau de l'amplitude. Cette constatation s'explique par le fait que le réseau ayant appris sur des signaux purs (amplitude fixe), a appris une seule amplitude (référence fixe) et par conséquent fait une bonne généralisation au niveau de cette dernière. Par contre, le réseau ayant appris sur des morceaux de voyelles, a appris sur différents niveaux d'amplitudes (amplitude variable); ce qui nécessite plus de noeuds dans la couche cachée et plus de temps de convergence. C'est pour cela qu'il génère des dégradation au niveau de l'ampilude

3.5 Analyse et discussion

Dans ce chapitre, nous avons vu les résultats des simulations et des expériences effectuées sur les possibilités de filtrage d'un réseau de neurones. Nous avons mis en pratique les résultats observés, déduits du chapitre précédent, particulièrement la généralisation de l'amplitude et la généralisation des fréquences. Ces résultats permettent la conception de filtres basés sur des réseaux de neurones pour traiter l'information dans une plage de fréquences particulière (filtre passe-bas, passe-haut, passe-bande et coupe-bande). Les expériences ont montré qu'il est possible de concevoir ces genres de filtre assez facilement. Il suffit de bien choisir les données de l'apprentissage, car ce sont ces données qui régissent les performances du filtre. La comparaison avec les filtres classiques mon-

tre que les réseaux de neurones peuvent effectivement réaliser les fonctions du filtrage, mais de manière moins satisfaisante que les filtres classiques. Nous voulons dire par là, que les réseaux de neurones peuvent ajouter certaines composantes fréquentielles, même à l'extérieur de la plage du filtre. Ces composantes peuvent être bénéfiques pour le traitement s'il s'agit de transmission de la parole par téléphonie (amélioration auditive), mais peuvent aussi être destructrices s'il s'agit de transmission des données binaires sur une ligne de transmission. Mais ce phénomène est largement régit par les données du fichier d'apprentissage. Il peut être introduit par la forme du signal mis dans le fichier, tout comme il peut être contrôlé par le réglage de l'information mis dans ce dernier. En somme ce changement est partiellement maîtrisable.

Le réseau effectue aussi un déplacement des fréquences d'une plage à l'autre. Signalons que cette opération mathématique est assez complexe à effectuer dans le domaine temporel. Néanmoins, les performances du réseau semblent assez intéressantes. Ce genre d'opération n'est pas réalisable avec les filtres classiques.

Pour conclure l'analyse du réseau sur les signaux périodiques, disons que le réseau de neurones est en mesure d'effectuer le filtrage d'un signal dynamique ayant des paramètres constants dans le temps (signal sinusoïdal) même si celui-ci est bruité avec des rapports signaux/bruit très élevés (-10 dB), ce qui est très intéressant. Ce filtrage devient plus compliqué dès que la fréquence augmente; le réseau pourrait confondre les hautes fréquences et le bruit. Cependant, l'augmentation de la fréquence d'échantillonage et du nombre de noeuds à l'entrée du réseau pourrait résoudre ce problème.

L'étude expérimentale du filtrage des réseaux de neurones à propagation-avant, fonctionnant sous l'algorithme de la rétropropagation rapide, a montré que le réseau de neurones est effectivement capable de supprimer une partie du bruit dans un signal. Cependant, la présence du bruit dans les fichiers d'apprentissage a des répercussions sur la convergence de l'algorithme, et ce, d'autant plus lorsque les données d'apprentissage sont complexes, malgré la variation d'un certain nombre de paramètres comme par exemple le nombre de

noeuds dans la couche d'entrée. Une des méthodes utilisées dans le cadre de ce travail consiste essentiellement à intervenir au niveau des données de l'apprentissage. Cependant, cette méthode s'est avérée inefficace en présence de bruit, surtout quand la durée d'apprentissage est limitée (quelques heures seulement).

L'insuffisance première du réseau se situe au niveau de la discrimination entre un signal haute fréquence et un bruit: les deux signaux sont analysés identiquement. La seule différence entre les deux signaux est l'amplitude, qui se veut constante pour un signal périodique et variable (aléatoire) pour le bruit. Cette difficulté peut être surmontée par l'augmentation de la fréquence d'échantillonnage. Cependant, l'augmentation de ce paramètre a pour conséquence la croissance du nombre de noeuds dans la couche d'entrée, ce qui entraîne l'augmentation des calculs et en l'occurrence le temps d'apprentissage. Rappelons que l'un des objectifs visés depuis le début, est de disposer d'une architecture des plus compactes et ainsi d'un temps d'apprentissage minime (implantation en temps réel). Rappelons aussi que la base de données de parole utilisée (BDSON3) est échantillonnée à 16 kHz.

Les résultats obtenus montrent que le fait de réaliser l'apprentissage sur des données bruitées (10, 6 , 3 , 0 et -10 dB) génère une difficulté de convergence (augmentation de la durée d'apprentissage). Ceci entraîne la réduction des capacités de filtrage du réseau. C'est pour cela qu'il semble bien filtrer le signal bruité en basses fréquences (2000 Hz) et coupe le signal des hautes fréquences. Sur ce point, les performances du modèle de Chin'ichi Tamura et Alex Waibel sont comparables. Cependant, les conditions expérimentales ne sont pas identiques, les données sont différentes (parole japonaise — abstraction faite sur le fait qu'il s'agit toujours de la parole d'un point de vue spectral et, elle est produite par le même type d'instrument :appareil phonatoire), la durée de l'apprentissage n'est pas comparable et finalement l'architecture est différente (4 couches au lieu de 3).

CHAPITRE 4

CONCLUSION ET RECOMMANDATIONS

4.1 Conclusion

L'analyse effectuée dans le cadre de ce travail a nettement démontré la capacité d'un réseau de neurones à propagation-avant à mémoriser un signal périodique, à le regénérer avec toute l'information requise (amplitude, fréquence et phase), de façon presque indéfinie, et enfin à généraliser au niveau de l'amplitude et des fréquences. Quelques unes de ces capacités n'avaient jamais été explicitement démontrées. Nous avons également souligné l'importance des données d'apprentissage pour le réseau de neurones — le réseau apprend les relations intrinsèques des données à l'apprentissage.

Il est évident que le traitement du réseau de neurones est statique. Cependant la stratégie utilisée pour l'apprentissage demeure importante. Il y a deux techniques possibles:

— 1° une approche en série, qui consiste à faire l'apprentissage sur des vecteurs de données contigües.

— 2° une approche aléatoire, qui consiste à faire l'apprentissage sur des vecteurs de données séléctionnés aléatoirement.

Nous commencerons l'analyse selon la deuxième méthode, dite aléatoire. Le fait de choisir aléatoirement un vecteur de données peut accèntuer la discontinuité des vecteurs de données à la sortie et par conséquent générer une déformation dans le signal (périodique) à la sortie du réseau. Le fait de choisir les vecteurs de façon aléatoire peut privilégier des vecteurs de données sur d'autres, ce qui modifiera le signal regénéré. Cependant, s'il s'agit d'un patron statique, le fait de choisir les vecteurs de données de façon aléatoire rendra

l'apprentissage du réseau plus robuste. L'abscence de relation horizontale dans les patrons statiques en est la cause principale. Cependant, cette relation horizontale est prédominante dans un patron périodique. Un patron périodique a besoin de l'information qui l'a précédé pour définir son état actuel dans le réseau de neurones. Par contre, un patron statique n'en a nullement besoin. C'est d'ailleurs là que se révèle l'importance de la récurrence dans un réseau de neurones qui traite un signal dynamique. Chaque information est caractérisée par plusieurs états.

En résumé, l'approche aléatoire, n'est pas aisée et convient au traitement de données statiques.

L'approche séquentielle consiste à choisir un vecteur de données de façon contigüe. Le fait de les choisir séquentiellement aide le réseau à mémoriser l'information plus rapidement et à établir un lien solide entre les vecteurs. C'est ce dernier qui réduit l'écart que l'on remarque à la fin de vecteurs de données (discontinuité). Cela a pour conséquence d'aider le gradient de l'erreur à atteindre le minimum. La durée de l'apprentissage est fonction de la complexité des données à apprendre. Nous reconnaissons que chaque vecteur de données est présenté indépendamment du précédent, excepté que le traitement se base sur l'information présentée auparavant. Cette idée est plus simple à implanter lorsque les données d'apprentissage sont simples (sinusoïdes de fréquence f et d'amplitude A).

La méthode choisie dans ce travail se base sur l'approche série. Le réseau peut interpoler et extrapoler, ce qui facilite sa généralisation. Cependant, l'interpolation est faible quand les niveaux d'amplitude et de fréquence sont trop bas.

Ces qualités nouvellement observées sur une structure à propagation-avant pourraient éventuellement être utilisées dans la conception de filtres, tels qu'un passe-bas, un passe-haut, un passe-bande et un coupe-bande.

La comparaison de notre travail à celui de Ryotaro Kamimura montre que notre modèle a effectivement appris à mémoriser et à regénérer un signal périodique sans aucun problème. De plus, le réseau est en mesure de reproduire des signaux qu'il n'a jamais vus auparavant,

mais qui sont les résultats d'opérations mathématiques (addition et multiplication de deux signaux) sur les signaux appris. Cette tâche semble difficilement réalisable avec le modèle de Ryotaro Kamimura [41], car ce modèle est testé uniquement sur les signaux appris lors de l'apprentissage (présence de dégradation). Ce travail lève le voile sur les capacités d'un réseau de neurones à propagation-avant au niveau du traitement d'un signal périodique — signal dynamique ayant des propriétés variables dans le temps. Le noeud du problème réside dans la technique de réalisation de l'apprentissage.

Nous avons déjà discuté de la caractérisation d'un patron quelconque dans un réseau de neurones et nous pensons que le réseau récurrent caractérise l'information avec plusieurs états, de façon équivalente à un biais flottant (variable), c'est-à-dire qui change même après l'apprentissage. Cela n'est pas le cas pour un réseau à propagation avant, qui possède une contribution du seuil toujours stable et fixe. Nous avons voulu étudier la contribution du seuil après l'apprentissage; nous n'avons néanmoins pas pu établir un lien intéressant avec le signal appris. Peut être n'avons nous pas fait l'effort nécessaire.

Concernant la convergence de l'algorithme utilisé par Ryataro Kamimura, les performances sont comparables. Nous aimerions préciser que c'est la complexité des données qui régit ce paramètre. Dans notre travail, les données d'apprentissage utilisées sont principalement des fonctions sinusoïdales pures, qui se manipulent bien et se prêtent bien à ce genre de travail.

Les performances des filtres basés sur les réseaux de neurones sont intéressantes. Leur méthode consiste à apprendre l'information intrinsèque des données (signaux purs: amplitude et fréquence) et à généraliser à partir de ceux-ci. L'opération est assez simple, au sens où nous n'avons pas besoin de faire une programmation de l'algorithme. Nous concentrons notre effort plutôt sur les données d'apprentissage. Rappelons que notre tâche dans ce travail consistait à explorer les réseaux de neurones pour effectuer du filtrage. La comparaison de ce modèle de filtre (passe-bas) à des filtres passe-bas classiques est intéressante. Le nombre d'opérations mathématiques (multiplication et addition) est

nettement supérieur avec l'approche neurale qu'avec l'approche classique. La comparaison est de l'ordre de: $[n_h.(n_{input}+1)+n_{output}.(n_h+1)]$ multiplications, et $[n_h.(n_{input})+n_{output}.(n_h)]$ additions, pour le filtre basé sur l'approche neurale, où n_h est le nombre de noeuds dans la couche cachée, n_{input} est le nombre de noeuds dans la couche d'entrée et n_{ouput} le nombre de noeuds dans la couche de sortie. Par contre, pour le filtre classique passe-bas (FIR) d'ordre n, nous avons n multiplications et (n+1) additions pour un échantillon. Cela nous montre clairement que le filtre basé sur l'approche classique (passe-bas) est nettement supérieur à celui basé sur l'approche neurale du filtre en terme de vitesse de traitement. Cependant, ce fait nous était connu dès le départ.

Par contre, du point de vue de l'adaptation, les filtres basés sur l'approche neurale sont facilement adaptables pour filtrer dans une plage spécifique. Un filtre passe-bas peut devenir un passe-haut ou un passe-bande ou même un coupe-bande, il suffit de faire un apprentissage adéquat (matrice de poids). Par contre, pour un filtre classique, cela est complètement impossible. Il faut absolument modifier la programmation de l'algorithme.

L'expérimentation nous a montré et prouvé que nous pouvons concevoir un filtre capable de réaliser des déplacements partiels de fréquences — performances ordinaires mais possibles. C'est une opération mathématique très complexe. Il est important de rappeler que le traitement se fait dans le domaine temporel et non dans le domaine spectral.

Les filtres basés sur les réseaux de neurones posent un problème, notamment celui des fréquences de coupures. Nous avons constaté que les filtres basés sur les réseaux de neurones dépassent légèrement la fréquence imposée. Cela est principalement causé par un phénomène similaire au repliement qui est créé par le réseau de neurones —tendance à généraliser.

