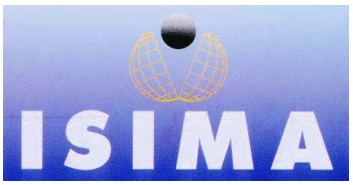
Le Calcul du Gradient d'Erreur dans les Réseaux de Neurones : Applications aux Telecom et aux Sciences Environnementales

Alexandre Aussem



UNIVERSITÉ BLAISE PASCAL
CLERMONT-FERRAND II
Laboratoire d'Informatique, de Modélisation
et d'Optimisation des Systèmes,
LIMOS (UMR 6158, CNRS)

T H E S E

présentée par

Alexandre Aussem

pour obtenir le diplôme :

HABILITATION A DIRIGER DES RECHERCHES

Spécialité : **Informatique**

Le Calcul du Gradient d'Erreur dans les Réseaux de Neurones Discrets Bouclés à Délais : Applications aux Télécom et aux Sciences Environnementales

Soutenue le 19 Décembre 2002 devant le jury composé de :

Président,	Alain Quilliot,	Professeur,	Univ. Blaise Pascal.
Rapporteurs,	Yoshua Bengio,	Professeur,	Univ. de Montréal.
	Younès Bennani,	Professeur,	Univ. Paris 13.
	Patrick Gallinari,	Professeur,	Univ. Paris 6.
	Erol Gelenbe,	Professeur,	Univ. Center Florida.
Examinateurs,	Stéphane Canu,	Professeur,	INSA Rouen.
	Philippe Mahey,	Professeur,	Univ. Blaise Pascal.
	Fionn Murtagh,	Professeur,	Queen's Univ. Belfast.
Invités,	Fabrice Chauvet,	Docteur,	Bouygues Telecom R& D.
	Patrice Kiener	Ingénieur,	NETRAL.

Table des matières

Remerciements

Je souhaite exprimer ma profonde gratitude aux membres du jury pour la confiance qu'ils m'ont témoignée et le travail considérable qu'ils ont consacré à la lecture exhaustive de ce long "mémoire de synthèse", et ce malgré les brefs délais impartis en cette période de surcharge notoire.

Je tiens également à adresser mes plus vifs remerciements au Professeur Alain Quilliot, Directeur du LIMOS et de l'ISIMA, pour avoir contribué à créer des conditions humaines et matérielles stimulantes, propices à l'exercice de mes activités d'enseignant-chercheur, dans un esprit de liberté et de confiance. Sa grande sagesse et ses qualités humaines ont été les meilleurs atouts durant ces années.

Je tiens enfin à adresser ma profonde sympathie à tous mes collègues de l'ISIMA et du LIMOS, qui ont su instaurer et préserver l'atmosphère chaleureuse et amicale qui règne dans notre communauté.

Chapitre 1

INTRODUCTION

1.1. Cadre de cette étude

Ce "document de synthèse" intitulé - Calcul du Gradient d'Erreur dans les Réseaux de Neurones Discrets Bouclés à Délais : Applications aux Télécom et aux Sciences Environnementales - dresse un panorama de mes travaux de recherche, entamés au cours de ma dernière année de thèse en 1995 et poursuivis depuis mon arrivée en 1996 en qualité de Maître de Conférences, jusqu'à aujourd'hui à l'Institut Supérieur d'Informatique, de Modélisation et de leurs Applications (ISIMA), école d'ingénieur rattachée à l'université Blaise Pascal (Clermont-Ferrand II). Mes recherches s'inscrivent dans le thème "Modélisation, Prévision et Décision" des systèmes biologiques, écologiques et environnementaux, au sein de l'axe "Informatique et Calcul de l'Aide à la Décision et Recherche Opérationnelle" du Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes (LIMOS, UMR 6158 CNRS).

Au delà de la description des contextes scientifiques dans lesquels ces travaux ont été conduits, cette présentation tente de restituer ce qui a constitué l'une des motivations essentielles de mon activité d'enseignant-chercheur : travailler à l'interface entre des disciplines connexes (traitement du signal, fouille de données, apprentissage statistique, algorithmique, filtrage adaptatif, ingénierie des réseaux télécom, etc.), s'enrichir et faire coopérer des savoirs, des personnalités, des compétences et des démarches distinctes. C'est dans cet esprit qu'est organisé ce document. Après quelques chapitres relativement théoriques sur le calcul algorithmique du gradient d'erreur dans les réseaux bouclés, le texte s'efforce, non pas de restituer les détails de mise en oeuvre des différentes méthodes employées pour la prévision, mais d'identifier un certain nombre de nouvelles thématiques prometteuses, à cheval entre plusieurs disciplines. Le lecteur peu

enclin à se pencher sur les premiers chapitres techniques peut directement accéder aux chapitres applicatifs, et passer de l'un à l'autre à sa guise.

1.2. Présentation des travaux de recherches

Il est communément admis depuis la contribution majeure de Hebb en 1949 [HEB 49] que l'apprentissage dans les systèmes biologiques résulte de la modification progressive des synapses. Ces modifications sont le résultat de mécanismes électrochimiques dans l'environnement immédiat de la synapse ; les opérations sont *locales*. La modification synaptique et le comportement collectif que l'on souhaite enseigner au réseau sont deux processus qui opèrent à des niveaux hiérarchiques distincts. La modification synaptique n'a pas connaissance de la tâche globale que le système cherche à apprendre. Dés lors, selon quels principes faut-il régir les modifications synaptique locales pour faire émerger collectivement le comportement complexe que l'on souhaite enseigner au réseau ? La question taraude depuis ces vingt dernière années l'esprit des chercheurs dans la communauté connexioniste. Deux éléments complémentaires de réponse ont vu le jour dans la littérature.

L'idée de Hebb est la suivante : lorsque l'activité de deux neurones connectés est corrélée positivement dans le temps, le poids de la synapse qui les unit doit être renforcé et vice-versa. En dépit de la singulière simplicité de cette vague formulation et ses multiples déclinaisons [OJA 82, SAN 89], de nombreux travaux ont montré les correspondances fécondes de ce principe d'auto-organisation avec l'analyse (statistique) en correspondances principales (PCA) [HAY 94] et la théorie de l'information [LIN 89].

La *descente du gradient* est la seconde idée majeure qui suscita une extraordinaire résurgence des réseaux de neurones sur la scène de l'intelligence artificielle, en proposant un principe pour guider l'organisation globale des modifications synaptiques. L'implémentation algorithmique particuliérement attractive de ce principe aux réseaux de neurones multi-couches est l'*algorithme de rétro-propagation du gradient* (backprop) inventé à l'origine par Werbos en 1974 [WER 74], puis redécouvert indépendamment en 1985 par Rumelhart [RUM 86] et d'autres. Levant définitivement les limitations du Perceptron de Rosenblatt, cet algorithme élégant préfigura le renouveau du connexionisme dans les années 1980 en cristallisant les énergies autour d'une voie nouvelle. De part sa simplicité d'usage, l'outil connexioniste désormais banalisé, a servi de pâture aux chercheurs de tous horizons scientifiques comme en attestent les nombreuses applications qui ont vu le jour dans le domaine des sciences de l'ingénieur où des avancées fructueuses ont été accomplies, et dont la portée s'étend la reconnaissance de l'écriture manuscrite cursive [GAR 96] à la commande optimale

de processus physico-chimiques complexes, de la modélisation de procédé industriel au contrôle de l'actionneur hydraulique d'un bras de robot [DRE 02] en passant par les prédictions environnementales. L'essor du connexionisme a aussi entraîné dans son sillage une myriade de produits commerciaux couronnés de succès parmi lesquels les ordinateurs à crayon optique, l'analyse de séquence ADN, la détection de fraude dans les transactions bancaires etc.

Le traitement de séquences temporelles nécessite toutefois l'introduction de délais [VRI 92, ELM 90, JOR 92, WAI 89, WAN 93, DAY 93] dans les transmissions synaptiques. Le *time-delay neural network* (TDNN) [WAI 89] dans lequel les connexions entre couches sont retardées, a été une des premières extensions apportées au réseau non bouclé statique pour le traitement de la parole. Depuis, une multitude de modèles localement ou globalement bouclés [TSO 94], discrets ou continus [BAL 95], à délais fixes ou ajustables [BOD 90, DAY 93] ont vu le jour. Toutefois, les modèles non bouclés ne peuvent rendre compte correctement des processus non-linéaires qui admettent une représentation d'état et dont les observations sont entachées d'un bruit de sortie. En effet, l'identification de système dynamique n'est envisageable avec des réseaux non-bouclés que lorsque toutes les variables d'état du système dynamique sous-jacent sont mesurées [NAR 91, SRI 94, DRE 02]. Ce n'est pas toujours le cas bien entendu. C'est pourquoi ce document passe en revue les principaux algorithmiques du calcul du gradient d'erreur dédiés aux réseaux de neurones discrets bouclés à délais, sous l'angle de la complexité en temps et en espace mémoire, et de la facilité de mise en oeuvre (e.g. localité des opérations, implémentation temps réel, stabilité numérique, calcul du gradient exact/approché, etc). Une fois identifié un algorithme de complexité et de mise en oeuvre attrayante, ce document se poursuit par une présentation des applications des réseaux bouclés à la simulation, à la prévision et à la segmentation de séries temporelles, réalisées dans le cadre de mes projets de recherche menés ces dernières années au LIMOS.

1.3. Organisation et contenu du mémoire

Donnons à présent un bref aperçu du contenu des différents chapitres.

Le **chapitre 2** présente les différentes implémentations pratiques de l'idée de la descente du gradient. Les algorithmes saillants qui sont parus dans la littérature ces dix dernières années, à savoir la rétro-propagation (BP) classique et la rétro-propagation dans le temps (BPTT) [RUM 86, WER 90], la rétro-propagation récurrente pour des réseaux statiques récurrents [ALM 87, PIN 87], la rétro-propagation temporelle pour les réseaux FIR non-bouclés [WAN 93], l'algorithme 'real-time recurrent learning' (RTRL) [WIL 89], le 'fast forward propagation' (FFP) [TOO 92], l'approche par fonction de Green (GF)

[SUN 92], et l'approche 'block-update' (BU) [SCH 92], sont énumérés ici dans un nouveau cadre formel unificateur et examinés au vu de leur complexité. Ces derniers ont été introduits pour des architectures de réseau spécifiques (temps continu/discret, modèle additif/d'ordre supérieur, statiques/à délais etc) pour des problèmes particuliers (apprentissage de point fixe, apprentissage de trajectoire) en usant de techniques variées (calcul variationnel, méthode d'adjoint, intégration numérique etc.).

Le **chapitre 3** établit les versions *forward* (FP) et *backward* (BPTT) du calcul du gradient pour une classe plus générale de d'architectures à délais pour l'apprentissage de points fixes et l'apprentissage de trajectoires : les réseaux FIR bouclés. Ce sont des réseaux discrets bouclés à délais dont les synapses sont représentées par des filtres linéaires à réponse impulsionnelle finie (FIR) : des connexions arbitrairement retardées et bouclées sont autorisées entre les neurones. Cette architecture générale porte le nom de "Dynamical Recurrent Neural Networks" (DRNN) [AUS 95b, AUS 02b], et fédère un grand nombre d'architectures localement et globalement récurrentes proposées dans la littérature pour le traitement temporel (voir par exemple [KRE 01, PIC 94, TSO 94, BAL 95, CAM 99, DUR 99, WAN 93, WIL 89]) ainsi que les réseaux bouclés à point fixe [ALM 87, PIN 87]. Des conditions suffisantes garantissant l'existence, l'unicité et la stabilité asymptotique du point fixe ainsi que la stabilité asymptotique du réseau en bouclage fermé sont établis.

Au **chapitre 4**, il est montré pourquoi les réseaux bouclés sont réputés incapables d'apprendre des dépendances à longue portée, même élémentaires. Le problème du *temporal credit assignment* demeure l'une des thématique de recherche de la communauté connexioniste comme en témoigne une récente taxonomie sur les *réseaux connexionistes spatio-temporels* (STCN) [KRE 01]. En effet, la *décroissance rapide du flot arrière du gradient d"erreur* ("gradient error back flow" et notée **GEBF**), rend quasiment impossible l'apprentissage de dépendances à longue portée entre les entrées/sorties par des méthodes fondées sur le gradient. Cette faiblesse qualifiée de *forgetting behavior*, est au coeur des préoccupations d'un grand nombre de travaux depuis l'article de Bengio et al. [BEN 94b]. Dans ce chapitre, l'analyse de l'GEBF étend les travaux de [FRA 92, BEN 94b, AUS 95b, HOC 97b, LIN 96] au réseaux FIR bouclés, y compris les réseaux à point fixe, et apporte un éclairage nouveau sur la difficulté de la descente du gradient à capturer des contingences temporelles à longue portée. Des conditions suffisantes pour garantir la convergence de l'EGBF sont établies. Celles-ci s'expriment explicitement en fonction de la matrice de poids et s'appliquent à de nombreux réseaux bouclés introduits dans la littérature ces dernières années [KRE 01, TSO 94]. A la lumière de ce résultat, une borne supérieure sur le nombre de rétro-propagations dans le temps est établie pour limiter l'erreur de (BPTT). Ces résultats théoriques sont confirmés par simulation.

Le **chapitre 5** illustre les aptitudes des réseaux bouclés à délais dans le domaine de la simulation et de la prévision à court terme de séries temporelles issues des Sciences Environnementales au sens large. Dans un premier temps, des réseaux bouclés à délais sont entraînés comme prédicteurs à un pas sur des suites chaotiques synthétiques en délivrant au réseau de neurones, une information tronquée du vecteur d'état du système. Il incombe au modèle la lourde tâche d'inférer les variables d'état cachées du système à chaque instant. Une fois itéré sur lui-même en bouclage fermé, le réseau de neurone est le siège d'un comportement chaotique comme en témoigne les attracteurs reconstruits.

Fort de ces observations, plusieurs applications ont été réalisées depuis 1996 dans le domaine des sciences environnementales. Je présente en premier lieu un travail visant à coupler un modèle de simulation numérique de la circulation océanique avec des réseaux de neurones afin de prédire la température à la surface de la mer (Sea Surface Temperature, SST) sous forme de cartes 2D quelques jours à l'avance, dans une zone maritime où l'on observe des mouvement ascendants d'eau froide, que l'on désigne par le phénoméne d'*upwelling*. Ce travail a été mené au cours des années 1998 et 1999 dans le cadre d'un projet de recherche avec Marc Fuentes, actuellement en thèse à l'université de Montréal, avec le Marine Environment Unit au (European) Joint Research Center JRC à ISPRA (Italie).

Dans un second temps, je dresse les grandes lignes d'un projet de recherche mené avec le European Southern Observatory (ESO) au cours des années 1999 et 2001 en collaboration avec Germain Tran (Ingénieur ISIMA) et Marc Sarazin (ESO), qui traite de la prévision à court terme des *fluctuations* d'une mesure de la diffraction des ondes lumineuses dûe aux perturbations atmosphériques : le *seeing*. Ses fluctuations sont d'une importance majeure pour les astronomes parce que les opérations de calibrage des télescopes sont menées quelque temps avant l'observation.

Ce paragraphe applicatif s'achève sur l'esquisse d'un travail mené avec David Hill (LIMOS), dans le cadre d'un projet LIFE "Control of the spread of the *Caulerpa Taxifolia* in the Mediterranean" (programme DG XI) consacré à la prévision de la surface contaminée par la caulerpe aprés plusieurs années dans la bassin méditerranéen. Le principe d'entraîner un réseau de neurones grâce aux traces issues des réplications des simulations stochastiques est qualifié dans la littérature par le terme de *méta-modélisation* [KIL 94]. Le réseau de neurones permet, au terme de l'apprentissage, un gain en temps de calcul considérable puisqu'il permet d'anticiper l'évolution d'un système stochastique complexe (le modèle), par un processus déterministe plus simple (le méta-modéle).

Le **chapitre 6** présente une méthode hybride pour prédire les séries temporelles à mémoire longue par l'utilisation d'une communauté de réseaux opérant sur des échelles temps-fréquences distinctes. Ce travail initié par Fionn Murtagh

(Univ. Belfast) et moi-même en 1996 est fondé sur une analyse multirésolution de la suite de manière à contourner le problème de l'*évanouissement du gradient* dans des réseaux de neurones. Une décomposition en ondelettes discrète est effectuée par l'algorithme dit "à trous". Chaque échelle, qualifiée d'*octave*, est alors traitée individuellement par un réseau de neurones afin de fournir une estimation des futurs coefficients d'ondelettes. Ces derniers sont alors recombinés pour fournir la prédiction finale du modèle. Cette technique est ici illustrée sur un problème de prévision, une minute à l'avance, du volume de données téléchargés sur un serveur Web. Depuis l'année 2002, une collaboration avec Patrice Abry (Lab. de Physique, ENS Lyon) spécialiste des lois d'échelles, et Pierre Chainais (LIMOS) est menée dans ce sens pour caractériser et prédire le comportement du télétrafic. Cette collaboration s'inscrit dans le cadre de l'Action Spécifique 'Métrologie Internet" du CNRS qui a débutté fin 2002.

Le **chapitre 7** fait état des derniers développements d'un projet au long cours dédié à la gestion des ressources dans un réseau télécom multiservice, initié par Erol Gelenbe et moi-même au milieu des années 90 [AUS 94b], poursuivi par plusieurs stagiaires de DEA successifs [AUS 94b, AUS 99c] au LIMOS en collaboration avec Raymond Marie (IRISA), et enfin repris par Antoine Mahul (dont j'encadre la thèse au LIMOS depuis septembre 2000) dans le cadre du projet RNRT OPIUM (Optimisation de la Planification des Infrastructures des réseaUx Mobiles). Ce projet, dont la partie routage incombe au LIMOS, vise à offrir une solution intégrée pour la planification et l'optimisation de réseaux de télécommunications mobiles. Le travail d'Antoine Mahul a pour objet de substituer *in fine* à la formule M/M/1 classique dans le code de l'algorithme d'optimisation multiflots développé par Philippe Mahey et Christophe Duhamel, un réseau de neurones entraîné par simulation pour prédire la QoS en chaque noeud en termes de délai de de perte.

Le **chapitre 8** présente un modèle auto-régressif non-linéaire à changement de régime markovien pour la *segmentation* de séries temporelles stationnaires par morceaux. La segmentation opère *en amont* de la prédiction, en ce sens qu'il est assez aisé de construire ultérieurement un prédicteur à partir du comité des experts. Ce travail, initié en 1999 à la lecture d'un article de J. Kohlmorgen et al. [KOH 99], estime les paramètres par le principe du maximum de vraisemblance ; un algorithme EM *off-line* est employé pour l'estimation des paramètres du modèle, en particulier les paramètres des réseaux de neurones, les probabilités de transitions et la variance du bruit. Des exemples d'application sont présentés sur des données artificielles et financières menées en collaboration avec Marc Fuentes (doctorant à l'Univ. Montréal) et Corinne Boutevin (doctorante au LIMOS). Notons qu'une version *on-line* de cet algorithme a fait l'objet d'une thèse récente [RYN 00] basée sur une formulation plus astucieuse de la (log)vraisemblance. Ce modèle hybride HMM/MLP présente, au demeurant, de profondes similitudes avec ceux employés actuellement dans la reconnaissance

de l'écriture cursive [GAR 96].

Et enfin, le **chapitre 9** dresse une synthèse des travaux présentés dans ce document et dégage, en guise de conclusion, les nouvelles thématiques de recherche que je souhaite aborder au cours de ces prochaines années.

1.4. Avis au lecteur

Ce "document de synthèse" n'est nullement un ouvrage didactique, ni une taxonomie des réseaux de neurones bouclés à délais. Les trois premiers chapitres rendent exhaustivement compte de mon travail de recherche - plus théorique - entrepris depuis ma thèse en 1995 sur les réseaux récurrents. La seconde partie constitués des chapitres 4, 5, 6, 7 et 8 aborde des applications diverses et variées, sans parfois de dénominateur commun, ni référence à la première partie.

Ce document évacue (littéralement) un certain nombre de problématiques *essentielles* comme la régularisation dite *formelle* (au sens de Tikhonov, par l'adjonction d'un terme de pénalisation) et la régularisation dite *structurelle* (élimination de connexions jugées superflues, les techniques d'identification presque sûre du "vrai modèle") du réseau, en dépit du grand nombre de paramètres ajustables. Les méthodes pour le calcul d'intervalles de confiance associés aux prédictions sont également omises ainsi que les méthodes pour approximer la matrice hessienne de la sortie du modèle par rapport à ses paramètres. Par ailleurs, le détail des modes opératoires, la description des données, le choix des architectures, la mise en oeuvre exacte de la validation croisée, etc. figurent dans les articles, accessible en ligne sur la page www.isima.fr/aussem.

Les trois premiers chapitres sont dédiés uniquement au calcul du gradient d'erreur dans les réseau bouclés. Les *algorithmes* d'apprentissage, à proprement parler, ne sont pas présentés par concision. On les trouvera en dans les premiers chapitres de toutes les thèses du domaine (voir par exemple [AUS 96, GOU 97, MAN 95, RYN 00]).

Le lecteur désireux de se plonger plus en détail dans les questions relatives au filtrage adaptatif, aux architectures et aux problèmes de stabilité des réseaux bouclés, pourra consulter l'ouvrage récent de D.P. Mandic et J.A. Chambers consacré aux réseaux récurrents pour la prédiction [MAN 01]. Les problèmes de régularisation - orientés Statistique - sont traités plus en détail, par exemple, dans les thèses de Morgan Mangeas [MAN 95] et de Cyril Goutte [GOU 97]. Parmi les ouvrages récents qui s'adressent à un public plus large, on pourra consulter [BIS 95, GOL 96, HAY 94], ainsi que [DRE 02] en langue française.

Je m'excuse auprès du lecteur pour les imperfections, les erreurs, et coquilles qui subsistent encore dans cette version du document malgré tous mes efforts.

Chapitre 2

APPRENTISSAGE DES RESEAUX BOUCLES STANDARDS

2.1. Introduction

Ce chapitre présente un cadre unificateur pour le calcul du gradient d'erreur dans les réseaux de neurones bouclés opérant en temps discret. Le gradient d'erreur servira directement ou indirectement (e.g. algorithmes pseudo Newton) à l'ajustement des poids, au calcul d'intervalles de confiance, à l'approximation du hessien, etc. [BIS 95]. Ce cadre formel fondé sur la théorie de la commande optimale [ATI 00, BRY 75], est dédié aux réseaux bouclés discrets dits de Williams et Zipzer [WIL 89]), pour lesquels chaque synapse est retardée d'une unité de temps. Il fédère les principales méthodes algorithmiques mises en oeuvre pour le calcul du gradient d'erreur, plus spécifiquement : 'forward propagation algorithm' (FP) ou 'real-time recurrent learning' (RTRL) [WIL 89] , 'back-propagation through time' (BPTT) [RUM 86, WER 90], 'fast forward propagation' (FFP) [TOO 92], Green's function approach' (GF) [SUN 92], et l'approche 'block-update' (BU) [SCH 92].

La neuro-dynamique du modèle bouclé standard ([WIL 89]) est régie par les équations

$$\mathbf{v}_k = g(\mathbf{W}^T \mathbf{v}_{k-1}) + \mathbf{i}_k, \quad \forall k = 1, \ldots, K, \tag{2.1}$$

où $\mathbf{v}_k$ désigne le vecteur d'état du réseau à l'instant k et K désigne la durée de l'époque (voir les notations en Table 2.1). On suppose que le réseau est constitué de N neurones totalement bouclés à réponse continue dans $[0, 1]$. w_{ij} représente le poids de la connexion retardée d'une unités de temps, reliant le neurone i au neurone j. La fonction d'activation $g()$ est supposée non-linéaire,

bornée et dérivable sur $]-\infty,\infty[$. La famille de fonctions d'activation continuement dérivables *sigmoïdes*, caractérisée $\forall c,k,r \in \mathbb{R}$ et $c,k>0$ par

$$\sigma_{c,k,r}(x) = c\frac{e^{kx}-1}{e^{kx}+1} + r, \tag{2.2}$$

est la plus utilisée [MAN 01]. La fonction logistique, $g(s) = 1/(1+e^{-s})$ est une fonction sigmoïde particulière définie pour $c = 1/2, k = 1, r = 1/2$. Par souci de généralité, $g()$ sera supposée dans la suite appartenir à la famille des *sigmoïdes* et on posera $\mu = \max_u(g'(u))$. On notera d'ores et déjà que toute fonction σ issue de cette famille est lipschitzienne et que sa dérivée d'ordre $m \geq 1$ quelconque est encore lipschitzienne. Le Perceptron étant une combinaison linéaire de fonctions sigmoïdes [MAN 95], il implémente donc une fonction lipschitzienne. Cette remarque sera utile lorsqu'il s'agira au Chapitre 4 d'exhiber des conditions suffisantes pour assurer la stabilité asymptotique du réseau en bouclage fermé.

Note : les fonctions montonones croissantes bornées de type sigmoïde émergent naturellement lorsque, dans un cadre bayésien, la distribution des entrées est prise en compte dans le terme de régularisation [CAN 99]. Néanmoins, les propriétés d'approximation des RN s'étendent à d'autres fonctions d'activation, en particluier les fonctions gaussiennes (RBF), polynomiales et rationnelles [POG 90, MAN 01]. A ce titre, le lecteur curieux pourra consulter un type de RN à fonction d'activation rationnelle [GEL 91, GEL 99, GEL 02] présentant une analogie élégante entre un réseau de neurones et un réseau de files d'attente visitées par des clients dits *positifs* et *négatifs*. L'activation du neurone est interprétée comme le taux d'occupation de la file en régime stationnaire, lequel s'exprime sous la forme d'une fonction rationnelle.

Comme à l'ordinaire, un sous-ensemble des neurones est dédié à la réception et à la propagation du vecteur d'entrée à travers le réseau. Ces neurones portent le nom de neurones d'entrée et possèdent une activation fixée par les composantes du vecteur d'entrée $\mathbf{i}_k$. Hormis les signaux externes, les neurones d'entrée ne reçoivent pas de signaux émanant d'autres neurones, i.e., $w_{ij} = 0, \forall i$, si j désigne un neurone d'entrée. D'une façon similaire, certains neurones, dits de sortie, possèdent une activation cible, ou valeur désirée, $\mathbf{d}_k$. Les neurones n'ayant pas de relation avec le monde extérieur sont les neurones dits cachés. Enfin, le rôle de biais incombe par définition au neurone d'indice 0. Ce dernier est perçu comme une neurone d'entrée supplémentaire dont l'activation est fixée à 1.0. Pour des raisons de commodité, nous ne distinguerons pas, dans les formules suivantes, les classes de neurones que nous venons d'introduire. Les notations sont exposées dans la Table 2.1.

$\mathbf{v}_k$	vecteur d'activation
$\mathbf{u}_k$	vecteur des entrées internes
$\mathbf{i}_k$	vecteur des entrées externes
w_{ij}	poids de la connexion entre le neurone i et le neurone j
$\mathbf{w}(j) = [w_{1j}, \ldots, w_{Nj}]^T$	vecteur de poids vers le neurone j
$\mathbf{W} = [\mathbf{w}(1), \mathbf{w}(2), \ldots, \mathbf{w}(N)]$	matrice de poids
$g()$	fonction d'activation sigmoïde
μ	$\max_u(g'(u))$
$\mathbf{G}'_k$	$N \times N$ matrice diagonale de $g'(u_k(j))$

Tableau 2.1. *Notations employées. k désigne l'indice de temps.*

L'objet de l'apprentissage est d'ajuster les matrice de poids $\mathbf{W}$ de façon à faire évoluer le réseau, sous l'action des entrées $\{\mathbf{i}_k\}$, d'un état $\mathbf{v}_0$ vers une suite de points fixes dont les composantes de sortie s'approchent des valeurs désirées $\{\mathbf{d}_k\}$. Comme à l'accoutumée, on cherche à minimiser l'erreur quadratique

$$E = \frac{1}{2} \sum_{k=0}^{K} \mathbf{e}_k^T \mathbf{e}_k , \tag{2.3}$$

où $\mathbf{e}_k = \mathbf{d}_k - \mathbf{v}_k$ mesure le vecteur d'erreur le long de la **trajectoire** (ou époque) de l'itération $k = 0$ à K. Rappelons que $e_k(i) = d_k(i) = 0$ si le neurone i n'est pas un neurone de sortie. Les entrées sont omises (i.e. $\mathbf{i}_k = 0$) dans un premier temps pour simplifier les calculs.

Dans ce chapitre, le calcul du gradient d'erreur, $\frac{\partial E}{\partial \mathbf{W}}$, est effectué en fin de trajectoire. Ce dernier servira par exemple à l'estimation des paramètres (e.g. descente du gradient, pseudo-Newton, LM, etc.), à l'approximation de l'inverse du hessien à la régularisation, ou encore au calcul d'intervalles de confiance [BIS 95].