Maintenant, si nous comparons les performances de notre modèle à celles du modèle proposé par Chin'ichi Tamura et Alex Waibel, nous dirons que les résultats obtenus sont comparables. Les performances du modèle de Chin'ichi Tamura sont presques identiques. Soulignons cependant que les conditions expérimentales, les données (parole japonaise),

l'architecture du réseau, l'algorithme d'apprentissage et le temps d'apprentissage sont différents. Cependant, les performances de notre modèle du point de vue de l'architecture, temps d'apprentissage, et réponse de filtre sont nettement supérieures à celui-ci (un rapport de 3 pour 1).

En résumé, le modèle proposé dans ce travail est un réseau de neurones capable de réaliser plusieurs filtres avec des performances intéressantes tels qu'un filtre passe-bas, un filtre passe-haut, un filtre passe-bande et un filtre coupe-bande (filtre à rejéction), facile à adapter aux besoins de l'usager. Finalement, tout apprentissage qui est fait sur des données de parole bruitée (gaussien) revient à concevoir un filtre passe-bas.

4.2 Recommandations

— éviter de faire l'apprentissage sur des signaux bruités, surtout si le nombre de noeuds dans la couche d'entrée est inférieur à 19 noeuds;
— augmenter la fréquence d'échantillonnage;
— augmenter le nombre de noeuds dans la couche d'entrée;
— introduire un mélange de type de fonction de transfert dans un réseau de neurones;
— utiliser un pré-traitement afin de mieux discriminer le bruit du signal haute fréquence.
— concevoir un réseau de neurones hybride pour le filtrage d'un signal dynamique.

BIBLIOGRAPHIE

[1] *NeuralWorks professional II*, 1988.

[2] F.R Hampel (1968). "Contribution to the Theory of Robust Estimation". Ph.D Dissertation, University of California, Berkeley, Departement of Statistics.

[3] M.J. Anderson. "Distributed memory". *Physics*, 1969.

[4] J.S Baras and A. La Vigna. "Convergence of Kohonen's Learning Vector Quantization". *IJCNN International Joint Conference On Neural Networks, New York: IEEE Press, III:17–20*, 1990.

[5] E.B. Baum. " On the Capabilities of Multilayer Perceptrons". *J. Complexity, Vol 4, P: 193–215*, 1988.

[6] Eric Baum and David Haussler. "What Size Net Gives Valid Generalisation". *Neural Computing 1, P: 151–160*, 1989.

[7] Mark Beale and Howard Demuth. "Preparing controllers for Nonlinear Systems". *AI Expert, P: 42–46*, July 1992.

[8] Y. Bengio and Bengio S. "Learning a synaptic learning rule". Technical report, (# 751), Computer Science Departement, Université de Montreal.

[9] Yoshua Bengio. *"Artificial Neural Network and their Application to sequence recognition "*. PhD thesis, departement of computer science, Mc Gill University, Montreal, 1991.

[10] Chris Bishop. "Improving The Generalisation Properties Of Radial Basis Function Neural Networks". *Neural Computing 3, P:579–588*, 1991.

[11] J. Blum. "Multidimentional Stochastic Approximation Procedure". *Annals of Mathematical Statistics, 25 , 737–744*, 1954.

[12] René Boite. *"Traitement de la parole"*. Presses Polytechniques Romandes, 1987.

[13] Blanchet P. Bottou L., Fogelman Soulie F. and Lienard J.S 1990. "Speaker independent isolated digit recognition: multilayer perceptrons vs dynamic time warping". *Neural Networks 3, P: 453–465*, 1990.

[14] H. Bourlard and C.J. Wellekens. "Speech Dynamics and Recurrent Neural Networks". *ICASSP 89, P: 33– 36*, 1989.

[15] Pierre Cardaliaguet. "Approximation of a Function and it Derivate with a Neural Network". *Neural Network, VOL. 5, N0 2, P: 207–217*, 1992.

[16] Maureen Caudill. "The View from Now". *AI Expert, P: 24–31*, June 1992.

[17] Maureen Caudill. "Evolutionary Neural Network". *AI Expert, P: 28–33*, March 1991.

[18] Maureen Caudill. "Driving Slo". *AI Expert, P: 26–31*, September 1991.

[19] Khalid Choukri. "Speech Recognition using Connectionnist Approches". *Neural Computing, P: 262–269*, 1991.

[20] Kurt Hormink Chung-Ming Kuan and Halbert White. "Some Convergence Results For Learning in Recurrents Neural Networks". *International Joint Conference on Neural Networks*, 1989.

[21] Ted Crooks. "Care and Feeding of Neural Networks". *AI Expert, P: 36–41*, July 1992.

[22] Hinton D.E. Rumelhart and R.J. Williams. "Learning Internal Representations by Error Propagation". *D.E. Rumelhart, McClelland and The PDP Research Group, Parallel Distributed Processing: Explorations in The Microstructures of Cognition, Vol. 1, Cambridge: MIT Press, pp. 318–362*, 1986.

[23] Dowla F. U E. M. Johanson and Goofman D.M. "Back-propagation learning for multi-layer feed-forward neural networks using the conjugate gradient method". Technical report, (UCRL-JC-104850) Preprint, Lawrence Livermore National laboratory, 1990.

[24] Eric R. Buhrke et Josef L. Lociero. "Speech Recognition With Neural Networks And Networks Fusion". Technical report, Illinois Institute of Technoly, Departement of Electrical and Computer Engineering, Chicago, Illinois 60616 USA, 1991.

[25] John Gambale and Dave Holden. "Neural Works Professinal II/ Plus". *AI Expert, P: 51–65*, July 1991.

[26] L. Goldstein. "Mean Square Optimality in continuous Time Robbins-Monro Procedure". Technical report, DRB-306, University of Southern California, Departement of Mathematics, 1987.

[27] F.R Hampel. "Robust Estimation". *A condensed Partial Survey, "Zeitschrift für Wahrschrscheinlichkeitstheorie und Verwandete, 27, 87–104*, 1973.

[28] Jean-Paul Haton and Michel-Yves Bernard. *"La parole et son traitement automatique"*. Calliope, 1989.

[29] Simon Haykin. *"Adaptive Filter Theory"*. Prentice-Hall , England Cliffs, 1986.

[30] H.Boulard and C. T. Wellekens. "Speech Dynamic and recurent Neural Network". *ICASSP*, 1989.

[31] Robert Hetch-Nielsen. *"Neuro Computing "*. Addition-Wesley Publishing Company, 1990.

[32] David Hillman. "Knowledge Systems Based on Cascading Neural Nets". *AI Expert, P: 46–53*, December 1991.

[33] S.C Huang and Y.F Huang. "Bounds on Numbers of Hidden Neurons in Multilayer Perceptrons". *IEEE , Neural Networks, vol 2. P: 47–55*, 1991.

[34] P. Huber. *"Robust Statistics"*. New York : John Wiley, 1981.

[35] R. A. Jacobs. "Increased rates of convergence throught learning Rate adapation". *Neural Networks 1, P: 295–307*, 1988.

[36] J.Elman. "Finding Structure in Time". Technical report, 8801, University of California, San Diego, Center for Research in Language", 1988.

[37] Michael I. Jordan. "Indeterminate motor skill learning problems". Technical report, Department of Brain and Cognitive Science, Massachusetts Institute of Technology, University of California, San Diego, USA, 1989.

[38] Michael I. Jordan. "Seriel order: A parallel distribued processing approch". Technical report, Department of Brain and Cognitive Science, Massachusetts Institute of Technology, University of California, San Diego, USA, May, 1986.

[39] Michel I. Jordan. " Statistical Theory can provide valuable insight in the advantage and desadvantage of neural network learning procedures". Technical report, Department of Brain and Cognitive Science, Massachusetts Institute of Technology, University of California, San Diego, USA, 1986.

[40] Ryotaro Kamimura. "Application of Temporal supervised Learning Algorithm to generation of naturel language". *IJCNN '90, P: I-201 – I–207*, 1990.

[41] Ryotaro Kamimura. "Recognition And Restoration of Periodic Pattrons with Recurent Neural Net". *IJCNN 90, P:689–698*, 1990.

[42] Fumio Kanaya and Shigeki Miyake. "Bayes Statistical Behavior And Valid Generalization of Pattern Calssifying Neural Networks". *IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 2, No. 4, P: 471–475*, 1991.

[43] Coleman Kevin G. and Susan Watenpool. "Neural Networks in Knowledge Acquisition". *AI Expert, P: 36–39*, January 1992.

[44] Jessica Keyes. "Getting Caught in a Neural Network". *AI Expert, P: 44–50*, July 1991.

[45] Setrag Khoshafian. "Modeling Object-Oriented Databases". *AI Expert, P: 26–33*, October 1991.

[46] Chung-Ming Kuan and Halbert White. "Recursive M-Estimation, Nonlinear Regression and Neural Network Learning With Dependant Observation". Technical report, University Of California, San Diego, USA, June 1990.

[47] Chung-Ming Kuan and Halbert White. "Predicting Appliance Ownership Using Logit, Neural Network and Regression Tree Models". Technical report, University Of Illinois, Urbana-Champaign, October 1989.

[48] H.J Kushner and D.S Clark. *"Stochastic Approximation Methods for Constrained and Unconstrained Systems"*. New York: Springer Verlag, 1978.

[49] Bottou L. *"Une approche théorique de l"apprentissage connexioniste; application à la reconnaissance de la parole"*. PhD thesis, Université de Paris XI, 1991.

[50] CH. Jutten et J. Caelen L. Nguyen Thi. "Rehaussement de la parole par un modèle de séparation de sources". *13 ème Colloque GRETSI, JUAN-LES-PINS*, 16–20 Septembre 1991.

[51] CH. Jutten et J. Caelen L. Nguyen Thi. "Séparation aveugle de parole er de bruit dans un mélange convolutif". *13 ème Colloque GRETSI, JUAN-LES-PINS*, 16–20 Septembre 1991.

[52] J. Caelen et CH. Jutten L. Nguyen Thi. "Rehaussement de la parole par séparation de sources additives ". *13 ème Colloque GRETSI, JUAN-LES-PINS*, 16–20 Septembre 1991.

[53] Chafic Mokbel Laurent Barbier and Gérard Cholet. "Trainable Noise Substraction Filters For Speech Enhancement In the Car". *SIGNAL PROCESSING V: Theories and Applications, L.Torres, E.Masgrau and M.A. Lagunas (eds), Elsevier Science Publishers B.V.,P: 1111–1115*, 1990.

[54] Jeannette Lawrence. "Data Preparation for a Neural Network". *AI Expert, P: 34–41*, November 1991.

[55] K. Jous Lee. "Constructive Approximation For Neural Network By Sigmoïd Fonction ". *Proceeding of the IEEE*, october 1990.

[56] L. Ljung. *"System Identification: Theory fot the User"*. Englwood Cliffs, N.J : Prentice Hall, 1987.

[57] L. Ljung and T. Soderstrom. *"Theory and Practice of Recursive Identification"*. Cambridge, Mass.: MIT Press, 1983.

[58] Donald B. Malkoff. "A Neural Network for Real Time Signal Processing". *Neural Computing, P :248–255*, 1991.

[59] Mario Marchand. "Les réseaux de neurones". *Interface*, Nov.-Dec. 1990.

[60] McClelland. *"Parallel Distributed Processing"*. University of California, 1988.

[61] S.Manetti M.Forti and M.Marini. "Neural Network For Adaptive FIR Filtering". *ELECTRONICS LETTERS, VOL. 26, No 14, P: 1018–1019*, April 1990.

[62] J. Mivellan. "Error Functions to Improve Noise Resistance and Generalization in Backpropagation Networks". *IJCNN-90–WASH D.C,International Joint Conference on Neural Networks. Hillsdale, N;J : Lawrence Erlbaum Assoc., I: 557–560*, 1990.

[63] Chafic Mokbel and Gerard Chollet. "Word Recognition in The Car: Speech Enhancement / Spectrale Transformations". *ICASSP 91,P: 925–932*, 1991.

[64] R.J.T Morris and W.S Wong. "A New Approach to Choosing Initial Points In Local Search". *Information Procesing Letters, 30, 67–72*, 1989.

[65] Patrick Naim and Eric David. *"Réseaux de neurones"*. EYROLLE, 1989.

[66] N.J. Nilsson. *"Learning Machines"*. New York: McGraw-Hill, 1965.

[67] Murray P. E. Gill and right M.H. *"Practical Optimization"*. Academic Press, 1981.

[68] A. H.Waibel P. Haffner and K. Shikano. "Fast Back-propagation learning nethods for large phonemic neural networks". *Proceeding of Eurospeech*, September 1989.

[69] Francesco Palmieri and Samir A. Shah. "Fast Training of Multilayer perceptron using multilinear parametrisation". *IJCNN 90, P:696–699*, 1990.

[70] Barak A. Pearlmutter. "Learning State Trajectories In recurrent Neural Networks". *Neural Computing 1, P: 263–269*, 1989.