2.2. Notations et rappels

Introduisons quelques rappels et notations concernant l'expansion des dérivées partielles dans les systèmes d'équations ordonnées [PIC 94, WER 90]. Considérons un ensemble de n variables $\mathbf{z}_1, \ldots, \mathbf{z}_n$ dont les valeurs sont déterminées par un ensemble de n équations *ordonnées*

$$\mathbf{z}_i = f_i(\mathbf{z}_1, \ldots, \mathbf{z}_{i-2}, \mathbf{z}_{i-1}) \tag{2.4}$$

dans lesquelles chaque variable $\mathbf{z}_i$ est fonction de $\mathbf{z}_1, \ldots, \mathbf{z}_{i-1}$. Pour permettre le calcul des dérivées partielles, il faut spécifier les variables assimilées à des constantes, des autres. Typiquement, lorsque rien n'est spécifié, nous supposerons que ces variables sont maintenues constantes, excepté celles qui apparaissent au dénominateur des dérivées partielles. Une dérivée partielle ordonnée est une dérivée partielle pour laquelle les variables constantes sont déterminées grâce à un ensemble d'équations ordonnées. Selon les notations mathématiques consacrées [PIC 94], on a

$$\frac{\partial^+ z_j}{\partial z_i} = \left[\frac{\partial z_j}{\partial z_i}\right]_{\{z_1,\ldots,z_{i-1}\}}. \tag{2.5}$$

Il vient les relation suivantes,

$$\frac{\partial^+ \mathbf{z}_{i+1}}{\partial \mathbf{z}_i} = \frac{\partial \mathbf{z}_{i+1}}{\partial \mathbf{z}_i}, \tag{2.6}$$

et

$$\frac{\partial^+ \mathbf{z}_j}{\partial \mathbf{z}_i} = 0 \quad \text{pour} \quad j < i. \tag{2.7}$$

Lorsque $j > i+1$, les dérivées ordonnées s'obtiennent par les lois d'expansion suivantes

$$\frac{\partial^+ \mathbf{z}_j}{\partial \mathbf{z}_i} = \frac{\partial \mathbf{z}_j}{\partial \mathbf{z}_i} + \sum_{k=i+1}^{j-1} \frac{\partial^+ \mathbf{z}_j}{\partial \mathbf{z}_k} \frac{\partial \mathbf{z}_k}{\partial \mathbf{z}_i}, \tag{2.8}$$

et

$$\frac{\partial^+ \mathbf{z}_j}{\partial \mathbf{z}_i} = \frac{\partial \mathbf{z}_j}{\partial \mathbf{z}_i} + \sum_{k=i+1}^{j-1} \frac{\partial \mathbf{z}_j}{\partial \mathbf{z}_k} \frac{\partial^+ \mathbf{z}_k}{\partial \mathbf{z}_i}, \tag{2.9}$$

Selon le cas, j'opterai pour l'une ou l'autre des lois d'expansion (2.8) et (2.9) pour le calcul de $\frac{\partial^+ E}{\partial \mathbf{W}}$. Avant de procéder au calcul du gradient, un commentaire sur les notations mathématiques employées. Je suppose implicitement par la suite que la dérivée partielle $\partial \mathbf{u}/\partial \mathbf{v}$, où $\mathbf{u}$ et $\mathbf{v}$ sont des vecteurs de taille N et M respectivement est la matrice jacobienne de dimension $N \times M$. Par ailleurs, $\frac{\partial^+ E}{\partial \mathbf{W}}$ désignera la matrice de composants $\frac{\partial^+ E}{\partial w_{ij}}$.

2.3. Un cadre unificateur

Ce chapitre fédère les principales méthodes algorithmiques mises en oeuvre, dans la littérature, pour calculer le gradient d'erreur dans les réseaux bouclés en se fondant sur des éléments de théorie de la commande optimale [ATI 00, BRY 75] et de programmation dynamique. Pour commencer, formulons le problème de l'identification des paramètres du modèle sous la forme d'un problème de minimisation sous contraintes égalité

$$\begin{cases} \text{Minimiser } E \\ \text{sous les contraintes } \mathbf{h}_k = g(\mathbf{W}^T \mathbf{v}_{k-1}) - \mathbf{v}_k = \mathbf{0}, \quad k = 1, \ldots, K. \end{cases} \tag{2.10}$$

Les poids sont les variables de contrôle ou de décision dans la terminologie de la commande optimale [BRY 75]. Les $\mathbf{v}_k$ sont les variables d'état, dont les valeurs sont déterminées par les contraintes $\mathbf{h}_j$. Arrangeons les colonnes $\mathbf{w}(i)$ de $\mathbf{W}$ en un long vecteur colonne, ainsi que $\mathbf{v}$ et le vecteur de contraintes $\mathbf{h}$

$$\mathbf{w} = \begin{pmatrix} \mathbf{w}(1) \\ \vdots \\ \mathbf{w}(N) \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} \mathbf{v}_1^t \\ \vdots \\ \mathbf{v}_N^t \end{pmatrix} \quad \text{et} \quad \mathbf{h} = \begin{pmatrix} \mathbf{h}_1^t \\ \vdots \\ \mathbf{h}_N^t \end{pmatrix}. \tag{2.11}$$

Pour clarifier les dépendances, écrivons E par $E(\mathbf{v}(\mathbf{w}))$ et $\mathbf{h}$ par $\mathbf{h}(\mathbf{v}(\mathbf{w}), \mathbf{w})$. Appliquons (2.9)

$$\frac{\partial^+ E(\mathbf{v}(\mathbf{w}))}{\partial \mathbf{w}} = \frac{\partial E(\mathbf{v}(\mathbf{w}))}{\partial \mathbf{w}} + \frac{\partial E(\mathbf{v}(\mathbf{w}))}{\partial \mathbf{v}} \frac{\partial^+ \mathbf{v}(\mathbf{w})}{\partial \mathbf{w}}. \tag{2.12}$$

où $\partial^+ E(\mathbf{v}(\mathbf{w}))/\partial \mathbf{w}$ est nulle car les poids n'apparaissent pas explicitement dans l'expression de E. Appliquons maintenant (2.9) au calcul de $\mathbf{h}(\mathbf{v}(\mathbf{w}), \mathbf{w}) = 0$, on obtient

$$\frac{\partial \mathbf{h}(\mathbf{w}, \mathbf{v}(\mathbf{w}))}{\partial \mathbf{w}} + \frac{\partial \mathbf{h}(\mathbf{w}, \mathbf{v}(\mathbf{w}))}{\partial \mathbf{v}} \frac{\partial^+ \mathbf{v}(\mathbf{w})}{\partial \mathbf{w}} = 0. \tag{2.13}$$

En combinant (2.12) et (2.13), il vient

$$\frac{\partial^+ E(\mathbf{v}(\mathbf{w}))}{\partial \mathbf{w}} = -\frac{\partial E(\mathbf{v}(\mathbf{w}))}{\partial \mathbf{v}} \left(\frac{\partial \mathbf{h}(\mathbf{w}, \mathbf{v}(\mathbf{w}))}{\partial \mathbf{v}} \right)^{-1} \frac{\partial \mathbf{h}(\mathbf{w}, \mathbf{v}(\mathbf{w}))}{\partial \mathbf{w}} \tag{2.14}$$

On retrouve ces équations dans [BRY 75]. Par concision, les dépendances explicites aux variables sont omises. Ainsi, nous obtenons l'expression matricielle

$$\frac{\partial^+ E}{\partial \mathbf{w}} = -\frac{\partial E}{\partial \mathbf{v}} \left(\frac{\partial \mathbf{h}}{\partial \mathbf{v}}\right)^{-1} \frac{\partial \mathbf{h}}{\partial \mathbf{w}} \tag{2.15}$$

C'est l'**équation de base qui unifie tous les algorithmes**. $\partial \mathbf{h}/\partial \mathbf{v}$ désigne la matrice d'éléments $\partial h_i/\partial v_j$. L'évaluation des matrices dans (2.15) est la suivante :

$$\frac{\partial E}{\partial \mathbf{v}} = (\mathbf{e}_1^t, \ldots, \mathbf{e}_K^t), \tag{2.16}$$

$$\frac{\partial \mathbf{h}}{\partial \mathbf{v}} = \begin{pmatrix} -\mathbf{I} & 0 & 0 & \cdots & 0 \\ \mathbf{G}_2'\mathbf{W}^T & -\mathbf{I} & 0 & \cdots & 0 \\ 0 & \mathbf{G}_3'\mathbf{W}^T & -\mathbf{I} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{G}_K'\mathbf{W}^T & -\mathbf{I} \end{pmatrix} \tag{2.17}$$

où $\mathbf{G'}_k$ est donnée par $\mathbf{G'}_k = 2\beta \mathbf{G}_k(1 - \mathbf{G}_k)$ avec $\mathbf{G}_k$ définie comme la matrice diagonale $N \times N$ construite à partir de $g(s_k(j))$ pour $j = 1, \ldots, N$. L'inverse de $\partial \mathbf{h}/\partial \mathbf{v}$ s'exprime explicitement

$$\left(\frac{\partial \mathbf{h}}{\partial \mathbf{v}}\right)^{-1} =$$

$$\begin{pmatrix} \mathbf{I} & 0 & 0 & \cdots & 0 \\ \mathbf{G}_2'\mathbf{W}^T & \mathbf{I} & 0 & \cdots & 0 \\ \mathbf{G}_3'\mathbf{W}^T\mathbf{G}_2\mathbf{W}^T & \mathbf{G}_3'\mathbf{W}^T & \mathbf{I} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{G}_K'\mathbf{W}^T \cdots \mathbf{G}_2'\mathbf{W}^T & \mathbf{G}_K'\mathbf{W}^T \cdots \mathbf{G}_3'\mathbf{W}^T & \mathbf{G}_K'\mathbf{W}^T \cdots \mathbf{G}_4'\mathbf{W}^T & \cdots & \mathbf{I} \end{pmatrix}. \tag{2.18}$$

Il vient également

$$\frac{\partial \mathbf{h}}{\partial \mathbf{w}} = \begin{pmatrix} \mathbf{G}_1'\mathbf{V}_0 \\ \mathbf{G}_2'\mathbf{V}_1 \\ \vdots \\ \mathbf{G}_K'\mathbf{V}_{K-1} \end{pmatrix}. \tag{2.19}$$

avec

$$\mathbf{V}_k = \begin{pmatrix} \mathbf{v}_k^t & 0 & \cdots & 0 \\ 0 & \mathbf{v}_k^t & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{v}_k^t \end{pmatrix}. \tag{2.20}$$

Dans la suite, $\partial E/\partial \mathbf{W}$ désignera la matrice de composants $\partial E/\partial w_{ij}$.

2.4. La propagation en avant (FP)

Posons $\mathbf{Y} = (\partial \mathbf{h}/\partial \mathbf{v})^{-1}\partial \mathbf{h}/\partial \mathbf{w}$. Cette grandeur peut s'écrire en blocs

$$\mathbf{Y} = \begin{pmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \\ \vdots \\ \mathbf{Y}_K \end{pmatrix} \tag{2.21}$$

où $\mathbf{Y}_k$ est une matrice $N \times N^2$. D'après $(\partial \mathbf{h}/\partial \mathbf{v})\mathbf{Y} = \partial \mathbf{h}/\partial \mathbf{w}$, il vient

$$\mathbf{Y}_k = \mathbf{G}_k' \mathbf{W}^T \mathbf{Y}_{k-1} - \mathbf{G}_k' \mathbf{V}_{k-1}, \quad k = 2, \ldots, K. \tag{2.22}$$

avec les conditions aux limites

$$\mathbf{Y}_1 = -\mathbf{G}_1' \mathbf{V}_1 \tag{2.23}$$

Donc dans l'approche 'forward', la récursion se fait dans le sens du temps et le gradient final est obtenu par

$$\frac{\partial^+ E}{\partial \mathbf{w}} = -\frac{\partial E}{\partial \mathbf{v}}\mathbf{Y} = -\sum_{k=1}^{K} \mathbf{e}_k^T \mathbf{Y}_k. \tag{2.24}$$

Remarquons qu'en posant $\mathbf{Y} = (\partial \mathbf{h}/\partial \mathbf{v})^{-1}\partial \mathbf{h}/\partial \mathbf{W}$ (2.22) s'écrit

$$\mathbf{Y}_k = \mathbf{G}_k' \mathbf{W}^T \mathbf{Y}_{k-1} - \mathbf{G}_k' \cdot (\mathbf{v}_{k-1}, \ldots, \mathbf{v}_{k-1}). \tag{2.25}$$

avec les conditions aux limites

$$\mathbf{Y}_1 = -\mathbf{G}'_1(\mathbf{v}_1, \ldots, \mathbf{v}_1). \tag{2.26}$$

D'où une formulation matricielle que nous retrouverons par la suite

$$\frac{\partial^+ E}{\partial \mathbf{W}} = -\frac{\partial E}{\partial \mathbf{v}}\mathbf{Y} = -\sum_{k=1}^{K} \mathbf{e}_k^T \mathbf{Y}_k. \tag{2.27}$$

Cet algorithme à propagation avant porte le nom de real-time-recurrent learning algorithm (RTRL) [WIL 89]. Cette procédure est fortement grevée par la masse de calcul et de mémoire requise. En premier lieu, chaque dérivée doit être stockée ce qui entraîne une capacité de mémorisation de l'ordre de $O(N^3)$. Ensuite une quantité draconienne d'opérations, de l'ordre de $O(N^4)$ est nécessaire à chaque itération car l'adaptation les N^3 dérivées requiert chacune $O(N)$ opérations.

2.5. La rétro-propagation dans le temps (BPTT)

Dans la méthode BPTT, on évalue d'abord $\mathbf{y}^T = -\partial E/\partial \mathbf{v}(\partial \mathbf{h}/\partial \mathbf{v})^{-1}$ avant de le multiplier par la suite par $\partial \mathbf{h}/\partial \mathbf{w}$. Par définition,

$$\frac{\partial E}{\partial \mathbf{v}} = -\mathbf{y}^T \frac{\partial \mathbf{h}}{\partial \mathbf{v}}, \tag{2.28}$$

avec $\mathbf{y}^T = (\mathbf{y}_1^T, \ldots, \mathbf{y}_K^T)$. En substituant les expressions (2.16) et (2.17), il vient

$$\mathbf{y}_k = \mathbf{e}_k + \mathbf{W}\mathbf{G}'_{k+1}\mathbf{y}_{k+1}, \tag{2.29}$$

avec la condition aux limites $\mathbf{e}_K = \mathbf{y}_K$. D'après la définition de $\mathbf{y}$

$$\frac{\partial^+ E}{\partial \mathbf{w}} = \mathbf{y}^T \frac{\partial \mathbf{h}}{\partial \mathbf{w}}. \tag{2.30}$$

Substituons $\mathbf{y}$ dans cette l'expression et réarrangeons les poids en matrice $\mathbf{W}$, on obtient l'équation matricielle

$$\frac{\partial^+ E}{\partial \mathbf{W}} = \sum\nolimits_{k=1}^{K} \mathbf{G}'_k \mathbf{y}_k \mathbf{v}_{k-1}^T. \tag{2.31}$$

Posons $\boldsymbol{\delta}_k = \mathbf{G}'_k \mathbf{y}_k$, on retrouve une généralisation de la règle delta pour un réseau non-bouclé.

$$\frac{\partial^+ E}{\partial \mathbf{W}} = \sum_{k=1}^{K} \boldsymbol{\delta}_k \mathbf{v}_{k-1}^T. \tag{2.32}$$

La rétro-progation dans le temps (BPTT) est très efficace : sa version en mode *batch* est de l'ordre de $O(N^2)$. Toutefois, la mémoire requise varie en $O(K)$, il est peu pratique de rétro-propager le réseau complètement, la *troncature* du gradient est souvent inéluctable [WIL 90]. BPTT(h, h') est une version accélérée qui lance le réseau h pas en avant, et rétro-propage $h' > h$ pas en arrière le gradient, ajuste les poids et recommence.