[71] Fernado J. Pineda. "Generalisation of back propagation to recurent Neural Networks". *PHYSICAL REVIEW LETTERS, VOL. 59, No 19, P: 2229–2232*, November 1987.

[72] D. A POMERLEAU. "An Autonomous Land Vehicle in a Neural Network". *Advances in Neural Information Processing System, San Mateo, C.A, Morgan Kaufman Publ. , P: 305–313*, 1989.

[73] P. Révész. "How to apply The Method of Stochastic Approximation to the Non Parametric Estimation of a Regression Function". *Math. Operationsforsh. Statis., Ser., Statistics, 8, 119–226*, 1977.

[74] H. Robbibs and S. Monro. "A Stochastic Approximation Method". *Annals of Mathematical Statistics, 22, 400–407*, 1951.

[75] Le Cun Y. S. Becker. "Improving the convergence of back-propagation learning with second-order methods". *In Touretzky, Hinton and Sejnowski eds., Proc. of the 1988 Connectionist Summer School, P: 29–37, San Mateo. Morgan Kaufman*, 1989.

[76] P. Saratchandran. "Dynamic Programming Approach To Optimal Weight selection In Multilayer Neural Networks". *IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 2, No. 4, P: 465–467*, July 1991.

[77] Michael A. sartori and Panos J. Antsaklis. "A Simple Method to Derive Bounds On the Size To Train Multilayer Neural Networks". *IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 2, No 4, P: 467–471*, July 1991.

[78] Robert S. Scalero. "A Fast Training Algorithm For Neural Networks". *IJCCN 90, P: I-715– I-718*, 1990.

[79] Tom J. Schwartz. "A Neural Chpis Survey". *AI Expert, P: 34–39*, December 1990.

[80] Eric J. Smythe. "Temporal Representations In A Connectionnist Speech System". *Neural Computing, P:241–247*, 1989.

[81] Chin'ichi Tamura and Alex Waibel. "NOISE REDUCTION USING CONNECTIONIST MODELS". *ICASSP 88, P: 553–556*, 1988.

[82] Samad Tariq. "Back-Propagation is significantly faster if the expected value of the source unit is used for update". In *international Neural Network Society Conference*

Abstracts, 1988.

[83] Samad Tariq. "Back-Propagation Extensions". Technical report, Honeywell SSDC, 1000 Boone Ave. N., Golden Valley, NM 55427, 1989.

[84] James Ting-Holo. "Backestimation for training Multilayer Peceptron". *ICASSP 91, P: 1065 – 1069, Toronto — Canada*, 1991.

[85] Don Tveter. "Getting a Fast Break with Backprop". *AI Expert, P: 36–43*, July 1991.

[86] Paul. J. Werbos. "Backpropagation througth Time : what it does and how it do ". *Proceeding of the IEEE*, October, 1990.

[87] Halbert White. "Statistical Theory can Provide Insight Into The Adavantage of Neural Network Learning Procedures". *AI EXPERT*, 1989.

[88] Halbert White. "Supervised Learning As Stochastic Approximation". *International Joint Conference on Neural Networks*, 1990.

[89] Halbert White. *"An overview of representation and convergence results for multilayer feedforward networks"*. ed. D.S Toureszky Morgan Kauffman Publ., 1991.

[90] Halbert White. "Some Asymptotic Results for Learning in Single Hidden-Layer Feedforward Network Models". *Journal of the American Statistical Association, P: 1003–1013*, December, 1989.

[91] Halbert White. "Connectionist Nonparametric Regression: Multilayer FeedForward Networks Can Learn Arbitrary Mappings". Technical report, University Of California, San Diego, USA, November 1989.

[92] Halbert White. "Learning In Artificial Neural Networks : A Statistical Perspective". Technical report, University Of California, San Diego, USA, October, 1989.

[93] Halbert White and Jeffrey M. Woodridge. "Some Results on Sieve Estimation With Dependent Observations". Technical report, University Of California, San Diego, USA, May 1989.

[94] Bernard Widrow. *"Adaptive Signal Processing"*. Prentice-Hall Inc., 1985.

[95] Bernard Widrow and Michael A. Lehr. "30 years of Adaptive Neural Network: Perceptron, Madaline and BackPropagation". *Proceeding of the IEEE*, September, 1990.

[96] Ronald J. Williams and P. Zipser. "A Learning Algorithm For Continually Running Fully Recurent Neural Networks". *Neural Computing 1, P: 270–280*, 1989.

[97] Ronlad J. Williams and David Zipser. "Experimental Analysis of the Real time Recurrent Learning Algorithm". *Connection Science, VOl. 1, No 1, P: 87–111*, 1989.

[98] Le Cun Y. "Generalization and network design strategies". Technical report, (CRG-TR-89–4) Departement of Computer Science, University of Toronto, 1989.

[99] Le Cun Y. "Generalization and network design strategies. Connectionism in Perspective". *(Pfeifer, Schreter, Fogelman, Steels eds), North Holland, P: 143–155*, 1989.

[100] P. Gallinari Y. Bennani, N.Chaouar and A.Mellouk. "Comparaison of Neural Network Models on Speech Recognition tasks". Technical report, University of Paris Sud, LRI, 1990.

APPENDICE A

FORMULATION MATHÉMATIQUE: ALGORITHME D'APPRENTISSAGE

Cette section porte sur le développement de l'algorithme d'apprentissage utilisé — rétropropagation rapide [82, 83]. Nous allons débuter notre analyse par la discusion des éléments à seuil unitaire et sortie linéaire; la règle de correction de l'erreur n'utilisant pas de gradient; la règle linéaire: α—LMS ou Widrow-Hoff ou règle delta; la règle non linéaire: perceptron; la règle de correction de l'erreur utilisant la méthode du gradient; la règle linéaire utilisant le gradient; la règle non linéaire du gradient; la règle du "steepest descent": "multi-éléments"- Rétropropagation; la rétropropagation récurrente; la rétropropagation cumulée; et finalement la rétropropagation rapide; Nous allons aussi effectuer un rappel de l'algorithme "TSLA".

Règles de correction de l'erreur n'utilisant pas la méthode du gradient

Nous débutons l'analyse des algorithmes d'apprentissage par la description des méthodes et techniques qui sont à la base de la rétropropagation rapide [82, 83]. Nous présenterons les règles de correction de l'erreur qui n'utilisent pas la méthode du gradient et par la suite, celles qui recourrent à ce dernier. Les principales fonctions de ces algorithmes seront illustrées par des équations mathématiques.

Les différents algorithmes sont présentés sur des architectures multi-couches (feed-forward) où seulement le nombre et le type de noeuds dans les couches d'entrée, cachée et de sortie peuvent varier, exception faite de l'adaline et du perceptron qui sont composés d'une structure mono-couche.

L'équation qui régit la mise à jour des poids lors de l'apprentissage est exprimée par l'équation (A-6)

$$W_{k+1} = W_k + \Delta W_k \qquad \text{A-6}$$

où

W_{k+1} représente la valeur du poids à l'itération k+1;

W_k représente la valeur du poids à l'itération k (itération précédente);

et

ΔW_k représente la valeur de la variation qui s'impose.

Régle linéaire: α LMS ou Widrow Hoff delta

Cet algorithme est régi par l'équation (A–7), qui permet la mise à jour des poids du réseau de neurones (connexions synaptiques), pour une structure mono-couche : adaline (adaptive linear). La sortie de l'adaline peut être linéaire ou binaire.

$$W_{k+1} = W_k + \alpha . e_k . \frac{X_k}{||X_k||^2}; \qquad \text{A- 7}$$

où l'erreur e_k est exprimée par l'équation (A–8):

$$e_k = d_k - W_k^T . X_k; \qquad \text{A- 8}$$

$$0.0 < \alpha < 2.0 \qquad \text{A- 9}$$

où

— α représente un facteur de relaxation permettant la stabilisation de la convergence;

— X_k représente le vecteur de données à l'entrée du réseau de neurones, à l'itération k;

— $s_k = W_k.X_k$ représente la sortie d'un noeud (unité) de la couche cachée (ou la couche d'entrée);

— y_k représente le vecteur de sortie obtenu, où $y_k = f(s_k)$; $f(s_k) = +/- 1$ dans le cas binaire ou $f(s_k) = s_k$ dans le cas linéaire.

Remarque: On notera que cet algorithme ne fait intervenir aucune dérivée ni gradient. Le traitement est purement linéaire.

Règle non linéaire: perceptron

Cet algorithme est gouverné par l'équation (A–10). Il ne permet d'avoir que des sorties binaires, soit {–1,1} ou {0 ,1} — le cas précédent permet d'avoir des sorties linéaires et discrète. Il est à noter que les données ne sont pas normalisées

$$W_{k+1} = W_k + \alpha.\frac{e_k}{2}.X_k; \qquad \text{A- 10}$$

où

e_k est totalement différent du précédent cas (équation A-8); l'erreur e_k est exprimée par: $e_k = y_k - d_k$ où $y_k = (1,-1)$ et $d_k = (1,-1)$. La mise à jour des poids se fait avec la règle du perceptron de Rosenblatt.

$\alpha = 1$, contrairement au cas précédent , ce paramètre n'affecte nullement la convergence.

Cet algorithme permet d'avoir deux valeurs binaires à la sortie de l'unité de décision : {-1,1} ou {0,1}, situées par rapport à un seuil; d'où la non-linéarité.

De plus, cet algorithme a permis à David Rumelhart (1957) , pour la première fois de réaliser la reconnaissance d'objets (image).

Règles de correction de l'erreur utilisant la méthode du gradient

Plus haut, nous avons décrit la règle de correction n'utilisant pas le gradient, règle destinée à réduire l'erreur lors de la présentation des données de l'apprentissage. Souvent,

l'objectif d'adaptation est de réduire l'erreur globale dans l'apprentissage. La fonction d'erreur la plus populaire est l'erreur quadratique moyenne "MSE". Cette méthode utilise le gradient pour la minimisation de l'erreur et est appelée la méthode de la "steepest descent". D'autres approches sophistiquées (quasi-Newton, le gradient conjugué etc...), basées sur l' approche du gradient donnent de meilleurs résultats en ce qui concerne la convergence. Cependant, la complexité croît avec les performances. Dans certains cas, l'amélioration d'un paramètre (la convergence) peut engendrer la détérioration d'un autre (la généralisation).

Règle linéaire utilisant le gradient

Cet algorithme est gouverné par l'équation (A–18). Nous nous poserons d'abord, la question suivante: d'où vient la linéarité ? En guise de réponse, nous démontrerons que le changement de poids est proportionnel à l'erreur (linéarité).

L'erreur quadratique peut être exprimée par l'équation (A–11):

$$e_k^2 = \left(d_k - X_k^T.W_k\right)^2 \qquad \text{A- 11}$$

où

— d_k représente la sortie désirée pour une unité à l'itération k ;

— X_k représente le vecteur d'entrée pour une unité à l'itération k ;

— W_k représente le vecteur des poids à l'itération k.

Chaque élément va produire à l'itération k une erreur dont l'espérance mathématique est exprimée par l'équation (A–12):

$$E(e_k^2) = E(d_k^2) - 2.E\left(d_k.X_k^T\right).W_k + W_k^T.E\left(X_k.X_k^T\right).W_k; \qquad \text{A- 12}$$

Pour alléger la formulation de l'écriture, on posera:

$$P^T = E\left(d_k.X_k^T\right) \qquad \text{A- 13}$$

où P^T représente la corrélation entre la sortie (d_k) et l'entrée (X^T_k).

et

$$R = E\left(X_k.X_k^T\right); \qquad \text{A- 14}$$

R représente l'auto-corrélation.

L'équation précédente (A–12) devient (A-15):

$$\zeta_k = E\left(d_k^2\right) - 2.P^T.W_k + W_k^T.R.W_k; \qquad \text{A- 15}$$

On notera ζ_k l'espérance mathématique de l' erreur quadratique. Il s'agit d'une équation quadratique par rapport à la variable W_k, de la forme $a.w^2 + b.w + c$. C'est une surface convexe (surface ayant un seul minimum global). Alors, si on calcule le gradient de ζ_k (équation A–15), avec $W_k = W$, on obtient l'équation (A–16):

$$\nabla_k = -2.P + 2.R.W; \qquad \text{A- 16}$$

où ∇_k représente le gradient de l'espérance mathématique de l'erreur à l'itération k.

Il s'agit alors d'une fonction linéaire des poids. Le poids optimal sera donné par l'équation (A–17):

$$W^* = R^{-1}.P; \qquad \text{A- 17}$$

Dans ce qui a précédé, nous avons effectué un développement théorique permettant de calculer l'espérance mathématique de l'erreur. Cependant, qu'est ce qui se passe dans le cas où nous ne pouvons pas calculer cette espérance mathématique ? En guise de réponse, nous devons l'estimer.

Soit ∇'_k le gradient instantané estimé, basé sur la valeur instantanée de l'erreur quadratique e_k^2.

Si nous reportons les changements dans l'équation d'apprentissage (A–6), nous obtenons l'équation (A-18: A-18d)

$$
\begin{aligned}
W_{k+1} &= W_k + \mu(-\nabla_k); \\
-\nabla'_k &= -\frac{\partial e_k^2}{\partial W_k}; \\
W_{k+1} &= W_k - 2.\mu.e_k.\frac{\partial(d_k - W_k^T.X_k)}{\partial W_k}; \\
W_{k+1} &= W_k + 2.\mu.e_k.X_k; \\
0.0 < \mu &< \frac{1}{trace(R)};
\end{aligned}
\qquad \text{A- 18}
$$

Cet algorithme porte le nom de : μ—LMS.

Il est à noter que la sortie de chaque unité est linéaire (il n'y pas de sigmoïde).

Règle non linéaire du gradient

Dans le paragraphe précédent, la sortie des neurones demeure une combinaison linéaire des entrées. Nous décrivons maintenant un autre type de sortie utilisant une fonction non linéaire (la sigmoïde).

Soit l'équation permettant la modification des poids dans un réseau (A–19):

$$
\begin{aligned}
W_{k+1} &= W_k + \mu.\left(-\nabla_k^{'}\right);\\
e_k &= d_k - y_k = d_k - sgm(s_k);\\
\nabla_k^{'} &= \frac{\partial(e_k)^2}{\partial W_k} = 2.(e_k).\frac{\partial(e_k)}{\partial W_k};\\
\frac{\partial(e_k)}{\partial W_k} &= -sgm'(s_k).\frac{\partial(s_k)}{\partial W_k};\\
s_k &= X_k^T.W_k;\\
\nabla_k^{'} &= -2.e_k.sgm'(s_k).X_k;\\
W_{k+1} &= W_k + 2.\mu.e_k.sgm'(s_k).X_k;
\end{aligned}
\qquad \text{A- 19}
$$

Si nous prenons comme approximation de la fonction non linéaire la sigmoïde, l'équation A-19g devient (A–20c):

$$
\begin{aligned}
y_k &= tanh(s_k) = \frac{\left(1 - e^{-2.s_k}\right)}{\left(1 + e^{-2.s_k}\right)};\\
sgm^{'}(s_k) &= \frac{\partial tanh}{\partial s_k} = 1 - \left(tanh(s_k)^2\right) = 1 - y_k^2;\\
&alors\\
W_{k+1} &= W_k + 2.\mu.e_k.\left(1 - y_k^2\right).X_k;
\end{aligned}
\qquad \text{A- 20}
$$

L'équation (A–20) représente la mise à jour des poids par la méthode non linéaire du gradient pour un réseau multi-couche. La différence entre l'équation (A–18) et l'équation (A–20) réside dans l'ajout d'un facteur qui tient compte de la dérivée de la fonction de sortie ($sgm'(s_k)$); facteur atténuateur qui tient compte de la sortie dans la mise à jour des connexions synaptiques (poids).

La règle de " la steepest descent" —multi-éléments: algorithme de la rétro-propagation

Maintenant que nous avons exposé la base de la théorie qui a été à l'origine du développement de l'algorithme de la rétropropagation, nous étudierons de manière détaillée cet algorithme, afin de pouvoir comprendre et résumer ce qui suivra.

Soit W une matrice de m éléments, où m représente toutes les différentes connexions synaptiques du réseau.

L'erreur quadratique moyenne instantanée (globale) est donnée par l'expression (A–21):

$$e_k^2 = \sum_{i=1}^{Ny} e_{ik}^2; \qquad \text{A- 21}$$

où

$$e_{ik} = (d_{ik} - y_{ik}) \qquad \text{A- 22}$$

avec

e_{ik} représentant l'erreur de l'unité i à l'itération k;

d_{ik} représentant la sortie désirée de l'unité i à l'itération k;

y_{ik} représentant la sortie désirée de l'unité i à l'itération k;

e_k représentant l'erreur globale de la couche de sortie à l'itération k;

i le rang de l'unité de traitement;

N_y nombre d'unités de sortie.

Dans sa forme la plus simple, l'algorithme de la rétropropagation fonctionne comme suit: si un vecteur de données est présenté à l'entrée X, le réseau de neurones formel génère une sortie Y. L'algorithme d'apprentissage calcule l'erreur entre la sortie désirée D et la sortie obtenue Y. La prochaine étape consiste à tenir compte de l'effet de l'erreur globale sur les connexions synaptiques précédentes, pour associer une contribution à la

dérivée δ à chaque connexion. Le gradient est calculé à partir de chaque δ et une mise à jour des différentes connexions synaptiques (poids) est effectuée et ce, de façon inverse à la propagation de l'information (données). Et ainsi de suite, ce calcul se perpétue jusqu'à la précision désirée. De plus, le processus se répète pour tous les autres patrons. Il est à noter que les poids initiaux sont toujours pris de façon aléatoire [95].

Une fois que l'erreur de la sortie finale est déterminée, l'algorithme calcule la dérivée instantanée de cette dernière (carré de l'erreur) qui est associée à chaque noeud dans le réseau. La dérivée du carré de l'erreur associée avec le $j^{ème}$ noeud dans la couche l est définie par l'équation (A–23):

$$\delta_j^l = -\frac{1}{2}.\frac{\partial e_k^2}{\partial s_j^l} \qquad \text{A- 23}$$

Chacune de ces dérivées nous renseigne sur la façon dont la mise à jour des poids devrait être faite pour améliorer la tendance de l'erreur globale.

La dérivée instantanée de l'erreur quadratique est précisément calculée pour chaque noeud dans la **couche de sortie**, comme le montre l'équation (A–24):

Afin d'alléger la démonstration, on supprimera l'indice k;

$$\delta_j^{(l)} = -\frac{1}{2}.\frac{\partial e_k^2}{\partial s_j^{(l)}} = -\frac{1}{2}.\frac{\partial\left(\sum_{j=1}^{N}(d_j - y_j)^2\right)}{\partial s_j^{(l)}}; \qquad \text{A- 24}$$

Si nous prenons comme exemple d'application un réseau ayant une couche cachée de 3 noeuds, une couche d'entrée de 3 noeuds et une couche de sortie de 2 noeuds, alors:

$$\delta_1^{(2)} = -\frac{1}{2}.\frac{\partial\left((d_1 - y_1)^2 + (d_2 - y_2)^2\right)}{\partial s_1^{(2)}};$$

$$\delta_1^{(2)} = -\frac{1}{2}.\frac{\partial\left(\left(d_1 - sgm\left(s_1^{(2)}\right)\right)^2\right)}{\partial s_1^{(2)}} - \frac{1}{2}.\frac{\partial\left(\left(d_2 - sgm\left(s_2^{(2)}\right)\right)^2\right)}{\partial s_1^{(2)}}; \qquad \text{A- 25}$$

où 1 = 2 caractérisant la couche de sortie.

Or, le deuxième terme est nul, ce qui donne:

$$\delta_1^{(2)} = -\frac{1}{2}.\frac{\partial\left((d_1 - y_1)^2\right)}{\partial s_1^{(2)}};$$

$$y_1 = sgm\left(s_1^{(2)}\right); \qquad \text{A- 26}$$

$$\delta_1^{(2)} = \left(d_1 - sgm\left(s_1^{(2)}\right)\right).sgm'\left(s_1^{(2)}\right);$$

d'où

$$\delta_1^{(2)} = e_1^2.sgm'\left(s_1^{(2)}\right) \qquad \text{A- 27}$$

Or, on a calculé précédemment que:

$$sgm'\left(s_1^{(2)}\right) = 1 - y_1^2 \qquad \text{A- 28}$$

D'où

$$\delta_1^{(2)} = e_1^2.(1 - y_1^2) \qquad \text{A- 29}$$

Si nous développons maintenant l'expression de la dérivée de l'erreur au carré, mais par rapport à la **couche cachée**, nous trouvons (gardant toujours le même exemple) l'équation

(A–30):

$$\delta_j^{(l)} = -\frac{1}{2}.\frac{\partial e^2}{\partial s_j^{(l)}}; => \delta_1^{(1)} = -\frac{1}{2}.\frac{\partial e^2}{\partial s_1^{(1)}}; \qquad \text{A- 30}$$

Développons cela par la règle de dérivée (fonction de fonction) (A–31):

$$\delta_1^{(1)} = -\frac{1}{2}.\frac{\partial e^2}{\partial s_1^{(2)}}.\frac{\partial s_1^{(2)}}{\partial s_1^{(1)}} - \frac{1}{2}.\frac{\partial e^2}{\partial s_2^{(2)}}.\frac{\partial s_2^{(2)}}{\partial s_1^{(1)}};$$

$$\delta_1^{(1)} = \left(\delta_1^{(2)}.\frac{\partial s_1^{(2)}}{\partial s_1^{(1)}} + \delta_2^{(2)}.\frac{\partial s_2^{(2)}}{\partial s_1^{(1)}}\right); \qquad \text{A- 31}$$

$$\delta_1^{(1)} = \delta_1^{(2)}.\left(\omega_{10}^{(2)} + \sum_{i=1}^{3} w_{1i}^{(2)}.sgm\left(s_i^{(1)}\right)\right) + \delta_2^{(2)}.\left(\omega_{20}^{(2)} + \sum_{i=1}^{3} w_{2i}^{(2)}.sgm\left(s_i^{(1)}\right)\right);$$

or

$$\frac{\partial sgm\left(s_i^{(l)}\right)}{\partial s_j^{(l)}} = 0, i \mp j; \qquad \text{A- 32}$$

ce qui laisse A–33

$$\delta_1^{(1)} = \delta_1^{(2)}.w_{11}^{(2)}.sgm'\left(s_1^{(1)}\right) + \delta_2^{(2)}.w_{21}^{(2)}.sgm'\left(s_1^{(1)}\right);$$

$$\delta_1^{(1)} = \left(\delta_1^{(2)}.w_{11}^{(2)} + \delta_2^{(2)}.w_{21}^{(2)}\right).sgm'\left(s_1^{(1)}\right) \qquad \text{A- 33}$$

Par conséquent, nous pouvons donner la définition suivante:

$$e_1^{(1)} = \delta_1^{(2)}.w_{11}^{(2)} + \delta_2^{(2)}.w_{21}^{(2)}; \qquad \text{A- 34}$$

sachant que $\delta_1^1 = e_1^1.sgm'(s_1^1)$.

Nous avons maintenant établi une formule générale afin de trouver la dérivée δ pour chaque noeud dans le réseau i.e par rapport à la couche de sortie (δ_1^2) et la couche cachée (δ_1^1). Nous allons utiliser cette formule pour obtenir le gradient correspondant.

Soit un vecteur de données quelconque, qui lors de la $k^{ème}$ présentation est noté: X_k; le produit de ce dernier avec le vecteur des poids W_k, donnera la sortie $s_k = W_k . X_k$;

Le gradient instantané de l'erreur quadratique est (A–35):

$$\nabla'_k = \frac{\partial e_k^2}{\partial W_k} \qquad \text{A- 35}$$

cela peut être écrit comme (A–36):

$$\nabla'_k = \frac{\partial e_k^2}{\partial W_k} = \frac{\partial e_k^2}{\partial s_k} . \frac{\partial s_k}{\partial W_k}; \qquad \text{A- 36}$$

On note que W_k et X_k sont indépendants (A–37):

$$\frac{\partial s_k}{\partial W_k} = \frac{\partial W_k^T . X_k}{\partial W_k^T} = X_k; \qquad \text{A- 37}$$

alors

$$\nabla'_k = \frac{\partial e_k^2}{\partial s_k} . X_k; \qquad \text{A- 38}$$

d'où

$$\nabla'_k = -2.\delta_k . X_k \qquad \text{A- 39}$$

Alors, la mise à jour se fait de la manière suivante (A–40):

$$W_{k+1} = W_k + \mu . \left(-\nabla'_k\right) = W_k + 2.\mu.\delta_k . X_k \qquad \text{A- 40}$$

N.b : δ_k prend différentes formes en fonction de son emplacement par rapport à la couche d'entrée et la couche cachée.

En résumé, la mise à jour des poids se fait par l'équation A–40.

Brève illustration de la rétropropagation et de l'information temporelle

Après avoir décrit le formalisme de la méthode de la rétropropagation, nous analyserons sa version temporelle ou récurrente.

La version temporelle de cette méthode est largement utilisée pour la reconnaissance des patrons variant dans le temps. Cet algorithme peut être appliqué à la reconnaissance de la parole, la détection sous-marine, etc.... La classification dans le temps serait plus précise au temps t si on connaissait ce qui s'est passé avant . Ceci nécessite la mémorisation des états précédents aux temps (t-1), (t-2) ..... et (t-n) et surtout les relations liant ces états i.e la matrice W_{t-1}, etc.... Ce qui implique une mémoire dont la taille est proportionnelle à l'étendue de l'information. Cependant, la durée de l'apprentissage est courte, en comparaison à la rétropropagation standard qui n'a pas besoin de mémoire mais plutôt de temps pour converger à l'état optimal.