2.6. La propagation en avant rapide (FFP)

Supposons connue la séquence des $\mathbf{y}_k$ obtenue par la formule (2.29) sur une trajectoire de longueur K, il n'est pas nécessaire de recalculer ces grandeurs lorsque le point $K+1$ est disponible. Il suffit de changer le sens de propagation. Pour cela il faut substituer la condition aux limites à l'instant $k = K$ par une condition à l'instant $k = 1$. La propagation en avant rapide (FFP) repose sur l'existence d'une expression explicite de $\left(\frac{\partial \mathbf{h}}{\partial \mathbf{v}}\right)^{-1}$, exprimée Eq. (2.18), et de $\mathbf{W}^{-1}$. Soit $\mathbf{y}_k^{(K)}$ la solution du système avec K points. En appliquant successivement Equation (2.29), on obtient

$$\mathbf{y}_k^{(K)} = \mathbf{A}_k \mathbf{y}_1^{(K)} + \mathbf{b}_k, \tag{2.33}$$

avec

$$\mathbf{A}_k = \mathbf{G'}_k^{-1} \mathbf{W}^{-1} \ldots \mathbf{G'}_2^{-1} \mathbf{W}^{-1}, \tag{2.34}$$

et

$$\begin{aligned} \mathbf{b}_k &= \mathbf{G'}_k^{-1} \mathbf{W}^{-1} \ldots \mathbf{G'}_2^{-1} \mathbf{W}^{-1} \mathbf{e}_1 \\ &+ \mathbf{G'}_k^{-1} \mathbf{W}^{-1} \ldots \mathbf{G'}_3^{-1} \mathbf{W}^{-1} \mathbf{e}_2 \\ &+ \ldots \\ &+ \mathbf{G'}_k^{-1} \mathbf{W}^{-1} \mathbf{e}_{k-1}. \end{aligned} \tag{2.35}$$

$\mathbf{A}_k$ et $\mathbf{b}_k$ se calculent par récurrence avant. Il reste à recalculer la condition initiale $\mathbf{y}_1^{(K)}$. Elle est obtenue grâce à l'inverse de $\partial\mathbf{h}/\partial\mathbf{v}$ et à la définition $\mathbf{y}^{(K)} = \partial E/\partial\mathbf{v}(\partial\mathbf{h}/\partial\mathbf{v})^{-1}$

$$\mathbf{y}_1^{(K)} = \mathbf{y}_1^{(K-1)} - \mathbf{W}\mathbf{G'}_2 \ldots \mathbf{W}\mathbf{G'}_K \mathbf{e}_K. \tag{2.36}$$

Sachant que

$$\frac{\partial^+ E}{\partial \mathbf{w}} = \sum_{k=1}^{K} \mathbf{G'}_k \mathbf{V}_{k-1} \mathbf{y}_k, \tag{2.37}$$

on en déduit

$$\frac{\partial^+ E}{\partial \mathbf{w}} = \sum_{k=1}^{K} \mathbf{G'}_k \mathbf{V}_{k-1} \mathbf{A}_k \mathbf{y}_1^{(K)} + \sum_{k=1}^{K} \mathbf{G'}_k \mathbf{V}_{k-1} \mathbf{b}_k. \tag{2.38}$$

L'idée qui sous tend la méthode FFP est de calculer $\mathbf{y}_1^{(K)}$ récursivement selon (2.36) puis de calculer $\mathbf{A}_k$ et $\mathbf{b}_k$ récursivement. La méthode FFP vise donc à palier les lacunes de BPTT en temps réel, tout en calculant le gradient exact, mais le nombre d'opérations passe de $O(N^2)$ à $O(N^3)$. Par ailleurs, les opérations ne sont plus locales.

2.7. L'approche par fonction de Green (GF)

L'approche par fonction de Green GF exploite le fait que $\mathbf{V}_k$ est une matrice fortement creuse. Afin de réduire la complexité, une relation récursive est exhibée sur le gradient d'erreur, $dE/d\mathbf{w}(K)$, obtenu avec sur une trajectoire de K exemples. Grâce aux Equations (2.18) et (2.15), on obtient

$$\frac{\partial^+ E}{\partial \mathbf{w}}(K) - \frac{\partial^+ E}{\partial \mathbf{w}}(K-1) =$$

$$\mathbf{e}_K^T \cdot \begin{pmatrix} \mathbf{W}\mathbf{G'}_2 \cdots \mathbf{W}\mathbf{G'}_K \\ \vdots \\ \mathbf{W}\mathbf{G'}_K \\ \mathbf{I} \end{pmatrix}^T \cdot \begin{pmatrix} \mathbf{G'}_1 \mathbf{V}_0 \\ \mathbf{G'}_2 \mathbf{V}_1 \\ \vdots \\ \mathbf{G'}_K \mathbf{V}_{K-1} \end{pmatrix}. \tag{2.39}$$

Posons

$$\mathbf{U}_K = \mathbf{W}^T \mathbf{G}'_{K-1} \dots \mathbf{G}'_2 \mathbf{W}^T \mathbf{G}'_1. \tag{2.40}$$

On observe que $\mathbf{U}_K$ s'obtient facilement à partir de $\mathbf{U}_{K-1}$

$$\mathbf{U}_K = \mathbf{W}^T \mathbf{G}'_{K-1} \mathbf{U}_{K-1}. \tag{2.41}$$

Posons

$$\mathbf{S}(K, j) = \mathbf{S}(K-1, j) + \mathbf{U}_{K-1}^{-1} \mathbf{v}_{K-1}(j), \tag{2.42}$$

on vérifie que

$$\frac{\partial^+ E}{\partial \mathbf{w}_{ij}}(K) = \frac{\partial^+ E}{\partial \mathbf{w}_{ij}}(K-1) + \mathbf{e}_k^T \mathbf{G'}_K \mathbf{U}_K \mathbf{S}_i(K, j) \tag{2.43}$$

où $S_i(K, j)$ est la i-ième colonne de la matrice $S(K, j)$ de taille $N \times N$. La méthode de Green nécessite une inversion matricielle ; les opérations ne sont pas locales. L'algorithme réduit la complexité de l'approche FP en considérant une récursion directement sur le gradient d'erreur. Le nombre d'opérations varie en $O(N^3)$. Sun, Chen et Lee [SUN 92] ont appliqué l'algorithme sur un problème de classification de trajectoires 2D, étudié à l'origine par Williams et Zipzer, pour montré sa rapidité supérieure à RTRL.

2.8. L'approche par blocs (BU)

L'approche par 'Bloc-Update' (BU) combine les avantages de BPTT et de FD. Supposons le gradient calculé à l'instant $K - M$ et posons

$$\mathbf{G}(K) = \frac{\partial^+ E}{\partial \mathbf{w}}(K) - \frac{\partial^+ E}{\partial \mathbf{w}}(K - M) \tag{2.44}$$

Par linéarité des équations, calculer $\mathbf{G}(K)$ revient à résoudre (2.15) avec $\partial E / \partial \mathbf{v}$ remplacé par $(0, \dots, 0, e^T_{K-M+1}, e^T_{K-M+2}, \dots, e^T_K)^T$. $\mathbf{G}(K)$ s'obtient en résolvant les équations

$$\begin{array}{lcll} \mathbf{y}_k & = & \mathbf{W}\mathbf{G}'_{k+1}\mathbf{y}_{k+1}, & k = 1, \dots, K-M, \\ \mathbf{y}_k & = & \mathbf{e}_k + \mathbf{W}\mathbf{G}'_{k+1}\mathbf{y}_{k+1}, & k = K-M+1, \dots, K-1. \end{array}$$

avec $\mathbf{y}_K = \mathbf{e}_K$. Posons

$$\mathbf{Z}(k_1, k_2) = \sum_{k=k_1}^{k_2} \mathbf{G}'_k \mathbf{y}_k \mathbf{v}_{k-1}^T. \tag{2.45}$$

$\mathbf{Z}(K-M+1, K)$ s'obtient facilement grâce à ces équations. Il reste à déterminer $\mathbf{Z}(1, K-M)$ en appliquant (2.45) pour $k = 1, \ldots, K-M$, ainsi on aura

$$\mathbf{G}(K) = -\mathbf{Z}(1, K-M) - \mathbf{Z}(K-M+1, K) \tag{2.46}$$

Par récursion,

$$\mathbf{y}_k = \mathbf{W}\mathbf{G}'_{k+1} \ldots \mathbf{W}\mathbf{G}'_{K-M+1}\mathbf{y}_{K-M+1}, \quad k = 1, \ldots, K-M. \tag{2.47}$$

Il vient par substitution

$$\mathbf{Z}(1, K-M) = \sum_{k=1}^{K-M} \mathbf{G}'_k \mathbf{W}\mathbf{G}'_{k+1} \ldots \mathbf{W}\mathbf{G}'_{K-M+1}\mathbf{y}_{K-M+1}\mathbf{v}_{k-1}^T. \tag{2.48}$$

Chaque colonne de $Z(1, K-M)$ peut se calculer récursivement. Posons

$$\mathbf{Q}_i(K-M) = \sum_{k=1}^{K-M} \mathbf{v}_{k-1}(i)\mathbf{G}'_k \mathbf{W}\mathbf{G}'_{k+1} \ldots \mathbf{W}\mathbf{G}'_{K-M+1}\mathbf{y}_{K-M+1}. \tag{2.49}$$

Alors $\mathbf{Z}_i(1, K-M) = \mathbf{Q}_i(K-M)\mathbf{y}_{K-M+1}$ où $\mathbf{Z}_i(1, K-M)$ désigne la i-ème ligne de $\mathbf{Z}(1, K-M)$. $\mathbf{Q}_i(K)$ se calcule récursivement en termes de $\mathbf{Q}_i(K-M)$. Posons

$$\mathbf{\Gamma}(k, K) = \mathbf{G}'_K \mathbf{W}^T \ldots \mathbf{G}'_{k+1} \mathbf{W}^T \mathbf{G}'_k. \tag{2.50}$$

Il vient

$$\begin{aligned} \mathbf{Q}_i(K) &= \mathbf{\Gamma}(K-M+1, K)\mathbf{W}\mathbf{Q}_i(K-M) \\ &+ \sum_{k=K-M+1}^{K} \mathbf{v}_{k-1}(i)\mathbf{\Gamma}(k-1, K). \end{aligned} \tag{2.51}$$

$\mathbf{\Gamma}(k,K)$ se calcule récursivement par

$$\mathbf{\Gamma}(k,K) = \mathbf{\Gamma}(k+1,K)\mathbf{W}\mathbf{G}'_k. \tag{2.52}$$

Ainsi, une fois $\mathbf{G}(K)$ obtenu, le gradient est évalué par

$$\frac{\partial^+ E}{\partial \mathbf{w}}(K) = \frac{\partial^+ E}{\partial \mathbf{w}}(K-M) + \mathbf{G}(K). \tag{2.53}$$

L'approche par blocs (BU) combine les avantages de BPTT et de FD. Les poids sont ajustés tous les $O(N)$ instants en $O(N^4)$ opérations. Il faut donc $O(N^3)$ opérations à chaque instant en moyenne.

2.9. Apprentissage des délais

Aucun élément précis en biologie n'accrédite l'hypothèse d'un apprentissage continu des délais hormis la période de croissance de l'organisme. Pour autant, l'ajustement de certains délais ciblés est évoqués dans [DAY 93, BAL 94, BAL 95, PEA 95] comme un moyen pour les réseaux de réguler leur propre dynamique. Baldi et Atiya [BAL 94] ont établi des conditions simples sur les poids et les délais pour imposer un comportement oscillatoire dans des architectures neuronales simples (e.g. un réseau en anneau). Dans [BOD 90], une variante de TDNN explore des synapses d'enveloppe gaussienne dont les centres et les écarts-types sont ajustés. Mais d'une manière générale, les algorithmes présentés ne fonctionnent pas pour ajuster globalement et sans discernement tous les délais mais vise plutôt l'ajustement sélectif de certains délais. L'apprentissage des délais peut se montrer instable en raison de bifurcations du système dynamique [BAL 94].

Il est plus commode de partir d'une neurodynamique *continue* pour assurer l'existence du gradient d'erreur par rapport aux délais, puis de revenir ultérieurement aux réseaux discrets par la méthode d'Euler. Les règles d'expansion des dérivées partielles dans les systèmes à temps continu s'appliquent sans difficulté pour conduire à des versions BPTT et FD. La transposition aux réseaux à délais à valeur discrète exige toutefois des approximations sévères.

Les réseaux continus - La neuro-dynamique du modèle continu bouclé à délais est régie typiquement par les équations

$$h_j(t) := \tau_j \frac{dv_j}{dt} = -v_j + g(\sum_{i=1}^{N} w_{ij} v_i(t-\tau_{ij})) + i_j, \tag{2.54}$$

où les termes τ_i et les délais τ_{ij} sont des réels positifs, v_j désigne l'activation du neurone j à l'instant t et $[t_0, t_1]$ est la durée de l'époque. Les délais τ_{ij} sont ajustés de façon à faire évoluer le réseau, sous l'action des entrées $\mathbf{i}(t)$, vers une trajectoire désirée $\mathbf{v}^\star(t)$ sur l'intervalle de temps $[t_0, t_1]$. La fonction erreur est cette fois une fonctionnelle de la forme

$$E = \int_{t_0}^{t_1} e(\mathbf{v}^\star(t), \mathbf{v}(t), t)dt. \tag{2.55}$$

L'erreur des moindres carrés s'écrit

$$E = \frac{1}{2}\int_{t_0}^{t_1} (\mathbf{v}^\star(t) - \mathbf{v}(t))^2 dt. \tag{2.56}$$

Elle est calculée sur l'ensemble des neurones visibles, à des intervalles de temps où $\mathbf{v}^\star(t)$ est disponible. La descente du gradient en temps continu appliqué au délai τ s'écrit

$$\frac{d\tau}{dt} = -\eta \frac{\partial^+ E}{\partial \tau}, \tag{2.57}$$

où η, le pas d'apprentissage, est une constante positive choisie convenablement. Minimiser E sous les contraintes neurodynamiques est un problème d'optimisation en *dimension infinie*. Une analyse par le calcul variationnel est menée dans ([PEA 95]). Une autre façon de procéder s'appuie sur les dérivées partielles. Explicitons la dérivée de l'erreur par rapport à τ,

$$\frac{\partial^+ E}{\partial \tau} = \int_{t_0}^{t_1} \sum_{p=1}^{N} \frac{\partial e}{\partial v_p}\frac{\partial^+ v_p}{\partial \tau} dt = \int_{t_0}^{t_1} \frac{\partial e}{\partial \mathbf{v}} \cdot \frac{\partial^+ \mathbf{v}}{\partial \tau} dt, \tag{2.58}$$

avec $\partial e/\partial \mathbf{v}_j = 0$ pour des neurones cachés. Posons

$$\mathbf{z}^\tau(t) := \frac{\partial^+ \mathbf{v}(t)}{\partial \tau}. \tag{2.59}$$

Le vecteur $\mathbf{z}^\tau(t)$ représente la sensibilité des v_j à une variation d'un délai τ. Son calcul s'obtient par

$$\mathbf{z}^\tau(t) = \frac{d}{d\tau}\int_{t_0}^{t} \frac{d\mathbf{v}}{d\alpha} d\alpha = \int_{t_0}^{t} \frac{d\mathbf{h}}{d\tau} dt \tag{2.60}$$

et en dérivant,

$$\frac{d\mathbf{z}^{\tau}(t)}{dt} = \frac{\partial^{+}\mathbf{h}}{\partial\tau} = \frac{\partial\mathbf{h}}{\partial\mathbf{v}} \cdot \frac{\partial^{+}\mathbf{v}}{\partial\tau} + \frac{\partial\mathbf{h}}{\partial\tau} = \frac{\partial\mathbf{h}}{\partial\mathbf{v}} \cdot \mathbf{z}^{\tau}(t) + \frac{\partial\mathbf{h}}{\partial\tau}, \tag{2.61}$$

où la matrice Jacobienne $\partial\mathbf{h}/\partial\mathbf{v}$ dépend du temps.