Une fois introduit l'algorithme d'apprentissage des réseaux récurrents, nous allons brièvement illustrer le formalisme mathématique qui régit cette méthode (A–41):

$$y_i = \sum_{j=1}^{N} W_{ij} * x_j(t) + \sum_{j=1}^{n} W'_{ij} * x_j(t-1) + \sum_{j=1}^{n} W''_{ij} * x_j(t-2);$$

ou

$$W_{ij} : \textit{matrice des poids au temps } t;$$

$$W'_{ij} : \textit{matrice des poids au temps } (t-1);$$

$$W''_{ij} : \textit{matrice des poids au temps } (t-2);$$

$$N : \textit{nombre d'unités d'entrée};$$

$$n : \textit{nombred'unités de sortie}$$

A- 41

N.B : si nous annulons W'_{ij} et W''_{ij} , alors nous retrouvons la rétropropagation standard.

Les travaux de Paul J. Werbos [86] ont montré que la rétropropagation récurrente a l'avantage d'être relativement rapide et exacte. Cependant ses performances en présence du bruit sont faibles.

Analyse et discusion des différents travaux effectués sur l'algorithme de la rétropropagation

Maintenant que nous avons présenté le principe de l'algorithme, nous pourrons faire une synthèse de quelques travaux ayant trait à cet algorithme. Nous commencons par les travaux d'Halbert White.

Le but de l'apprentissage est d'arriver à une matrice optimale $\hat{W}$, qui peut nous permettre d'approximer la sortie du réseau, pour fournir la relation qui nous intéresse.

La méthode de la rétropropagation semble bien faire cette tâche. Si ∇' " nabla" dénote le gradient de l'erreur globale estimée par rapport à W, alors une des versions de cet algorithme se présente comme suit :

$$W_{k+1} = W_k + \mu.\left(-\nabla'_k\right) = W_k + 2.\mu.\delta_k.X_k \qquad \text{A- 42}$$

Cette équation correspond bien à celle de Rumelhart, Hinton et Williams (1986) [22]. Dans cette version de la rétropropagation, il n'y avait pas de regroupement de données pouvant mettre cet idéal d'apprentissage en ligne ou en temps réel.

Dans cette équation, nous constatons que les nouveaux poids sont égaux aux anciens, abstraction faite du facteur d'ajustement qui résulte de l'information fournie par le plus récent apprentissage. L'ajustement est proportionnel à l'erreur du réseau qui est propagée en sens inverse du système. Le gradient résout en quelque sorte le problème du "credit assignment " jusqu'à un facteur de proportionnalité, qui est donné par μ ("rate learning" ou régime d'apprentissage).

Les travaux d'Halbert White (lien parallèle) s'alignent sur ceux de Robin et Monro (1951), dont il se sert pour expliquer le fonctionnement de l' algorithme de la rétropropagation

En résumé, les travaux d'Halbert White [20, 46, 47, 87, 88, 90, 92, 93] suggèrent que tout ce qui est vrai en général pour les approximations stochastiques "Stochastic Approximation " l'est en particulier pour la rétropropagation.

Inconvénients de la rétropropagation

Nous désirons faire une bonne approximation de la sortie du réseau avec une convergence rapide. Les travaux de H. White [46, 88] ont démontré que la convergence peut être bonne si μ (dans ce cas-ci μ est variable, on peut l'écrire comme $\mu(t)$) tend vers 0 plus lentement que $n^{-1/2}$. D'autre part, L. Goldstrein (1987)[26] a démontré qu'une convergence rapide de la matrice des poids à un état optimum arrive quand $\mu(t)$ est proportionnel à n^{-1} où n est le nombre d'itérations. H. White a suggéré une approche pour améliorer la convergence, afin d'atteindre un minimum global. L'approche dite à " recommencement multiple" initialise l'apprentissage de la rétropropagation avec un grand nombre de valeurs de poids différentes pour débuter. De meilleurs résultats peuvent découler d'un partage de l'espace de poids en régions qui ne se recoupent pas ("non-overlapping regions"), et alors initialiser la rétropropagation à partir d'un point de recommencement aléatoire pour chaque région.

Une méthode standard pour obtenir une meilleure convergence est la méthode de regroupement. Au lieu de mettre à jour le réseau avec chaque exemple d'apprentissage, la correction est moyennée sur un certain nombre d'exemples (souvent sur toutes les données d'apprentissage). Il en résulte une règle d'apprentissage exprimée par l'équation (A–43):

$$W_{k+1} = W_k + 2.n^{-1}.\sum_{j=0}^{n} \mu^j.\delta_k^j.X_k \qquad \text{A- 43}$$

où n représente le nombre d'itérations cumulées — entre chaque nouvelle W_k—.

N.b : cette équation (A–43) diffère de la précédente (A–42) par le deuxième terme, car l'ajustement des poids comprend la moyenne cumulée de tout ce qui a précédé, au lieu d'une seule correction.

L'avantage de cette méthode est qu'elle élimine le bruit, alors que l'ajustement des poids évalue mieux la fonction du gradient à minimiser. L'approximation est d'autant meilleure que n est assez grand.

Halbert White [46] a établi un lien (parallèle) avec celui de "approximation stochastic". D'ailleurs, il se base sur ce dernier pour expliquer cette théorie.

Mais il est à noter que le fameux facteur μ^j (μ(t)) a une interprétation et une fonction qui diffère du régime d'apprentissage μ dans la rétropropagation non regroupée (partie précédente). Dans l'algorithme précédent, μ constitue en quelque sorte un facteur d'oubli " forgetting facteur", vu qu'il est requis pour la convergence (n —> 0). Il accomplit une sorte de lien de parenté avec celui de la rétropropagation regroupée.

Cependant, en rétropropagation regroupée, le pas μ^j ne doit pas converger à zéro. Il doit être gardé suffisamment petit pour maintenir l'erreur en deça d'un seuil.

En comparant les deux méthodes précédentes: rétropropagation regroupée et non-regroupée, nous pouvons dire que la première technique avec un contrôle adéquat de μ^j est préférable pour aboutir à un minimum local (optimum), car le gradient de l'erreur au carré est mieux estimé. Le bruit extérieur de la rétropropagation non regroupée, peut être mieux éliminé (rebondissement autour de la surface). Il est à noter également, qu'il y a une conséquence pour la rétropropagation regroupée: le problème de la mémoire pour le stockage de l'information.

N.b: Pour aboutir à un optimum global, il faut essayer à plusieurs reprises, en utilisant la méthode "multi-start" avec l'algorithme de la rétropropagation non regroupée. Le bruit généré peut aider à contrer le problème des minima locaux.

En effet, il est peut-être plus intéressant de combiner les procédures de la rétropropagation regroupée et la rétropropagation non regroupée: implanter la méthode de "

multi-start" en utilisant la rétropropagation sans regroupement d'abord, quand cela semble long pour faire un progrès appréciable (diminution de l'erreur quadratique) puis utiliser la rétropropagation regroupée. Prendre le meilleur résultat de cet exercice et le comparer avec celui de multi-start; sélectionner alors le meilleur de tous. Ceci bien entendu nécessite un long temps de calcul, et surtout beaucoup d'utilisation de temps CPU.

Récemment, Nyugen [95] a constaté qu' un choix judicieux de poids initiaux pour les valeurs des couches cachées peut réduire les problèmes de l'optimum local et de la croissance dramatique du temps de l'apprentissage.

Analyse de la rétropropagation

La rétropropagation est un algorithme assez simple à implanter. Il permet d'avoir une bonne performance sur une très grande variété de transformations. Cependant, il présente certaines contraintes comme suit:

— lenteur de la convergence;
— problème des minima locaux qui est caractéristique des méthodes du gradient;
— choix des paramètres: régime d'apprentissage, architecture du réseau, nombre de noeuds cachés, etc...

Techniques d'accélération

Dans ce paragraphe, nous allons effectuer une comparaison de 2 techniques basées sur la rétropropagation: la mise à jour stochastique et la mise à jour déterministe (voir ci-dessous). Ceci est fait dans le but d'améliorer et perfectionner la méthode de la descente du gradient: améliorer la convergence et contrer le problème des minima locaux:

1. avec une mise à jour déterministe c-à-d. regroupée;

la modification des poids se fait après le calcul du gradient de la fonction erreur sur tous les patrons d'apprentissage.

$$\Delta W_{ij} = -\mu . \sum \frac{\partial e_k}{\partial w_{ij}} \qquad \text{A- 44}$$

2. avec une mise à jour stochastique c-à-d. non regroupée;

la modification se fait après chaque présentation du patron, utilisant une estimation limitée au gradient basé sur l'erreur locale

$$\Delta W_{ij} = -\mu . \frac{\partial e_k}{\partial w_{ij}} \qquad \text{A- 45}$$

La dérivée de l'erreur pour le patron par rapport au paramètre W_{ij}, peut être considérée comme une estimation bruitée du gradient total. La méthode stochastique de la descente du gradient a été étudiée pour le traitement du signal adaptatif [57] et géneralement converge bien sous certaines conditions[49, 89]. Pour l'apprentissage des réseaux de neurones avec la rétropropagation, la mise à jour avec la méthode stochastique a été trouvée plus rapide que la déterministe, dans plusieurs cas et en particulier pour les problèmes de reconnaissance des patrons avec un grand nombre de données (parole) [13]. Les performances de la mise à jour stochastique, peuvent être expliquées ainsi:

Quand la taille des données est importante, celles-ci contiennent une information redondante. L'algorithme passe plus de temps à chercher une meilleure précision qu'à définir une nouvelle orientation des vecteurs — chaque vecteur caractérise une information dans le réseau. Les travaux de Yann Le Cun [98, 99] ont montré que la méthode stochastique est deux fois plus efficace (en terme de convergence) que la méthode dite déterministe.

Un autre avantage vient de l'aspect aléatoire introduit par la valeur bruitée du gradient. Ce bruit peut aider à sauter les minima locaux et amplifie la valeur du régime d'apprentissage. En débutant par un grand régime d'apprentissage et en le réduisant pro-

gressivement, nous atteignons presque (approcher) le minimum global de la fonction de l'erreur [13]..

D'autres méthodes sont suggérées pour accélérer la convergence:

— adaptation (Si μ variable) dans le régime d'apprentissage;

— utilisation de différents paramètres [35] (le momentum, etc...).

L'avantage de ces méthodes est souvent contre-balancé par les désavantages de la mise à jour des poids pour les problèmes de reconnaissance des patrons complexes (avec de grandes quantités de données).

Une autre alternative explorée par Bengio [8] consite à contrôler le régime d'apprentissage indirectement, en contrôlant la moyenne du changement de poids dans chaque couche du réseau. Ceci a été expérimenté sur de petits réseaux et a démontré une convergence rapide.

Méthodes basées sur des techniques d'optimisation

D.E. Rumelhart, Hinton and R.J. Williams [22] ont proposé l'utilisation du terme "momentum", ce qui veut dire l'ajout d'une portion de l'ancien changement du poids à l'actuel. Ceci tend à réduire l'effet de zigzag, souvent observé avec la descente du gradient, en stabilisant le changement dans la direction de la moyenne de la descente la plus forte. Cette accélération due au "momentum" semble effective avec la méthode regroupée (déterministe) au lieu de la stochastique. La méthode du moment est actuellement apparentée à celle du gradient conjugué [67, 23]; dans les deux cas la direction de la recherche est obtenue à travers une combinaison linéaire successive du gradient.

De l'autre côté, les méthodes directes Newton et quasi-Newton utilisent la dérivée seconde de la fonction d'erreur tenant compte des paramètres du réseau. La méthode de

Newton semble souvent difficile dans le cas d'un grand réseau, et elle requiert un grand calcul et une grande capacité de stockage de la matrice Hessienne.

La quasi-Newton [75] considère la diagonale de la matrice Hessienne seulement. Cela permet un calcul rapide et une inversion de la dérivée seconde. Cette méthode présente également l'avantage de pouvoir être utilisée avec la mise à jour stochastique.

Cependant, elle requiert une prudente attention concernant le passage du zéro et la seconde dérivée négative tel qu'illustré dans l'équation (A-46) qui suit:

$$\Delta W_{ij} = \frac{\frac{\partial e_k}{\partial W_{ij}}}{\left|\frac{\partial^2 e_k}{\partial W_{ij}^2}\right| + \mu} \qquad \text{A- 46}$$

où μ est un coefficient ajustable.

Relation étroite : accélération versus généralisation

Dans certains cas, les techniques employées pour l'accélération ont une conséquence néfaste sur la généralisation des résultats des réseaux [68]. Ceci est illustré par la rétropropagation non regroupée.

Mise à jour des poids par la méthode stochastique: résumé

La mise à jour des poids avec la méthode stochastique (non regroupée) peut aider à contrer le problème des minima locaux, car c'est le gradient bruité qui est utilisé au lieu de la vraie valeur du gradient pour chaque paramètre. Par conséquent, l'amplitude de ce bruit est modulée par le régime d'apprentissage. Ceci permet une convergence efficace si on débute avec un grand régime d'apprentissage qu'on réduit graduellement.

La rétropropagation rapide: un compromis entre une convergence rapide et une bonne généralisation

A partir de cette section, on remarquera un changement dans la notation, ceci est dû principalement à l'introduction du facteur temps.