Approche FD - L'intégration numérique de (2.61) revient à discrétiser $[t_0, t_1]$ en K intervalle de temps Δ,

$$\mathbf{z}^{\tau}(k+1) = \mathbf{z}^{\tau}(k) + \left[\frac{\partial\mathbf{h}}{\partial\mathbf{v}}(k) \cdot \mathbf{z}^{\tau}(k) + \frac{\partial\mathbf{h}}{\partial\tau}(k)\right] \Delta t. \tag{2.62}$$

Or,

$$\frac{\partial h_j}{\partial v_i} = -\delta_{ij} + \cdot w_{ij} \cdot g'(\sum_{p=1}^{N} w_{pj} v_p(t - \tau_{pj})), \tag{2.63}$$

on obtient, en remplaçant, une expression explicite pour le calcul des z^{τ},

$$z_q^{\tau_{ij}}(k+1) = w_{iq} g'(\sum_{p=1}^{N} w_{pq} v_p(k - \tau_{pq})) z_p^{\tau_{ij}}(t - \tau_{pq}) + \frac{\partial h_q}{\partial \tau_{ij}}, \tag{2.64}$$

avec

$$\begin{aligned} \frac{\partial h_q}{\partial \tau_{ij}} &= -\delta_{jq} \cdot w_{iq} \cdot g'(\sum_{p=1}^{N} w_{pq} v_p(t - \tau_{pq})) \cdot \frac{dv_q(t - \tau_{iq})}{dt} \\ &= -\delta_{jq} w_{iq} g'(\sum_{p=1}^{N} w_{pq} v_p(t - \tau_{pq})) \cdot h_q(t - \tau_{iq}), \end{aligned} \tag{2.65}$$

et

Les délais sont ajustés selon

$$\frac{\partial^{+} E}{\partial \tau} \simeq \sum_{k=1}^{K} \mathbf{e}(k)^T \mathbf{z}^{\tau}(k). \tag{2.66}$$

On retrouve la version FD (2.27) pour les délais. Mais l'intégration numérique est très coûteuse puisque il y a autant de variables, $\mathbf{z}^{\tau}$, à propager que de délais à ajuster. On lui préfère la version BPTT suivante.

Approche BPTT - La méthode de l'adjoint est courante dans les résolution des systèmes d'équations différentielles *linéaires* [BAL 95]. La résolution de (2.61) s'obtient grâce à la résolution d'un son système linéaire auxiliaire défini par

$$\frac{d\mathbf{y}}{dt} = -\left(\frac{\partial \mathbf{h}}{\partial \mathbf{v}}\right)\mathbf{y}^T - \frac{\partial e}{\partial \mathbf{v}}, \tag{2.67}$$

que l'on nomme le **système adjoint** de (2.61). D'après (2.61) et (2.70), les variables vérifient la relation

$$\frac{d(\mathbf{y}^T\mathbf{z}^\tau)}{dt} = \mathbf{y}^T\frac{\partial \mathbf{h}}{\partial \tau} - \left(\frac{\partial e}{\partial \mathbf{v}}\right)^T\mathbf{z}^\tau. \tag{2.68}$$

D"après (2.58), on obtient

$$\begin{aligned}\frac{\partial^+ E}{\partial \tau} &= \int_{t_0}^{t_1}\frac{\partial e}{\partial \mathbf{v}}\cdot\mathbf{z}^\tau dt \\ &= (\mathbf{y}^T\mathbf{z}^\tau)_{t=t_0} - (\mathbf{y}^T\mathbf{z}^\tau)_{t=t_1} + \int_{t_0}^{t_1}\mathbf{y}^T\frac{\partial \mathbf{h}}{\partial \tau}dt \\ &= \int_{t_0}^{t_1}\mathbf{y}^T\frac{\partial \mathbf{h}}{\partial \tau}dt,\end{aligned} \tag{2.69}$$

d'après les conditions aux limites $\mathbf{y}(t_1) = 0$ et $\mathbf{z}^\tau(t_0) = 0$. Remarquons que $\mathbf{y}(t)$ ne dépend pas de τ à la différence des $\mathbf{z}^\tau(t)$. L'intégration numérique de (2.67) revient à discrétiser $[t_0, t_1]$ en K intervalle de temps Δt, choisis ici égaux à l'unité de temps, de sorte que

$$y_i^\tau(k) = e^\tau(k) + \sum_{j=1}^{N} w_{ij}g'(s_j(k+\tau_{ij}))y_j^\tau(k+\tau_{ij}). \tag{2.70}$$

Le délais sont ajustés selon

$$\frac{\partial^+ E}{\partial \tau_{ij}} = \sum_{k=1}^{K} w_{ij}g'(s_j(k))h_j(k-\tau_{ij})y_j(k). \tag{2.71}$$

On retrouve la version BPTT (2.32) pour les délais. Toutefois, l'inconvénient majeur provient de ce qu'il faut mémoriser le passé de la trajectoire sur $[t, t - \max_{ij}\tau_{ij}]$.

Les réseaux discrets - La transposition de ces algorithmes aux réseaux discrets à délais à valeur continue ne peut être envisagée qu'au prix de quelques approximations grossières. Par exemple, Duro et Santo Reyes [DUR 99] choisissent d'interpoler linéairement l'activation des neurones entre deux instant $k+[\tau]$ et $k+[\tau]+1$ pour obtenir la valeur à $k+\tau$, $[\tau]$ étant la partie entière de τ. Supposons une lente évolution des valeurs activations entre deux instants consécutifs, on peut écrire

$$\frac{\partial v_k}{\partial \tau_{ij}} \simeq \delta_{jk} w_{ik} g'(\sum_{p=1}^{N} w_{pk} v_p(t-\tau_{pk}))\,[v_p(t-\tau_{pk}) - v_k(t-1-\tau_{pk})] \qquad (2.72)$$

La solution consistant à arrondir à l'entier le plus proche les délais conduit à de brusques sauts du gradient. C'est pourquoi il est préférable d'interpoler l'activation des neurones entre deux instant $k+[\tau]$ et $k+[\tau]+1$ pour obtenir la valeur à $k+\tau$, $[\tau]$ étant la partie entière de τ. L'usage de délais continus dans un réseau discret est ainsi autorisé. La version BPTT (2.32) pour les délais devient

$$\frac{\partial^+ E}{\partial \tau_{ij}} \simeq \sum_{k=1}^{K} w_{ij} \cdot g'(s_j(k)) \cdot (v_j(k-\tau_{ij} - v_j(k-\tau_{ij}-1)) \cdot y_j(k). \qquad (2.73)$$

Duro et Santo Reyes ont jaugé les performances de l'algorithme sur la prédiction à un pas de la série Mackey-Glass et sur la classification d'électroencéphalogrammes avec des réseaux à *un seul* neurone d'entrée.

2.10. Synthèse et conclusion

Ce chapitre a présenté une nouvelle formulation pour unifier les cinq algorithmiques majeurs parues dans la littérature ces dernières années pour le calcul exact du gradient d'erreur, parmi lesquels

- la propagation avant 'forward propagation algorithm' (FP) ou 'real-time recurrent learning' (RTRL) [WIL 89]) est un algorithme en ligne (*on line*). Les paramètres (poids et délais) à l'instant k sont adaptés de manière récursive, en fonction de ceux à l'instant $k-1$. La complexité temporelle en $O(N^4)$ le rend toutefois extrêmement lourd à mettre en oeuvre.
- la rétro-propagation dans le temps (back-propagation through time' (BPTT) [RUM 86, WER 90]). BPTT opère *hors ligne* : le réseau bouclé est transposé en réseau non-bouclé déplié dans le temps. Sa version tronquée très

efficace de complexité temporelle en $O(N^2)$ est d'usage courant. Ce n'est pas une méthode exacte ; la question de la troncature de BPTT fera l'objet du Chapitre 4. Nous verrons qu'il est possible de limiter le nombre de dépliements du réseau adjoint, tout en garantissant une borne sur l'erreur d'approximation du gradient d'erreur, tout en préservant une complexité attractive en $O(N^2 \ln(N))$.

- le 'fast forward propagation' (FFP) [TOO 92] vise à palier les lacunes de BPTT dans un fonctionnement en ligne tout en demeurant une méthode exacte. Les paramètres à l'instant sont adaptés de manière récursive. L'idée est de calculer les condition aux limites à l'instant $k = 1$ de façon récursive, plutôt que par propagation arrière, au prix d'une complexité temporelle en $O(N^3)$.
- l'approche par fonction de Green (GF) [SUN 92], améliore la complexité de FP en exhibant une relation récursive du gradient d'erreur par le biais d'une fonction de Green. La méthode fonctionne en ligne.
- l'approche par 'Bloc-Update' (BU) [SCH 92] combine BPTT et FD entre $K - N$ et K. L'ajustement des poids en $O(N^4)$ s'applique tous les N instants, la complexité est donc en $O(N^3)$.

Ce cadre unificateur pour le calcul exact du gradient d'erreur dans les réseaux discrets s'inscrit dans la lignée de travaux similaires [ATI 00, AUS 95b, BAL 95, PIC 94, PEA 95], certains fondés sur les systèmes linéaires adjoints ou le calcul variationnel, dont les vertus sont essentiellement pédagogiques. Ils ont vocation a montré que, en dépit des innombrables déclinaisons algorithmiques qui ont vues le jour, toutes peuvent se reformuler dans un cadre commun (i.e. , Eq. 2.15) qui met à jour les différences algorithmiques en termes de complexité, de mémoire, de localité des opérations.

Toute ces taxonomies montrent, au demeurant, que la version tronquée de BPTT en $O(N^2)$ a été largement adoptée dans la communauté connexioniste en vertu de la localité des opérations et de sa faible complexité en temps et en espace. Nous en reparlerons en détail au Chapitre 4.

Chapitre 3

APPRENTISSAGE DES RESEAUX BOUCLES A DELAIS

3.1. Introduction

Les versions *forward* (FP) et *backward* (BPTT) de calcul du gradient sont établies comme au chapitre précédent pour une classe plus générale de d'architectures à délais pour l'**apprentissage de points fixes** et l'**apprentissage de trajectoires** : les réseaux FIR bouclés. Ce sont des réseaux discrets bouclés à délais dont les synapses sont représentées par des **filtres linéaires à réponse impulsionnelle finie**[1] (FIR) : des connexions arbitrairement retardées et bouclées sont autorisées entre les neurones. Cette architecture générale porte le nom de "Dynamical Recurrent Neural Networks" (DRNN) [AUS 95b, AUS 02b]. Cette modification somme toute élémentaire qui reflête l'*intégration spatiale et temporelle* des signaux incidents au sein de la dentrite et du corps cellulaire, permet aux interactions entre neurones de représenter un vaste choix de mémoires formelles, variant continuement d'une mémoire à faible *résolution* mais de longue *étendue*, à une mémoire à forte *résolution* mais d'*étendue* limitée [VRI 92, WAN 93].

Les DRNN fédèrent donc un grand nombre d'architectures localement et globalement récurrentes proposées dans la littérature pour le traitement temporel (voir par exemple [KRE 01, PIC 94, TSO 94, BAL 95, CAM 99, DUR 99, WAN 93, WIL 89] ainsi que les réseaux bouclés à point fixe [ALM 87, PIN 87]. Les réseaux bouclés à points fixes sont, d'ordinaire, peu utilisés en prévision pour éviter d'avoir à relaxer le réseau vers un hypothétique point fixe dont d'existence et l'unicité demeurent d'ordinaire hypothétiques. On leur préfère un

[1] Filtre FIR : à une impulsion de durée finie, correspond une réponse de durée finie.

rôle de mémoire associative ou de satisfaction de contraintes [PEA 95]. L'absence de cycle de retard nul est parfois imposée dans la définition même des réseaux de neurones bouclés [DRE 02]. Or rien théoriquement ne s'y oppose dès lors que la stabilité asymptotique des points fixes est établie. Le bouclage est réputé contribuer à la robustesse du réseau au bruit d'entrée [PEA 95]. Rappelons que l'identification des attracteurs aux objets à mémoriser est un principe fondateur du paradigme neuronal [HAY 94]. D'ordinaire, la localisation des attracteurs est fixée au préalable à l'information que l'on désire coder [PIN 87] pour réaliser une mémoire associative [HOP 82]. Elle peut également être continûement ajustée pour décrire une trajectoire désirée [TOO 91, PEA 95, COH 97] comme c'est le cas pour le DRNN. Le vecteur d'état du DRNN est obtenu à l'issue d'une phase de relaxation vers un état d'équilibre

$$\mathbf{v}_k = g(\sum_{d=0}^{D} \mathbf{W}^{d\,T} \mathbf{v}_{k-d}) + \mathbf{i}_k. \tag{3.1}$$

dont nous discutons l'existence, l'unicité et la stabilité asymptotique à chaque instant k dans ce chapitre. Une idée classique pour rechercher un point fixe des équations (3.1), à chaque instant k, est d'appliquer la méthode des approximations successives [KHA 96, MAN 01]

$$\mathbf{x}(t+1) = g(\mathbf{W}^{0\,T} \mathbf{x}(t) + \mathbf{c}_k), \quad t = 1, 2, \ldots \tag{3.2}$$

où le vecteur $\lim_{t\to\infty} \mathbf{x}(t)$, lorsqu'il existe, est le vecteur $\mathbf{v}_k$ que l'on cherche ; $\mathbf{c}_k$ est le vecteur constant $\sum_{d=1}^{D} \mathbf{W}_k^{d\,T} \mathbf{v}_{k-d} + \mathbf{i}_k$ car indépendant de t.

Nous établissons des conditions suffisantes garantissant l'existence, l'unicité et la stabilité asymptotique du point fixe ainsi que la convergence asymptotique vers un point fixe du réseau en bouclage fermé. Les réseaux à point fixes sont réputés être plus robustes au bruit des entrées. Pour autant, la relaxation est gourmande en temps de calcul ; l'existence d'un point d'équilibre *stable* n'est pas même garantie en toute situation. Dans la pratique, **on peut s'affranchir de la relaxation** et donc des questions afférentes à l'existence des points fixes en prenant une matrice de poids **sans cycles de retard nul**, ou plus simplement en prenant une matrice $\mathbf{W}^0$ triangulaire ou nulle. Mais dans tous les cas, la stabilité asymptotique du modèle en mode bouclage fermé pour la simulation reste à établir.