Un algorithme qui semble donner de bonnes performances en ce qui a trait à la généralisation — données de l'apprentissage composées principalement de signaux purs — tout en ayant une convergence rapide est l'algorithme de la rétropropagation rapide développé par Tariq Samad [82, 83, 1]. On citera dans ce qui suit un résumé de la formulation mathématique de cet algoritme.

Soit $x_i(t)$ montrant l'activité de la $i^{ème}$ unité au temps t, et w_{ij} représentant la valeur de la connexion synaptique entre la $j^{ème}$ et la $i^{ème}$ unité. La valeur de l'activité de la $i^{ème}$ unité au temps t+1 est donnée par l'expression (A–47):

$$x_i(t+1) = f(s_i(t)); \qquad \text{A- 47}$$

où f est la fonction sigmoïde donnée par l'équation (A–48):

$$f(s_i(t)) = \frac{1}{1+e^{-s_i(t)}}; \qquad \text{A- 48}$$

où $s_i(t)$ est donnée par l'équation (A–49):

$$s_i(t) = \sum_j W_{ij}.x_j(t); \qquad \text{A- 49}$$

Le réseau de neurones est forcé d'apprendre les valeurs cibles en minimisant la différence entre la sortie désirée et la sortie obtenue. Soit e_i cette différence pour l'unité de sortie i.

$$e_i^{[k]}(t) = d_i(t) - x_i(t); \qquad \text{A- 50}$$

où $d_i(t)$ représente la sortie désirée et k l'état de la matrice de poids du réseau au temps t;

L'erreur globale au temps t est représentée par (A–51):

$$J(t) = \frac{1}{2}.\sum_i \left[e_i^{[k]}(t)\right]^2; \qquad \text{A- 51}$$

Le changement de n'importe quel poids, au temps t, est exprimé par l'équation (A–52):

$$\Delta W_{ij}^{[k]} = -\eta . \frac{\delta J(t)}{\delta W_{ij}}; \qquad \text{A- 52}$$

$$\Delta W_{ij}^{[k]} = -\eta . \frac{\delta J(t)}{\delta I_j} . \frac{\delta I_j}{\delta W_{ij}}; \qquad \text{A- 53}$$

où I_j est donnée par (A–54):

$$I_j = \sum_i W_{ij}^{[k]} . x_i^{[k-1]}(t); \qquad \text{A- 54}$$

La variation de poids donnée par [82] est:

$$\Delta W_{ij}^{[k]} = -\eta . e_j^{[k]} . \left(x_i^{[k-1]}(t) + e_i^{[k-1]}(t) \right); \qquad \text{A- 55}$$

Une forme génerale de l'équation précédente (A–55) est donnée par [83]:

$$\Delta W_{ij}^{s} = -\eta . e_j^{[k]} . \left(x_i^{[k-1]}(t) + K . e_i^{[k-1]}(t) \right); \qquad \text{A- 56}$$

L'équation (A–56) représente la rétropropagation rapide [82], [83] quand $k \mp 0$; sinon elle représente la rétropropagation standard [22] quand k = 0.

Un algorithme d'apprentissage pour un réseau récurrent [96]: TSLA

Notion de base de l'algorithme:

Soit un réseau de neurones ayant n unités, dont m sont des unités d'entrée (extérieures). Soit y(t) représentant (n-tuplets) les unités de sortie du réseau au temps t, et soit x(t) représentant les m-tuplets des unités d'entrées de signal au réseau dans le temps t. On fait une concaténation des y(t) et des x(t) pour former (n+m)-tuplets: z(t), avec U représentant le groupe des indices k tel que z_k est la sortie du réseau et I le groupe des indices k pour

lesquels z_k est l'entrée extérieure. Les indices sur y et x sont choisis pour correspondre au choix de z tel que l'indique l'équation (A–57):

$$Z_k(t) = \begin{cases} x_k(t) \text{ si } k \in I \\ y_k(t) \text{ si } k \in U \end{cases} \qquad \text{A- 57}$$

Soit W la matrice des poids pour le réseau de neurones. Chaque paire d'unités du réseau est reliée par une seule connexion synaptique. En adoptant la convention d'indexation décrite ci-dessus, nous pouvons incorporer tous les poids dans une seule matrice. De plus, pour permettre à chaque unité d'avoir une connexion avec le biais, nous avons inclu parmi m lignes d'entrées une entrée ayant une valeur fixe à 1.

Dans ce qui suit, nous utilisons une formulation discrète et nous assumons que le réseau est entièrement composé d'unités linéaires; nous avons prévu d'étendre cette approche au cas continu (dans le temps) et à d'autres formes de calculs d'unités ayant des fonctions différenciables.

On pose:

$$s_k(t) = \sum_{l \in U \cup I} w_{kl}.z_l(t); \qquad \text{A- 58}$$

Signalons que $z_l(t)$ peut être une entrée ou une sortie (récurrente) dependemment de son appartenance à l'ensemble U ou I et $s_k(t)$ représente l'entrée de la $k^{ème}$ unité, pour $k \in U$ (unités de sortie), avec sa sortie au temps t qui suit:

$$y_k(t+1) = f_k(s_k(t)); \qquad \text{A- 59}$$

où f_k est la fonction non linéaire utilisée dans le travail de Zipser et Williams pour les cellules de sortie seulement.

Le système d'équations (A–58) et (A–59) constitue le réseau dynamique en entier, où les valeurs des z_k sont définies par l'équation (A–57). On notera que les entrées extérieures au temps t n'influencent aucunement les sorties des unités. L'influence se fait sentir au temps t+1.

Maintenant, nous avons dérivé un algorithme pour entraîner un réseau de neurones appelé "temporal supervised learning", où certaines valeurs de sorties des unités sont bien spécifiées (ciblées) au moment opportun. Soit T(t) représentant le groupe d'indice $k \in U$ pour lequel il existe une valeur cible d_k telle que la $k^{ème}$ unité doit coincider au temps t. Alors nous définissons une variation temporelle n-tuplets par:

$$e_k(t) = \begin{cases} d_k(t) - y_k(t) & k \in T(t) \\ 0 & ailleurs \end{cases} \qquad \text{A- 60}$$

Nous noterons que cette formulation donne la possibilité de spécifier les différentes valeurs des unités de sortie ciblées à différents instants. Le groupe d'unités considérées visibles peut varier dans le temps.

Soit l'équation suivante (A–61) représentant l'erreur globale J(t) au temps t:

$$J(t) = \frac{1}{2} \cdot \sum_{k \in U} [e_k(t)]^2 \qquad \text{A- 61}$$

Pour le moment, nous assumons que le réseau s'éxécute du temps t_o à t_1. Nous prendrons comme objectif la minimisation de l'erreur totale (du temps t_0à t_1).

$$J_{total}(t_0, t_1) = \sum_{t=t_o+1}^{t_1} J(t) \qquad \text{A- 62}$$

Nous faisons cela avec une procédure de descente du gradient, en ajustant W le long des valeurs négatives $\nabla_W J_{total}(t_0, t1)$, de temps t_0 à t_1. L'erreur totale est la somme des erreurs individuelles à différentes étapes. Une façon de calculer ce gradient est d'accumuler les valeurs de$\nabla_W J(t)$ pour chaque instant. Le changement de poids pour n'importe quel poids w_{ij} dans le réseau peut être écrit comme suit:

$$\Delta W_{ij} = \sum_{t=t_0+1}^{t_1} \Delta W_{ij}(t) \qquad \text{A- 63}$$

avec

$$\Delta W_{ij}(t) = -\alpha.\frac{\partial J(t)}{\partial W_{ij}} \qquad \text{A- 64}$$

avec α un coefficient positif (régime d'apprentissage).

Maintenant:

$$-\frac{\partial J(t)}{\partial W_{ij}} = \sum_{k\in U} e_k(t).\frac{\partial y_k(t)}{\partial W_{ij}} \qquad \text{A- 65}$$

$\frac{\partial y_k(t)}{\partial \omega_{ij}}$ se calcule en dérivant les équations (A–58) et (A–59) régissant la dynamique du réseau,

$$\frac{\partial y_k(t+1)}{\partial W_{ij}} = f_k^{'}(s_k(t)).\left[\sum_{l\in k} W_{kl}.\frac{\partial y_l(t)}{\partial W_{ij}} + \delta_{ik}.z_j(t)\right] \qquad \text{A- 66}$$

où δ_{ik} représente le delta de Kronecker. Car, nous considérons que l'état initial du réseau n'a aucune dépendance fonctionnelle dans les poids et nous avons aussi

$$\frac{\partial y_k(t)}{\partial W_{ij}} = 0 \qquad (67)$$

. Ces équations sont valables pour :$k \in U, i \in U$ *et* $j \in U \cup I$.On peut alors créer un système dynamique d'équations avec des variables $\left\{p_{ij}^k\right\}$ pour tous les $k \in U, i \in U$ *et* $j \in U \cup I$ et la dynamique est donnée par :

$$p_{ij}^k(t+1) = f_k^{'}(s_k(t)).\left[\sum_{l\in k} W_{kl}.p_{ij}^k(t) + \delta_{ik}.z_j(t)\right]; \qquad \text{A- 68}$$

avec les conditions initiales suivantes:

$$p_{ij}^k(t_0) = 0 \qquad (69)$$

où

$$p_{ij}^k(t) = \frac{\partial y_k(t)}{\partial W_{ij} ij}; \qquad (70)$$

Pour tout temps t et pour toutes les propriétés de i, j, et k.

En résumé, l'algorithme consiste à calculer pour chaque temps t (de t_0 à t_1) , $\left\{p_{ij}^k\right\}$, en utilisant les équations (A–68) et (A–69), l'erreur $e_k(t)$ entre la sortie désirée et la sortie calculée.

Le changement de poids est représenté par l'équation suivante:

$$\Delta W_{ij}(t) = \alpha . \sum_{k \in U} e_k(t) . p_{ij}^k(t) \qquad \text{A- 71}$$

La correction globale qui peut être appliquée à chaque poids W_{ij} dans le réseau est la somme de ces valeurs $\Delta W_{ij}(t)$ pour chaque temps.

Dans le cas où chaque unité du réseau utilise la fonction sigmoïde, nous pouvons utiliser

$$f_k^{'}(s_k(t)) = y_k(t+1).[1 - y_k(t+1)] \qquad \text{A- 72}$$

dans l'équation A-68

Discussion

L'analyse de cet algorithme révèle une grande similarité avec la rétropropagation rapide dévelopée par Tariq Samad. Dans le cas de la rétropropagation rapide la mise à jour des poids se fait en tenant compte de l'erreur globale aux temps t, t-1 et de l'information (la donnée x) au temps t-1 comme nous l'avons montré dans l'équation A-56. Par contre, pour l'algorithme "TSLA", la mise à jour tient compte de l'erreur globale au temps t et de l'information (incluant la donnée d'entrée x et la sortie obtenue y) tel qu'illustré à l'équation A-68.

En récapitulant, l'algorithme "TSLA", où la mise à jour des poids est implicite, ressemble à celui de la rétropropagation rapide, où la mise à jour des poids est explicite; seulement la procédure de traitement diffère.

Brève analyse des modèles statistiques: "Backestimation for Training Multilayer Perceptrons" [84].

Les travaux de James Ting-Ho-Lo [84] proposent une nouvelle approche statistique, pour résoudre les problèmes souvent rencontrés avec les algorithmes existants dont la rétropropagation, à savoir:

— le choix des paramètres;
— le grand nombre d'itérations;
— leurs performances en terme d'erreur.

Les paramètres de l'algorithme proposé sont estimés avec une méthode statistique ou déterminés à partir des données. Le nombre d'itérations et de calculs semble être faible par rapport aux méthodes existantes. Cependant, le formalisme mathématique est un peu lourd, mais compréhensible. Si on se fie aux résultats obtenus, cet algorithme donne d'excellents résultats en terme de convergence. Néanmoins, cet algorithme est appliqué à l'apprentissage du perceptron et la fonction à apprendre est une fonction logique. Cela reste toutefois à vérifier.

Notre sujet traite d'un signal temporel dynamique, ce que le perceptron ne peut pas faire. Une étude approfondie permettrait de voir si ce dernier garde les mêmes performances pour le cas d'un signal temporel dynamique.

Pour vraiment comprendre les travaux de James Ting-Ho-Lo [84], il faut d'abord consulter les travaux d'Halbert White dans un premier temps, afin de comprendre par la suite le parallèle entre la rétropropagation et les travaux de Robbin et Monro (1951).

APPENDICE B

TABLEAUX ET GRAPHIQUES

Distribution des angles (vecteurs poids vs vecteur de données) dans un réseau de neurones

Données de l'expérience:

Nous avons pris un réseau de neurones multi-couches, avec 19 noeuds dans les couches d'entrée et de sortie. Le nombre de noeuds dans la couche cachée est variable (10, 20, 30 , 40 et 60). Nous avons effectué l'apprentissage sur des données simples: un signal sinusoïdal de fréquence 1200 Hz et d'amplitude 3000 (figures B.1 à B.3) et un signal sinusoïdal de fréquence variant de 50 Hz à 5050 Hz par intervale de 200 Hz ayant la même amplitude que le précédent (figures B.4 à B.8).

Une fois l'apprentissage terminé, nous avons effectué le rappel avec deux types de signaux: un signal sinusoïdal de fréquence 1200 Hz et une amplitude de 3000 et un signal de parole — mot français la bise —. Ensuite, nous avons calculé les angles que font les vecteurs de poids (connexions synaptiques) avec les vecteurs de données (pour les mêmes signaux de rappel: le signal sinusoïdal et le signal de parole. Le calcul de l'angle se fait entre les connexions se trouvant entre la couche cachée et la couche d'entrée et les données d'entrée d'une part; et celles (connexions) se trouvant entre la couche de sortie et la couche cachée d'autres part et les données de la sortie de la couche cachée. Et par la suite, nous avons reporté ces résultats sur un graphique.

En effet, nous avons constaté que chaque réseau de neurones possède une configuration spécifique et que chaque vecteur de données est contraint à respecter cette configuration — le vecteur de données est placé dans un espace de façon à former un angle spécifique

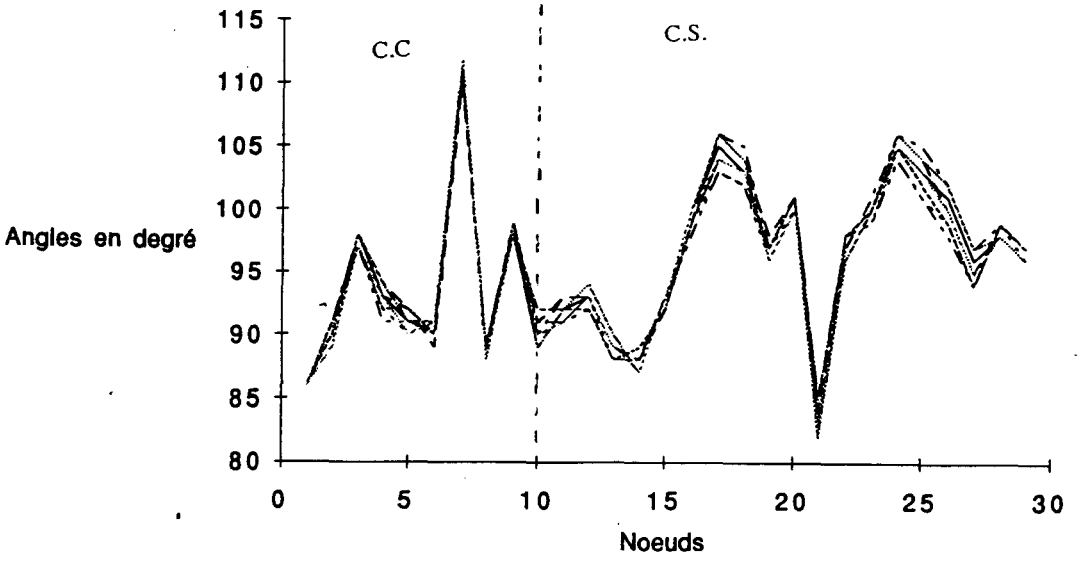

Figure B.1 Distribution des angles d'un réseau de neurones, composé de 10 noeuds dans la C.C et 19 noeuds dans les C.E et C.S. (signal pur de fréquence de 1200 Hz et d'une amplitude de 1000)

avec le vecteur des poids. Le résultat est illustré par les figures B.1 à B.8. Le seul paramètre qui pourrait changer est celui du nombre de noeuds dans la couche cachée du réseau. Les figures B.6 et B.8 illustrent une application sur un signal de la parole alors que l'apprentissage a été fait sur un signal sinusoïdal.

Malheureusement, l'étude de la distribution des vecteurs de poids vis-à-vis les vecteurs de données n'a pas été approfondie, car ce n'était pas l' objectif de notre projet. Cependant, cette alternative semble prometteuse et permet éventuellement d'ajuster les poids sans être obligé de faire un apprentissage. Même si elle présente une piste intéressante à explorer, une contrainte de temps nous en a empêché.

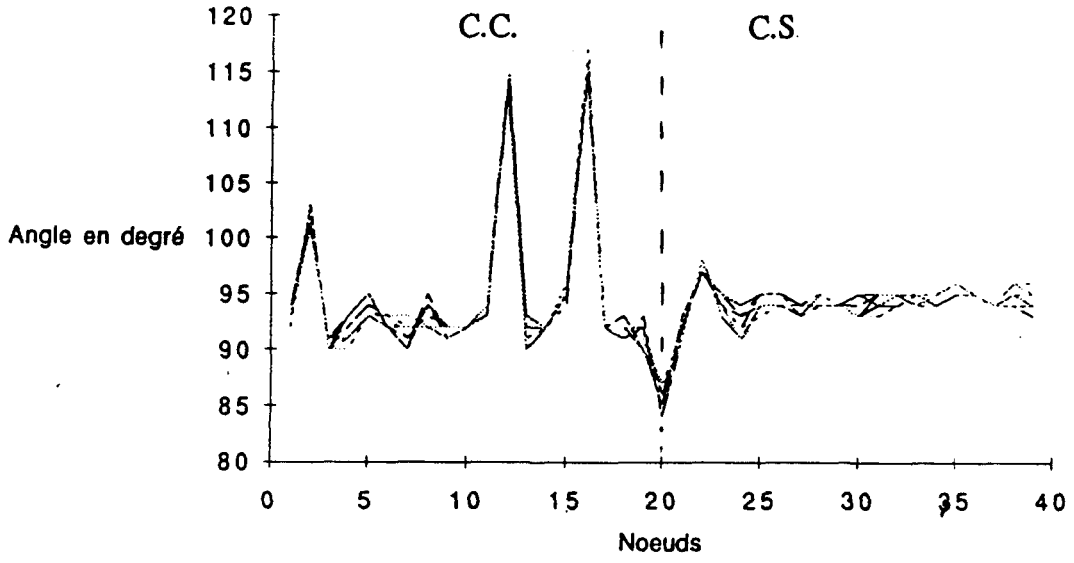

Figure B.2 Distribution des angles d'un réseau de neurones, composé de 20 noeuds dans la C.C et 19 noeuds dans les C.E et C.S. (signal pur de fréquence de 1200 Hz et d'une amplitude de 1000)

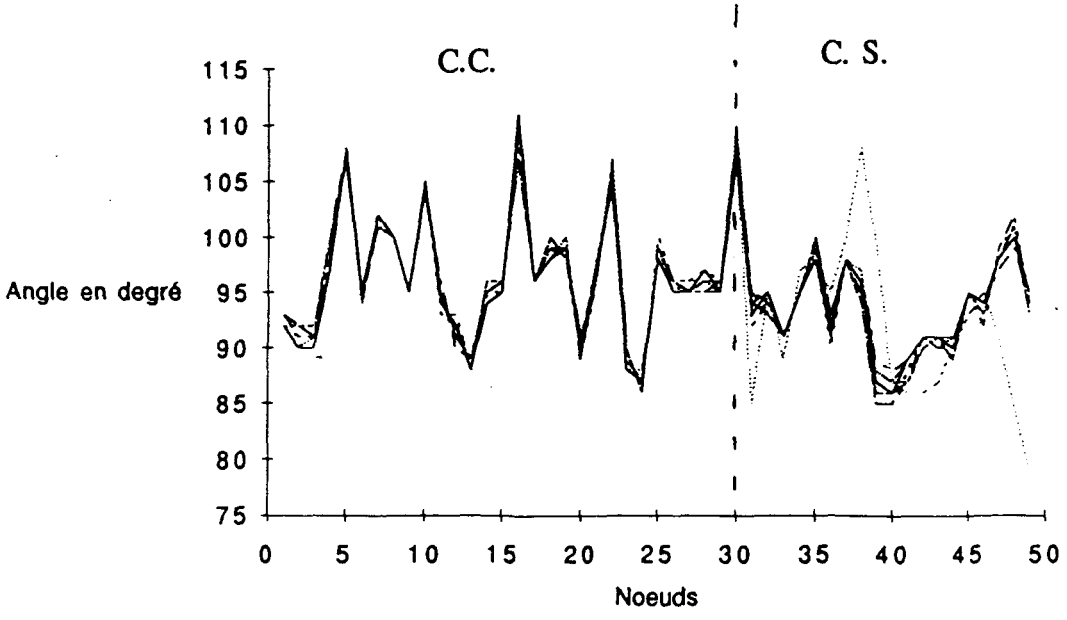

Figure B.3 Distribution des angles d'un réseau de neurones, composé de 30 noeuds dans la C.C et 19 noeuds dans les C.E et C.S. (signal pur de fréquence de 1200 Hz et d'une amplitude de 1000)

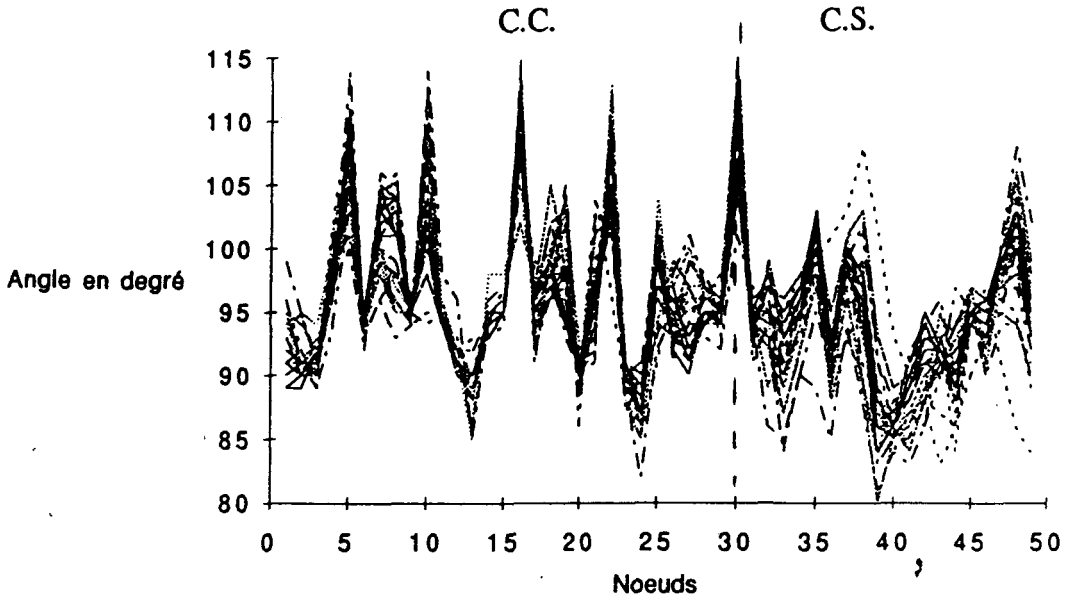

Figure B.4 Distribution des angles d'un réseau de neurones, composé de 30 noeuds dans la C.C et 19 noeuds dans les C.E et C.S. (signal de parole (50 Hz à 5 Khz))

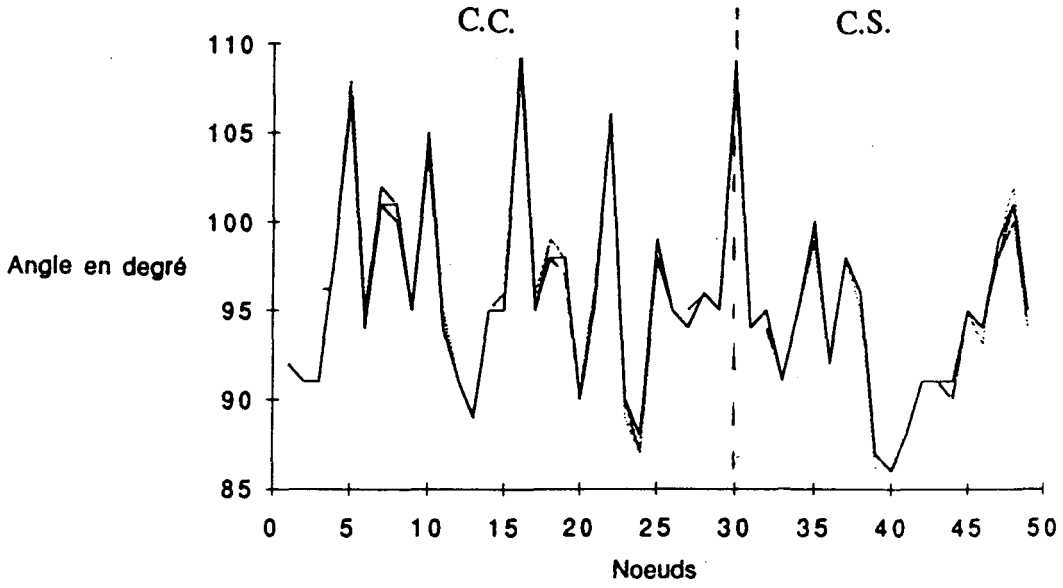

Figure B.5 Distribution des angles d'un réseau de neurones, composé de 30 noeuds dans la C.C et 19 noeuds dans les C.E et C.S. (signal de parole (50 Hz à 5 Khz) — (au début du signal)