$\mathbf{v}$	vecteur d'activation
$\mathbf{u}$	vecteur des entrées internes
$\mathbf{i}$	vecteur des entrées externes
w_{ij}^d	poids de la connexion retardée de d entre le neurone i et le neurone j
$\mathbf{w}^d(j) = [w_{1j}^d, \ldots, w_{Nj}^d]^T$	vecteur de poids de délai d vers le neurone j
$\mathbf{W}^d = [\mathbf{w}^d(1), \mathbf{w}^d(2), \ldots, \mathbf{w}^d(N)]$	matrice de poids de délai $d \leq D$
$g()$	fonction d'activation
μ	$\max_u(g'(u))$
$\mathbf{G}'$	$N \times N$ matrice diagonale de $g'(u(j))$

Tableau 3.1. *Notations du DRNN. Par simplicité, l'index de l'itération* k *est omis.*

3.2. Les modèles inclus dans le formalisme DRNN

De nombreux réseaux bouclés en temps discret sont inclus dans le formalisme DRNN [MAN 01]. Lorsque $\mathbf{W}_k^0 = 0$ et $\mathbf{W}_k^{d\,T}$ sont triangle supérieures pour chaque $d \neq 0$, on obtient le réseau FIR non bouclé de Eric Wan [WAN 93]. Beaucoup d'architectures dites "locally recurrent globally feedforward" sont des cas particuliers du DRNN (e.g., generalized Frasconi-Gori-Soda architecture [FRA 92]), l'architecture de Poddar-Unnikrishnan [TSO 94] que celle étudiée par Duro et Santos Reyes [DUR 99] et Cohen, Saad et Maromet [COH 97], sans oublier les réseaux d'Elman [ELM 90] et de Jordan [JOR 92]. Les réseaux bouclés (asymétriques) à point fixe étudiés par Pineda [PIN 87] sont également inclus dans le ce formalisme en posant $\mathbf{W}_k^d = 0$ pour $d > 0$.

Lorsque $\forall d \neq 1$, $\mathbf{W}_k^d = 0$, Le DRNN est réduit au réseau globalement bouclé standard [WIL 89, LIN 96]. Les modèles Non-linéaires Auto-Régressifs (à moyenne ajustée) avec entrées eXogène (NARMAX) [LIN 96] sont aussi représentés.

Soulignons enfin que mêmes si les architectures à base de synapses IIR[2], e.g. Tsoi et Back IIR [TSO 94], n'entrent pas dans ce formalisme, elles peuvent se reformuler sous la forme d'un DRNN avec des neurones supplémentaires à fonction d'activation linéaire et des connexions FIR localement bouclées localement. Ainsi, les conditions sur la matrice de poids pour assurer la stabilité de l'apprentissage par descente de gradient s'appliqueront également aux réseaux IIR, à condition qu'ils soient reformulés en DRNN. En tout état de cause,

[2]Lorsque le bouclage s'effectue avant la non-linéarité.

toutes ces architectures souffrent d'un comportement dit "oublieux" (*forgetting behavior*), au sens où il est difficile de mémoriser des événements survenus loin dans le passé.

3.3. Existence, unicité et stabilité du point fixe

Les paramètres du système sont déterminés, à chaque étape k, en relaxant le système vers un état d'équilibre caractérisé par (3.1). La convergence vers un point d'équilibre stable n'est pas garantie ; un comportement oscillatoire [BAL 94] ou une dynamique chaotique ne sont pas exclus [HER 91]. La **présence de cycles de retard nul** dans la formulation DRNN requiert une phase de relaxation pour atteindre un état d'équilibre stable. Or, le modèle continu dont le DRNN est l'approximation d'Euler, diffère de la neurodynamique du théorème de Cohen-Grossberg qui caractérise la stabilité de nombreux réseaux bouclés (à connexions symétriques), en particulier la modèle de Hopfield. L'existence, l'unicité et la stabilité asymptotique du point fixe dans un DRNN de connectivité et des délais quelconques sont établies dans ce qui suit.

Existence et unicité - Les systèmes *globalement asymptotiquement stables* [KHA 96] convergent presque sûrement vers un point d'équilibre. Les RN bouclés pour lesquels une fonction de Lyapounov a pu être exhibée, tels que le modèle de Hopfield ou le Brain-State-In-The-Box (BSB) [GOL 96] (dont la neurodynamique entre dans le cadre du théorème de Cohen-Grossberg [GRO 88]) sont globalement asymptotiquement stables. Ce sont typiquement des réseaux à matrice de poids symétrique. Les réseaux bouclés à matrice de poids non symétrique n'admettent de point fixe que sous certaines conditions comme nous allons le voir.

Le point fixe des équations (3.1) est obtenu, à chaque instant k, par la méthode des approximations successives [KHA 96]

$$\mathbf{x}(t+1) = g({\mathbf{W}^0}^T \mathbf{x}(t) + \mathbf{c}_k), \quad t = 1, 2, \ldots \tag{3.3}$$

où le vecteur $\lim_{t\to\infty} \mathbf{x}(t)$, lorsqu'il existe, est le vecteur $\mathbf{v}_k$ que l'on cherche ; $\mathbf{c}_k$ est le vecteur constant $\sum_{d=1}^{D} {\mathbf{W}_k^d}^T \mathbf{v}_{k-d} + \mathbf{i}_k$ car indépendant de t. La convergence de (3.2) est assurée par application directe du théorème du point fixe (dit de Brouwer ou des transformations contractantes [JIN 94, KHA 96, MAN 01]),

Théorème 1 *Pour toute matrice* $\mathbf{W}^0$, **si** *l'une de ces conditions est vérifiée,*

$$\mu \max_i \left(\sum_j |w_{ij}^0| \right) < 1, \quad \mu \max_j \left(\sum_i |w_{ij}^0| \right) < 1, \quad \mu \left[\sum_{i,j} |w_{ij}^0|^2 \right]^{1/2} < 1, \tag{3.4}$$

alors *le système dynamique (3.2) admet, pour tout vecteur constant,* $\mathbf{c}$, *un unique point d'équilibre asymptotiquement stable* $\mathbf{x}^\star$ *vérifiant* $\mathbf{x}^\star = g({\mathbf{W}^0}^T \mathbf{x}^\star + \mathbf{c})$. $\mathbf{x}^\star$ *est obtenu par la méthode des approximations successives, dont la convergence est asymptotiquement linéaire, en partant d'un point initial* $\mathbf{x}_O$ *quelconque.*

Preuve : l'existence s'établit immédiatement [JIN 94] en remarquant que $g(\mathbf{x})$, fonction continue de $[-1,+1]^N$ dans $[-1,+1]^N$, admet au moins un point fixe par le théorème du point fixe de Brouwer ($\mathbf{x}(t)$ est une suite de Cauchy dans un espace de Banach, donc convergente). L'unicité découle de ce que la fonction génératrice du système dynamique (3.2) soit Lipschitzienne sur la boule unité. En effet la matrice jacobienne du système (3.2) vérifie $\|\mathbf{G}'\mathbf{W}^T\| \leq \mu \|\mathbf{W}^T\| < 1$ d'après les conditions (3.4) obtenues en prenant des p-normes matricielles de $\mathbf{W}$ avec $p = 1, 2, \infty$. Donc si $\mathbf{x}_1^\star$ et $\mathbf{x}_2^\star$ sont deux points fixes,

$$\|\mathbf{x}_1^\star - \mathbf{x}_2^\star\| \leq \|\mathbf{G}'\mathbf{W}^T\| \|\mathbf{x}_1^\star - \mathbf{x}_2^\star\| < \|\mathbf{x}_1^\star - \mathbf{x}_2^\star\|, \tag{3.5}$$

donc $\mathbf{x}_1^\star = \mathbf{x}_2^\star$. On retrouve la condition d'unicité citée dans [PEA 95]. De même, la convergence asymptotique est linéaire car

$$\frac{\|\mathbf{x}_{k+1} - x^\star\|}{\|\mathbf{x}_k - x^\star\|} \leq \|\mathbf{G}'\mathbf{W}^T\|, \quad \forall k. \tag{3.6}$$

La stabilité asymtptotique résulte d'une application de la méthode indirecte de Lyapounov, i.e., le théorème de linéarisation des systèmes autonomes $dy/dt = f(y)$ (p. 127, [KHA 96]) avec $y = x - x^\star$. Le point d'équilibre $y = 0$ est asymptotiquement stable car toutes les valeurs propres de la matrice jacobienne $\mathbf{G}'\mathbf{W}^T$ du système (3.2) satisfont $\forall i, \mathrm{Re}\lambda_i < 0$ puisque $\|\mathbf{W}\| < 1/\mu$. Il en va de même pour $x^\star$. Les conditions (3.4) sont très similaires à celles énoncées dans [PIN 87, PEA 95].

□

Ce résultat ne dit rien, en revanche, sur la taille du bassin d'attraction du point fixe. Des études empiriques sur des réseaux initialisés avec des poids

aléatoires ont montré que des conditions beaucoup moins contraignantes sur les poids suffisent à assurer l'existence et la stabilité asymptotique des points fixes [REN 90, PEA 95].

Par ailleurs, même une fois garantie l'existence d'un point fixe, l'apprentissage peut rencontrer quelques problèmes. Le gradient de l'erreur sur les paramètres initiaux du réseau n'est pas forcément continu, même si la dynamique du réseau l'est. La composée successive de fonctions continues ne tend pas nécessairement vers une fonction continue. Un changement infinitésimal des poids à l'instant initial peut modifier de façon drastique la localisation du point d'équilibre dans lequel va s'établir le réseau [PEA 95], auquel cas le gradient peut ne pas être défini.

Stabilité des trajectoires en bouclage fermé - Lorsque le réseau fonctionne en bouclage fermé pour la simulation, un signal d'entrée constant est appliqué, $\mathbf{i}_k = \mathbf{i}$, et le modèle est itéré sur lui-même. La trajectoire peut converger (ou non) vers un point fixe, un cycle limite ou encore un attracteur chaotique comme nous le verrons dans le chapitre consacré à la reconstruction de certains attracteurs chaotiques. Dans la suite, des conditions suffisantes d'existence de points fixes et de leur stabilité asymptotique sont établies.

La mise jour des activations des neurones opère en mode *asynchrone* séquentiel ou aléatoire, les deux sont équivalents ici. L'important est qu'un seul neurone soit ajusté à chaque instant. Considérons l'équation récurrente des entrées $s_k(j) = g^{-1}(v_k(j))$,

$$s_k(j) = \sum_{d=0}^{D} \sum_{i=1}^{N} w_{ij}^d g(s_{k-d}(i)) + i(j). \tag{3.7}$$

pour $j = 1, \ldots, N$. Supposons l'existence de $\mathbf{s}^\star$, un point d'équilibre de (3.7) et posons $\boldsymbol{\alpha}_k = \mathbf{s}_k - \mathbf{s}_k^\star$, il vient

$$\alpha_k(j) = \sum_{d=0}^{D} \sum_{i=1}^{N} w_{ij}^d \left(g(s_{k-d}^\star(i) + \alpha_{k-d}(i)) - g(s_{k-d}^\star(i)) \right). \tag{3.8}$$

Il est clair que $(0, 0, \ldots, 0)^T$ est un point d'équilibre de Eq. (3.8).

Montrons le résultat suivant [AUS 02a], dont la demonstration est inspirée d'un résultat récent de C.Feng et R. Plamondon [FEN 01] pour les neurodynamiques continues (du type étudié par Cao et Zhou 98 [CAO 98]) :

Théorème 2 *Pour toutes conditions initiales, le DRNN opérant en bouclage fermé admet un point d'équilibre globalement stable asymptotiquement si*

$$\mu \sum_{d=0}^{D} \sum_{i=1}^{N} |w_{ij}^{d}| < 1, \quad \forall j = 1, \ldots, N. \tag{3.9}$$

Preuve : l'existence du point fixe s'établit comme précédemment par le théorème du point fixe de Brouwer appliqué au système (3.1) ; $g()$ étant bijective, $\mathbf{s}^{\star}$ s'obtient par $g^{-1}(v_k^{\star}(j))$. Montrer la globale stabilité asymptotique de Eq. (3.7) revient à établir celle de (3.8). Considérons la fonction "énergie" définie à l'instant $k \geq D$ par

$$V(\boldsymbol{\alpha}_k^D) = \sum_{j=1}^{N} \sum_{d=0}^{D} |\alpha_{k-d}(j)| + \mu \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{d=0}^{D} \sum_{n=k-d+1}^{k} |w_{ij}^{d}||\alpha_n(j)|. \tag{3.10}$$

où le vecteur $\boldsymbol{\alpha}_k^D$ de taille $(D+1)N$ est la concaténation des vecteurs $\boldsymbol{\alpha}_{k-D}, \boldsymbol{\alpha}_{k-D+1}, \ldots, \boldsymbol{\alpha}_k$. Pour déterminer la variation $V(\boldsymbol{\alpha}_{k+1}^D) - V(\boldsymbol{\alpha}_k^D)$, on observe que le système (3.7) est itéré selon une dynamique asynchrone *séquentielle* sans perte de généralité : tous les neurones sont supposés être mis à jour dans l'ordre des indices. Par définition de μ, $g(s+\alpha) - g(s) \leq \mu\alpha$ pour tout s, α. Posons

$$\Delta V_k := V(\boldsymbol{\alpha}_{k+1}^D) - V(\boldsymbol{\alpha}_k^D) \tag{3.11}$$

Il vient

$$\begin{aligned}
\Delta V_k &= \sum_{j=1}^{N}(|\alpha_{k+1}(j)| - |\alpha_{k-D}(j)|) \\
&\quad +\mu\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{d=0}^{D}\sum_{n=k-d+1}^{k}|w_{ij}^d|\left(|\alpha_{n+1}(j)| - |\alpha_n(j)|\right), \\
&\leq \sum_{d=0}^{D}\sum_{i=1}^{N}\sum_{j=1}^{N}|w_{ij}^d||g(s^\star_{k+1-d}(i) + \alpha_{k+1-d}(i)) - g(s^\star_{k+1-d}(i))| \\
&\quad -\sum_{j=1}^{N}|\alpha_{k-D}(j)| + \mu\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{d=0}^{D}|w_{ij}^d|\left(|\alpha_{k+1}(j)| - |\alpha_{k+1-d}(j)|\right), \\
&\leq \mu\sum_{d=0}^{D}\sum_{i=1}^{N}\sum_{j=1}^{N}|w_{ij}^d||\alpha_{k+1}(i)| - \sum_{j=1}^{N}|\alpha_{k-D}(j)|. \qquad (3.12)
\end{aligned}$$

Et pour tout entier K,

$$\begin{aligned}
\sum_{k=D}^{K+D-1}\Delta V_k &= V(\boldsymbol{\alpha}_{K+D}^D) - V(\boldsymbol{\alpha}_D^D) \\
&\leq -\sum_{k=1}^{K}\sum_{j=1}^{N}\left[1 - \mu\sum_{d=0}^{D}\sum_{i=1}^{N}|w_{ij}^d|\right]|\alpha_{k+D}(j)|, \\
&\quad +\mu\sum_{k=1}^{D+1}\sum_{d=0}^{D}\sum_{i=1}^{N}\sum_{j=1}^{N}|w_{ij}^d||\alpha_{k+K}(i)| \\
&\quad -\sum_{k=0}^{D}\sum_{j=1}^{N}|\alpha_k(j)|. \qquad (3.13)
\end{aligned}$$

D'après (3.8), on a pour tout k, $|\alpha_k(j)| < 2\mu\sum_{d=0}^{D}\sum_{i=1}^{N}|w_{ij}^d| < 2$. Posons

$$r := \min_{1\leq j\leq N}\left[1 - \mu\sum_{i=1}^{N}\sum_{d=0}^{D}|w_{ij}^d|\right] > 0. \qquad (3.14)$$

D'après (3.13), il vient

$$\lim_{K\to\infty}\left\{V(\boldsymbol{\alpha}_{K+D}^D) + r\sum_{k=1}^{K}\sum_{j=1}^{N}|\boldsymbol{\alpha}_{k+D}(j)|\right\} < \infty. \qquad (3.15)$$

d'où

$$\sum_{k=D}^{\infty} \sum_{j=1}^{N} |\alpha_k(j)| < \infty. \tag{3.16}$$

Il s'ensuit que

$$\lim_{k\to\infty} \sum_{j=1}^{N} |\alpha_k(j)| = 0. \tag{3.17}$$

Ainsi, $\lim_{k\to\infty} \|\alpha_k\| = 0$. $(0, 0, \ldots, 0)^T$ est un point d'équilibre globalement asymptotiquement stable du système (3.8), donc $\mathbf{s}^\star$ est un point d'équilibre globalement asymptotiquement stable du système (3.7).

□

Cette démonstration est inspirée de l'analyse des réseaux continus à délais menée dans l'article récent de Feng et Palmondon [FEN 01]. Ce résultat généralise la formule de Mandic et Chambers pour le Perceptron récurrent NARMA (page 129, [MAN 01])

3.4. Calcul du gradient

Les questions de stabilité et de convergence asymptotique étant traitées, examinons maintenant les modifications qu'apporte la relaxation induite par la récurrence et les délais arbitraires, sur le calcul du gradient. La formulation du problème de minimisation sous contraintes égalité prend la forme

$$\begin{cases} \text{Minimiser } E \\ \text{sous les contraintes } \mathbf{h}_k = g(\sum_{d=0}^{D} \mathbf{W}^{d\,T} \mathbf{v}_{k-d}) - \mathbf{v}_k = \mathbf{0}, \quad k = 0, \ldots, K. \end{cases} \tag{3.18}$$

Les contraintes $\mathbf{h}_j$ sont exprimées comme des équations à point fixe dont les variables sont les $\mathbf{v}_k$. Arrangeons les colonnes $\mathbf{w}^d(i)$ de $\mathbf{W}^d$ en un long vecteur colonne, ainsi que $\mathbf{v}$ et le vecteur de contraintes $\mathbf{h}$

$$\mathbf{w}^d = \begin{pmatrix} \mathbf{w}^d(1) \\ \vdots \\ \mathbf{w}^d(N) \end{pmatrix}. \tag{3.19}$$

On a toujours l'équation de base (2.15),

$$\frac{\partial^+ E}{\partial \mathbf{w}} = -\frac{\partial E}{\partial \mathbf{v}} \left(\frac{\partial \mathbf{h}}{\partial \mathbf{v}}\right)^{-1} \frac{\partial \mathbf{h}}{\partial \mathbf{w}}. \tag{3.20}$$

mais cette fois, $\frac{\partial \mathbf{h}}{\partial \mathbf{v}}$ est une matrice plus dense,

$$\frac{\partial \mathbf{h}}{\partial \mathbf{v}} + \mathbf{I} = \begin{pmatrix} \mathbf{G}'_1 \mathbf{W^0}^T & 0 & 0 & \cdots & 0 \\ \mathbf{G}'_2 \mathbf{W^1}^T & \mathbf{G}'_2 \mathbf{W^0}^T & 0 & \cdots & 0 \\ \mathbf{G}'_3 \mathbf{W^2}^T & \mathbf{G}'_3 \mathbf{W^1}^T & \mathbf{G}'_3 \mathbf{W^0}^T & \cdots & \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \mathbf{G}'_{D+1} \mathbf{W^D}^T & \ddots & \ddots & & \\ 0 & \mathbf{G}'_{D+2} \mathbf{W^D}^T & \ddots & \ddots & \vdots \\ \vdots & & \ddots & & \\ 0 & 0 & \mathbf{G}'_K \mathbf{W^D}^T & \cdots \quad \mathbf{G}'_K \mathbf{W^1}^T & \mathbf{G}'_K \mathbf{W^0}^T \end{pmatrix} \tag{3.21}$$

où $\mathbf{G'}_k$ est donnée par $\mathbf{G'}_k = \mathbf{G}_k(1 - \mathbf{G}_k)$ avec $\mathbf{G}_k$ définie comme la matrice diagonale $N \times N$ construite à partir de $g(s_k(j))$ pour $j = 1, \ldots, N$. L'inverse de $\partial \mathbf{h}/\partial \mathbf{v}$ ne s'exprime explicitement que pour $D = 1$: **les approches FFP et GF sont donc exclues**. On a également

$$\frac{\partial \mathbf{h}}{\partial \mathbf{w}^d} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \mathbf{G}'_d \mathbf{V}_0 \\ \mathbf{G}'_{d+1} \mathbf{V}_1 \\ \vdots \\ \mathbf{G}'_K \mathbf{V}_{K-d} \end{pmatrix}, \quad \forall d = 1, \ldots, D. \tag{3.22}$$

où d-ième bloc, $\mathbf{G}'_d \mathbf{V}_0$, est le premier bloc non-nul.

3.5. La propagation en avant (FP)

Posons $\mathbf{Y}^d = (\partial \mathbf{h}/\partial \mathbf{v})^{-1} \partial \mathbf{h}/\partial \mathbf{w}^d$. Cette grandeur dépend du délai d. $\mathbf{Y}^d$ peut s'écrire en blocs

$$\mathbf{Y}^d = \begin{pmatrix} \mathbf{Y}_1^d \\ \mathbf{Y}_2^d \\ \vdots \\ \mathbf{Y}_K^d \end{pmatrix}, \tag{3.23}$$

où $\mathbf{Y}_k$ est une matrice $N \times N^2$. D'après $(\partial \mathbf{h}/\partial \mathbf{v})\mathbf{Y}^d = \partial \mathbf{h}/\partial \mathbf{w}^d$, il vient pour $k = D+1, \ldots, K$,

$$\mathbf{Y}_k^d = [\mathbf{I} - \mathbf{G}'_k \mathbf{W}^{0^T}]^{-1} \mathbf{G}'_k \left[\sum_{d=1}^{D} \mathbf{W}^{\mathbf{d}^T} \mathbf{Y}_{k-d}^d - \mathbf{V}_{k-d} \right], \tag{3.24}$$

avec les conditions aux limites

$$\mathbf{Y}_j^d = -\mathbf{G}'_j \mathbf{V}_{j-d}, \quad j = 1, \ldots, D. \tag{3.25}$$

Donc dans l'approche 'forward', la récursion se fait dans le sens du temps et le gradient final est obtenu par

$$\frac{\partial^+ E}{\partial \mathbf{w}^d} = -\frac{\partial E}{\partial \mathbf{v}} \mathbf{Y}^d = -\sum_{k=1}^{K} \mathbf{e}_k^T \mathbf{Y}_k^d. \tag{3.26}$$

Remarquons qu'en posant $\mathbf{Y} = (\partial \mathbf{h}/\partial \mathbf{v})^{-1} \partial \mathbf{h}/\partial \mathbf{W}$ avec $\mathbf{W} = (\mathbf{w}^1, \ldots, \mathbf{w}^D)$, (3.24) s'écrit

$$\mathbf{Y}_k = [\mathbf{I} - \mathbf{G'}_k \mathbf{W}^{0^T}]^{-1} \mathbf{G}'_k \left[\sum_{d=1}^{D} \mathbf{W}^{\mathbf{d}^T} \mathbf{Y}_{k-d} - (\mathbf{V}_{k-1}, \ldots, \mathbf{V}_{k-D}) \right]. \tag{3.27}$$

D'où

$$\frac{\partial^+ E}{\partial \mathbf{W}} = -\frac{\partial E}{\partial \mathbf{v}} \mathbf{Y} = -\sum_{k=1}^{K} \mathbf{e}_k^T \mathbf{Y}_k. \tag{3.28}$$

Cet algorithme à propagation avant est la version RTRL [WIL 89] généralisée au DRNN. On remarque que les dérivées $\mathbf{Y}_k$ sont les sorties du réseau original *linéarisé*. La procédure est fortement grevée par la masse de calcul et de mémoire requise. En premier lieu, chaque dérivée doit être stockée ce qui entraîne une capacité de mémorisation de l'ordre de $O(DN^3)$. Ensuite une

quantité draconienne d'opérations, de l'ordre de $O(D^2N^4)$ est nécessaire à chaque itération car l'adaptation les DN^3 dérivées requiert chacune $O(DN)$ opérations.

3.6. La rétro-progation dans le temps (BPTT)

Dans la méthode BPTT, on évalue d'abord $\mathbf{y}^T = -\partial E/\partial \mathbf{v}(\partial \mathbf{h}/\partial \mathbf{v})^{-1}$ avant de le multiplier par la suite par $\partial \mathbf{h}/\partial \mathbf{w}$. Par définition,

$$\frac{\partial E}{\partial \mathbf{v}} = -\mathbf{y}^T \frac{\partial \mathbf{h}}{\partial \mathbf{v}} \tag{3.29}$$

Avec $\mathbf{y}^T = (\mathbf{y}_1^T, \ldots, \mathbf{y}_K^T)$. En substituant les expressions (2.16) et (3.21) dans la formule ci-dessus, il vient

$$\mathbf{y}_k = [\mathbf{I} - \mathbf{W}^0 \mathbf{G}'_k]^{-1} [\mathbf{e}_k + \sum_{j=1}^{D} \mathbf{W}^j \mathbf{G}'_{k+j} \mathbf{y}_{k+j}] \tag{3.30}$$

avec la condition aux limites $\mathbf{y}_K = [\mathbf{I} - \mathbf{W}^0 \mathbf{G}'_k]^{-1} \mathbf{e}_K$ et $\mathbf{y}_j = 0$ si $j > K$. D'après la définition de $\mathbf{y}$

$$\frac{\partial^+ E}{\partial \mathbf{w}^d} = \mathbf{y}^T \frac{\partial \mathbf{h}}{\partial \mathbf{w}^d}. \tag{3.31}$$

Après substitution de $\mathbf{y}$ dans cette l'expression, on obtient

$$\frac{\partial^+ E}{\partial \mathbf{W}^d} = \sum\nolimits_{k=d}^{K} \mathbf{G}'_k \mathbf{y}_k \mathbf{v}_{k-d}^T. \tag{3.32}$$

Posons $\boldsymbol{\delta}_k = \mathbf{G}'_k \mathbf{y}_k$, on retrouve une généralisation de la règle delta pour un réseau non-bouclé.

$$\frac{\partial^+ E}{\partial \mathbf{W}^d} = \sum_{k=d}^{K} \boldsymbol{\delta}_k \mathbf{v}_{k-d}^T. \tag{3.33}$$

La rétro-progation dans le temps (BPTT) est très efficace : sa version en mode *batch* est de l'ordre de $O(N^2)$. Toutefois, la mémoire requise varie en $O(K)$, il est peu pratique de déplier le réseau complètement, une *troncature* est souvent nécessaire. Figure (3.1) illustre le processus de dépliage de l'adjoint. Nous venons de démontrer le résultat suivant [AUS 95b],

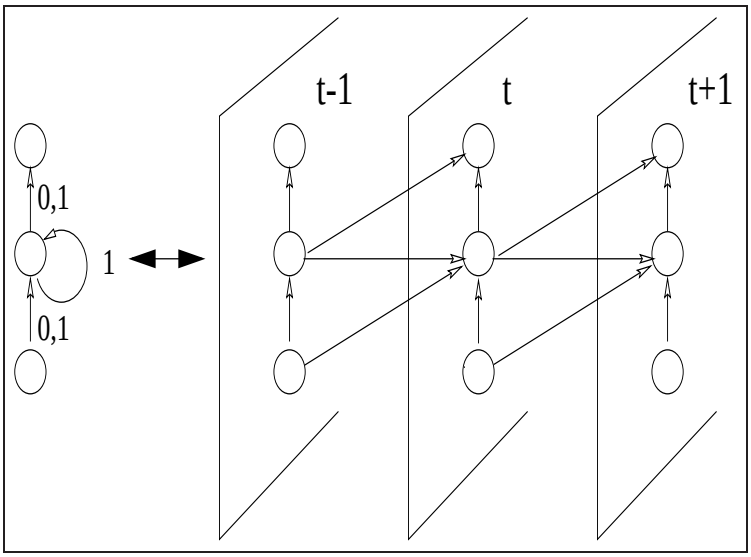


Figure 3.1. *Illustration du processus de dépliage pour un DRNN simple constitué d'une entrée, un neurone caché et une sortie. Les valuations des arcs désignent le(s) délai(s). A gauche, le réseau original. A droite, le réseau correspondant dit* adjoint, *ou* à propagation d'erreur, *déplié sans délai.*

Théorème 3 - *Si le DRNN est convergent sur la trajectoire* $0 \leq k \leq K$, *alors le gradient de l'erreur* $E = \frac{1}{2}\sum_{k=0}^{K} \mathbf{e}_k^T \mathbf{e}_k$ *se calcule à l'instant* K *par*

$$\frac{\partial^+ E}{\partial \mathbf{W}^d} = \sum_{k=d}^{K} \boldsymbol{\delta}_k \mathbf{v}_{k-d}^T, \quad \forall d \leq D \tag{3.34}$$

où la séquence des vecteurs $\boldsymbol{\delta}_k$, *pour* $k = d, \ldots, K$, *sont donnés par* $\boldsymbol{\delta}_k = \mathbf{G'}_k \mathbf{y}_k$ *et la séquence des vecteurs* $\mathbf{y}_k$ *sont les solutions des équations aux différences :*

$$\mathbf{y}_k = [\mathbf{I} - \mathbf{W}^0 \mathbf{G}'_k]^{-1} [\mathbf{e}_k + \sum_{j=1}^{D} \mathbf{W}^j \mathbf{G}'_{k+j} \mathbf{y}_{k+j}] \tag{3.35}$$

avec les conditions aux limites : $\mathbf{y}_K = [\mathbf{I} - \mathbf{G}'_K \mathbf{W}_K^0]^{-1} \mathbf{e}_K$ *et* $\mathbf{y}_j = \mathbf{0}$ *pour* $j > K - D$.

La procédure à laquelle nous arrivons englobe la rétro-propagation temporelle pour les réseaux FIR non-bouclés [WAN 93], la rétro-propagation récurrente pour des réseaux statiques récurrents [ALM 87, PIN 87], et la rétro-propagation dans le temps (BPTT) pour des réseaux bouclés standards [RUM 86].