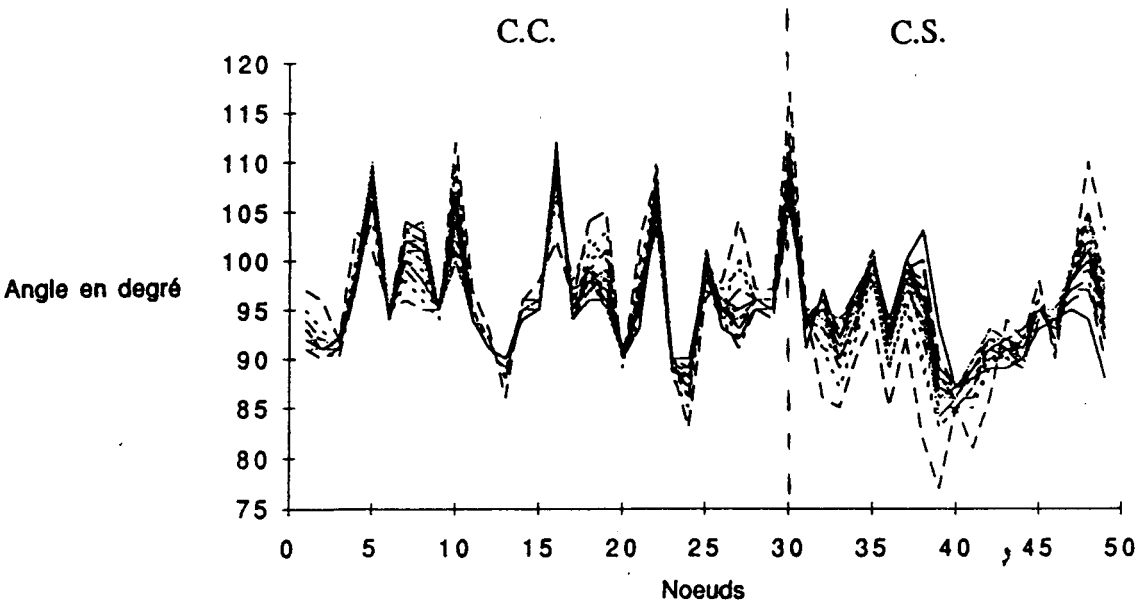

Figure B.6 Distribution des angles d'un réseau de neurones, composé de 30 noeuds dans la C.C et 19 noeuds dans les C.E et C.S. (signal de parole (50 Hz à 5 Khz) — (au début et au milieu du signal)

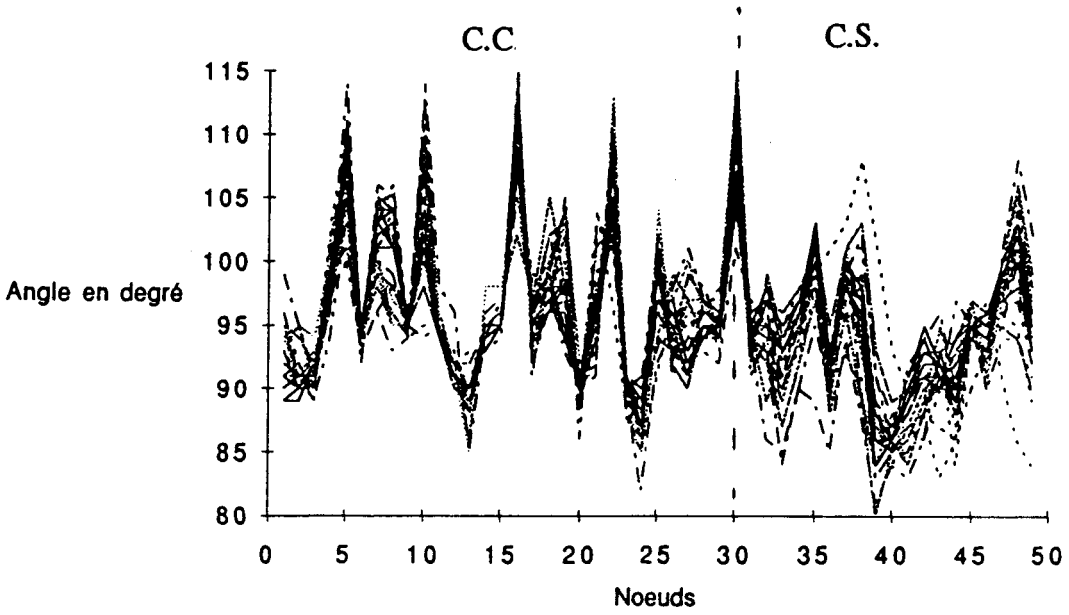

Figure B.7 Distribution des angles d'un réseau de neurones, composé de 30 noeuds dans la C.C et 19 noeuds dans les C.E et C.S. (signal de parole (50 Hz à 5 Khz) — (au milieu et à la fin du signal)

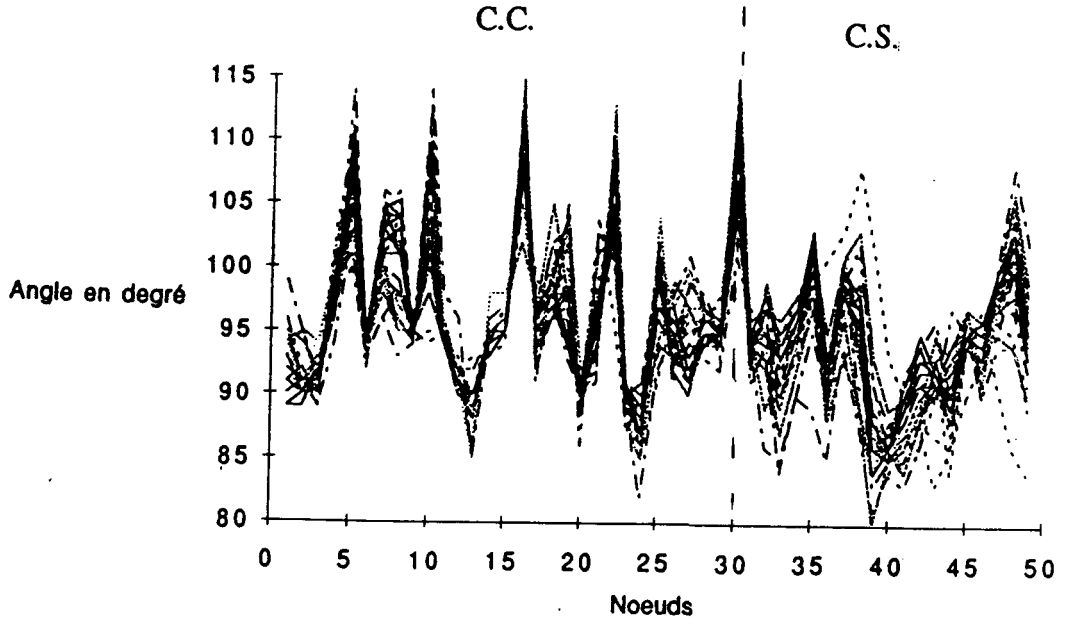

Figure B.8 Distribution des angles d'un réseau de neurones, composé de 30 noeuds dans la C.C et 19 noeuds dans les C.E et C.S. (signal de parole (50 Hz à 5 Khz) — (à la fin du signal)

APPENDICE C

ANALYSE DE LA FONCTION NON-LINÉAIRE: LA SIGMOÏDE

C.1 Choix du nombre de noeuds dans la couche de sortie

En principe, lorsqu'on souhaite traiter un signal (filtrage) par un réseau de neurones, on prend une dimension des vecteurs à l'entrée et à la sortie qui est la même. De plus, pour faciliter le traitement de ce signal, on doit le fragmenter (diviser) en plusieurs séquences (vecteurs de dimension n). Donc, après le passage de l'information dans le réseau, nous devrions retrouver la même information, sous la même forme et par conséquent sous le même vecteur. La dimension du vecteur à l'entrée du réseau est donc la même qu'à sa sortie.

C.2 Influence de la fonction non linéaire sur le réseau de neurones

La fonction non-linéaire est un paramètre très important pour un réseau de neurones. C'est elle qui attribue au réseau la capacité d'avoir des caractéristiques non linéaires. Elle peut être trigonométrique (sinus, cosinus) ou hyperbolique (tangente hyperbolique, sigmoïde, etc...). La sigmoïde simule le mieux la fonction échelon (perceptron) développée par Mc Cullogh en 1943.

Dans le cadre de notre travail, nous avons étudié particulièrement la sigmoïde dont l'équation se présente ainsi:

$$f(x) = \frac{1}{(1 + e^{-a.x})} \qquad \text{A-73}$$

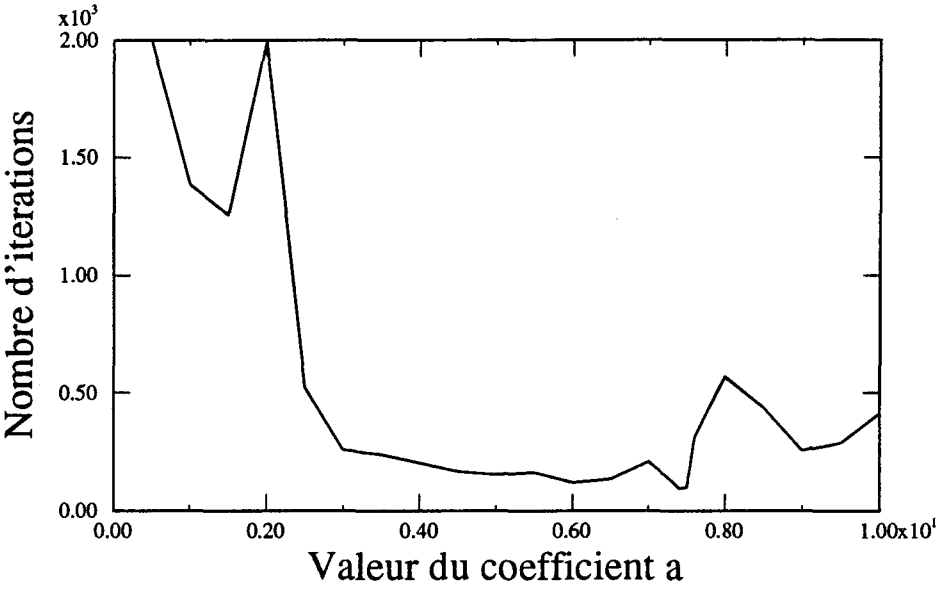

Figure C.1 Illustration de la convergence de l'algorithme de la rétropropagation standard en fonction de la valeur du coefficient " a " pour des patrons statiques.

où a est un coefficient réel positif.

Conditions expérimentales de cette étude:

Dans le cadre de notre travail, l'étude de la fonction non-linéaire (sigmoïde) est très importante. Nous avons choisi une architecture de type propagation-avant, avec 3 couches dont une cachée. La couche d'entrée contient trois noeuds linéaires. La couche cachée contient un noeud (couche non linéaire). La couche de sortie contient un seul noeud (non linéaire). Les données à l'apprentissage représentent la table de vérité d'une porte logique. L'algorithme d'apprentissage utilisé est celui de la rétropropagation standard, developpé par David Rumelhart en 1986 [22].

Alors pour étudier l'influence du coefficient "a", nous avons fixé le niveau d'erreur global à une valeur très petite ($\epsilon = 10^{-5}$) et le nombre d'itérations à un nombre maximal de 2000.

Nous avons constaté que pour des données statiques (tables de vérité de porte logique), une convergence de l'algorithme se situe pour une valeur du coefficient "a" comprise entre

2.5 et 7.5, tel que montré dans le tableau C.1 (figure C.1).

Les résultats de cette expérience ne peuvent pas être appliqués à notre modèle, car le logiciel utilisé pour le développement de réseaux de neurones ne permet pas de réaliser des changements, particulièrement en ce qui concerne la modification du coefficient "a", dont la valeur est toujours égale à 1. Cependant, les dits résultats peuvent nous éclairer sur le fonctionnement de ce dernier. La présente expérience est faite à titre informatif. Cependant, les résultats obtenus vont être utilisés dans nos futures expériences.

Nous constatons que le minimum d'itérations est atteint lorsque le coefficient "a" se trouve aux alentours de 7.4 (voir conditions de l'expérimentation).

Valeur du coefficient " a "	Nombre d'itérations atteint
0.025	2000
0.25	2000
0.5	2000
1.0	1384
1.5	1255
2.0	1999
2.5	523
3.0	257
3.5	237
4.0	202
4.5	165
5.0	152
5.5	161
6.0	119
6.5	133
7.0	207
7.3	125
7.4	94
7.5	98
7.6	311
8.0	311
8.5	436
9.0	254
9.5	287
10	411

Tableau C.1 Évolution du nombre d'itérations en fonction du coefficient " a".