3.7. Conclusion

Des conditions suffisantes d'existence et de stabilité asymptotique des points d'équilibre, à chaque instant, ont été établies pour une classe générale de réseaux bouclés à délais dont les synapses sont représentées par des filtres linéaires à réponse impulsionnelle finie. Des conditions suffisantes d'existence et de stabilité globale asymptotique de ces réseaux en bouclage fermé a aussi été exhibée. Les versions FD et BPTT ont été obtenues comme au chapitre précédent. Seule la version BPTT tronquée présente une complexité temporelle et spatiale acceptable. Nous discutons au chapitre suivant une manière astucieuse de tronquer le dépliement de l'adjoint, sans trop altérer l'approximation du gradient d'erreur.

Chapitre 4

L'EVANOUISSEMENT DU GRADIENT

4.1. Introduction

Les réseaux bouclés sont réputés incapables d'apprendre des dépendances à longue portée, même élémentaires, par descente du gradient. Le problème du *temporal credit assignment* demeure l'un des enjeu majeur de la communauté connexioniste comme en témoigne une récente taxonomie sur les *réseaux connexionistes spatio-temporels* (STCN) [KRE 01]. En effet, la *décroissance du flot arrière du gradient d"erreur* ("gradient error back flow" et notée **GEBF**), rend quasiment impossible l'apprentissage de dépendances à longue portée entre les entrées/sorties par des méthodes fondées sur le gradient. Cette faiblesse qualifiée de *forgetting behavior*, ou "comportement oublieux", est au coeur des préoccupations d'un grand nombre de travaux depuis l'article de Y. Bengio et al. [BEN 94b]. Dans ce chapitre, l'analyse de l'GEBF étend les travaux de [FRA 92, BEN 94b, AUS 95b, HOC 97b, LIN 96] au réseaux FIR bouclés, y compris les réseaux à point fixe, et apporte un éclairage nouveau sur ce problème.

Nous établissons également des conditions suffisantes pour garantir la convergence de l'GEBF vers zéro. Celles-ci s'expriment explicitement en fonction de la matrice de poids et s'appliquent à de nombreux les réseaux bouclés introduits dans la littérature ces dernières années [KRE 01, TSO 94]. Une borne supérieure sur le nombre de rétro-propagations dans le temps est a été établie pour limiter l'erreur de (BPTT). Ces résultats théoriques serons confirmés par simulation. On mettra en évidence une relation empirique entre l'erreur de troncature de BPTT et le coefficient de décroissance du gradient.

4.2. Conditions de convergence du gradient

BPTT est un algorithme réputé *off-line* car la démultiplication du réseau adjoint doit être répétée *ad infinitum*, c'est-à-dire jusqu'à la première itération ce qui est rédhibitoire du point de vue de l'implémentation. La troncature du réseau déplié est nécessaire. Nous allons montrer que le coût des duplications successives de l'adjoint peuvent être maintenue dans des limites acceptables tout en calculant le gradient d'erreur avec une précision désirée.

Revenons sur l'équation récursive de BPTT (3.30)

$$\mathbf{y}_k = [\mathbf{I} - \mathbf{W}^0 \mathbf{G}'_k]^{-1} [\mathbf{e}_k + \sum_{j=1}^{D} \mathbf{W}^j \mathbf{G}'_{k+j} \mathbf{y}_{k+j}] \tag{4.1}$$

$\mathbf{y}_k$ est la somme de deux termes : le premier est dû à l'erreur courante, $\mathbf{e}_k$, quant au second, il correspond à une erreur additionnelle qui est, par essence, l'erreur équivalente transposée à l'instant présent, k, due à l'influence du vecteur d'état courant sur l'évolution future du réseau de neurones. Soit $\mathbf{y}_k^{k+n}$ la solution de Eq. (3.30) à l'instant k quand tous les $\mathbf{e}_{k+j}$ sont mis à $\mathbf{0}$ sauf à l'instant $j = n$. Soit $\boldsymbol{\delta}_k^{k+n} = \mathbf{G'}_k \mathbf{y}_k^{k+n}$. En incorporant les termes $\mathbf{y}_k^{k+n}$ et $\boldsymbol{\delta}_k^{k+n}$ dans la règle delta (3.34), il vient par linéarité,

$$\Delta \mathbf{W}^d = \eta \sum_{k=d}^{K} \boldsymbol{\delta}_k \mathbf{v}_{k-d}^T = \eta \sum_{n=d}^{K} \sum_{k=d}^{n} \boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T \tag{4.2}$$

En mode *batch*, il faut s'interroger sur la convergence de cette somme lorsque $K \to \infty$. Il est difficile de répondre à cette question car l'évolution des poids dépend des données. En pratique les poids peuvent atteindre leur limite de saturation ou diverger. Mais en mode en-ligne, les poids sont ajustés à chaque passe. A l'itération n, les poids sont ajustés en sommant les $\boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T$ où l'index k porte sur les instants précédents, i.e. $\sum_{k=d}^{n} \boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T$. Il reste à savoir si cette somme diverge ou reste bornée quand $n \to \infty$.

Pour assurer la convergence il faut non seulement que $\boldsymbol{\delta}_{n-j}^n$ tende vers zéro par rapport à j mais la décroissance doit être suffisamment rapide. Plus explicitement,

Définition 1 - *La descente de gradient est dite convergente à chaque instant intermédiare si* $\lim_{n\to\infty} \sum_{k=d}^{n} \boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T$ *existe.*

Lorsque la descente de gradient n'est pas convergente à l'instant k le gradient d'erreur, quelque soit l'algorithme qui le calcule, n'existe pas. Les équations récursives sur lesquelles repose par exemple RTRL, sont alors instables.

Restreignons donc notre attention au *gradient error back flow*, GEBF, formellement défini par,

Définition 2 - *Le GEBF du à l'erreur instantanée E_n, est constitué de la séquence de vecteurs, $\boldsymbol{\delta}_k^n$ pour $k = n, n-1, \ldots, 1$, dans Eq. 4.2 pour ajuster les poids.*

Pour E_n fixé à l'instant n, l'GEBF peut être vu intuitivement comme la séquence de vecteurs qui propage l'information qui mesure à l'impact des paramètres aux instants passés sur l'erreur courante.

Dans le but d'établir des conditions suffisantes pour assurer la converge de la somme, démontrons le résultat suivant [AUS 96],

Théorème 4 - *Pour un coût instantané $E_n = \frac{1}{2}{\mathbf{e}_n}^T\mathbf{e}_n$, considérons le vecteur d' erreur ${\mathbf{y}^n_{n-k}}^T = \frac{\partial E_n}{\partial \mathbf{v}_{n-k}}[\mathbf{I} - \mathbf{G'}_{n-k}{\mathbf{W}^0}^T]^{-1}$. Soit $K > 0$ la durée de l'époque et $\mu = max(g'(h))$. Soit*

$$a_d = \max_k\left(\frac{\mu\|\mathbf{W}^d\|}{1-\mu\|\mathbf{W}^0\|}\right), \quad 1 \leq d \leq D \tag{4.3}$$

Si $\rho = \sum_{d=1}^{D} a_d < 1$ **alors** $\|\mathbf{y}^K_{K-(kD+j)}\| < \rho^{k+1}\|\mathbf{y}^K_K\|$, $\forall k \geq 0, \forall j = 1, \ldots, D$.

Preuve : Le comportement de $\mathbf{y}_k^K$ se déduit des équations récursives arrières (3.30). Il suffit de poser $\mathbf{e}_j = 0$ pour $j = 0, 1, \ldots, K-1$. Il vient,

$$\mathbf{y}_k^K = [\mathbf{I} - \mathbf{W}^0\mathbf{G'}_k]^{-1}\sum_{d=1}^{D}\mathbf{W}^d\mathbf{G'}_{k+d}\mathbf{y}^K_{k+d} \tag{4.4}$$

pour $k < K - D$, avec les conditions aux limites suivantes $\mathbf{y}^K_K = [\mathbf{I} - \mathbf{W}^0\mathbf{G'}_k]^{-1}\mathbf{e}_K$. L'équation vectorielle (4.4) gouverne la dynamique des termes erreurs. Soit $\mu = \max(g'(h))$ de sorte que $\|\mathbf{G'}_k\| < \mu$, $\forall k$. Supposons que $\|\mathbf{W}^0\| < 1/\mu$ pour $k = 0 \ldots, K$ de sorte que la convergence de $\mathbf{I} + \mathbf{W}^0 \cdot \mathbf{G'}_k + (\mathbf{W}^0 \cdot \mathbf{G'}_k)^2 + \ldots$ soit assurée ($\|\ \|$ est une norme matricielle *compatible*, i.e., $\|\mathbf{A}\|_p = \sup_{x\neq 0}\|\mathbf{A}x\|_p/\|x\|_p, p = 1, 2, \infty$). Donc à chaque itération $k = 0, \ldots, K-1$, on a

$$\begin{aligned} \|\mathbf{y}_k^K\| &\leq \|\mathbf{I} - \mathbf{W}^0\mathbf{G'}_k\|^{-1} \sum_{d=1}^{D} \|\mathbf{W}^d\| \, \|\mathbf{G'}_{k+d}\| \, \|\mathbf{y}_{k+d}^K\| \\ &\leq \frac{\mu}{1-\mu\|\mathbf{W}^0\|} \sum_{d=1}^{D} \|\mathbf{W}^d\| \, \|\mathbf{y}_{k+d}^K\| \end{aligned} \tag{4.5}$$

Soit $a_d = \max_k(\frac{\mu\|\mathbf{W}^d\|}{1-\mu\|\mathbf{W}^0\|})$, $1 \leq d \leq D$. D'après (4.4) et (4.5), on déduit les inégalités suivantes

$$\|\mathbf{y}_k^K\| \leq a_1\|\mathbf{y}_{k+1}^K\| + a_2\|\mathbf{y}_{k+2}^K\| + \ldots + a_D\|\mathbf{y}_{k+D}^K\| \tag{4.6}$$

La dépendance au temps a été éliminée. L'analyse de la suite $\{u_n\}$, définie par $u_n = \sum_{j=1}^{D} a_j u_{n-j}$, doit être mené car $u_n \geq \|\mathbf{y}_{K-n}^K\|$ for $n = 0, \ldots, K$, avec les conditions aux limites $u_n = \|\mathbf{y}_{K-n}^K\|$ pour $n = 0, \ldots, D-1$. Comme $u_n < \max_{j=1\ldots D}(u_{n-j}) \cdot \sum_{j=1}^{D} a_j$, et considérant que $u_j < (\sum_{j=1}^{D} a_j) \cdot u_0$ pour $j > 0$ et $u_0 = \|y_K^K\|$, il vient par récurrence

$$\|y_{K-(kD+j+1)}^K\| \leq u_{kD+j+1} < (\sum_{j=1}^{D} a_j)^{k+1}\|y_K^K\|, \quad \forall k \geq 0, \forall j = 0, \ldots, D-1. \tag{4.7}$$

Ainsi, $\sum_{j=1}^{D} a_j < 1$ assure la convergence de la dynamique de rétro-propagation.

□

Notons que la décroissante exponentielle de $\mathbf{y}_{K-n}^K$ est équivalente à celle de l'GEBF δ_{K-n}^K (rappelons que $\delta_k^{k+n} = \mathbf{G'}_k\mathbf{y}_k^{k+n}$ et $\mathbf{G'}_k^{-1}$ existe) et donc aussi équivalente la décroissante exponentielle du gradient de l'erreur par rapport à l'activation passée des neurones, $\partial E_K / \partial \mathbf{v}_{K-n}$.

Le théorème dit en essence, que les termes $\mathbf{y}_{K-n}^K$ sont majoreś par une suite décroissante à vitesse exponentielle par rapport à n si $\rho < 1$, d'où la définition,

Définition 3 - $\rho = \sum_{d=1}^{D} \max_k(\frac{\mu\|\mathbf{W}^d\|}{1-\mu\|\mathbf{W}^0\|})$ *est nommé taux de décroissance exponentiel de l'GEBF (error back flow exponential decay rate) ou simplement, le taux de décroissance du gradient.*

Le taux analytique de décroissance exponentiel de l'GEBF (error back flow exponential decay rate), ρ, majore le vrai GEBF dans le réseau. En batch mode (i.e., lorsque les poids sont ajustés à chaque passe dans la base d'apprentissage), $\rho < 1$ se traduit simplement par

Corollaire 1 - *En batch mode,* $\sum_{d=0}^{D} \|\mathbf{W}^d\| < 1/\mu$ *est une condition suffisante pour assurer la convergence de la procédure d'apprentissage.*

Cette inégalité pénalise les poids élevés, quelque soient les délais et la connectivité, pour assurer la décroissance de l'GEBF d'où le comportement "oublieux" (*forgetting behavior*) des réseaux bouclés standards. Le taux analytique de décroissance exponentiel de l'GEBF, ρ, peut s'écrire :

$$\rho = \sum_{d=1}^{D} \frac{\mu \|\mathbf{W}^d\|}{1 - \mu \|\mathbf{W}^0\|} \tag{4.8}$$

Avec des normes matricielles compatibles $\|\mathbf{A}\|_1 = \max_j \sum_i |w_{ij}|$, $\|\mathbf{A}\|_\infty = \max_i \sum_j |w_{ij}|$, et $\|\mathbf{A}\|_F = \left[\sum_{i,j} |w_{ij}|^2\right]^{1/2}$, on obtient trois conditions facilement interprétables pour assurer la convergence de la procédure d'apprentissage :

$$\begin{cases} \mu \sum_{d=0}^{D} \max_i \left(\sum_j |w_{ij}^d| \right) < 1 \\ \mu \sum_{d=0}^{D} \max_j \left(\sum_i |w_{ij}^d| \right) < 1 \\ \mu \sum_{d=0}^{D} \left[\sum_{i,j} |w_{ij}^d|^2 \right]^{1/2} < 1 \end{cases} \tag{4.9}$$

Ces conditions non mutuellement exclusives, sont satisfaites, par exemple, lorsque tous les poids sont dans l'intervalle $\pm \frac{1}{\mu N(D+1)}$.

On observe que (4.9) implique (3.4) et (3.9). En d'autres termes, (4.9) garantit que,

1. un point d'équilibre unique globalement asymptotiquement stable existe à chaque instant,
2. le réseau converge vers un point fixe en bouclage fermé quelque soit les conditions initiales,

3. la descente du gradient est convergente à chaque itération.

Résultats similaires - Les conditions sont en accord avec les résultats obtenus par Hochreiter et Schmidthuber en 1995 pour les réseaux bouclés standards [WIL 89] avec une fonction d'activation $g(s) = 1/(1 + e^{-\beta s})$. Par exemple, si N est l'ordre de la matrice, ils trouvent que pour $\forall i, j,\ |w_{ij}| < w_{max} < 4/N$ et $\beta = 1$, le taux de décroissance de l'GEBF est majoré par $\tau = N w_{max}/4 < 1$. $\mathbf{W}^0 = 0$ et $D = 1$ transforme le DRNN en un réseau bouclé standard, i.e., avec des délais fixés à 1. Pour les fonctions de transfert de type sigmoïde, $\mathbf{G'}_k$ est une matrice diagonale de taille $N \times N$ qui découle de $g'(h) = \beta g(h)(1 - g(h))$ de sorte que $g'(h) < \beta/4$. Donc $\mu = 1/4$. En Conséquence, $\rho < 1$ se réduit à une condition plus faible $\|\mathbf{W}^1\| < 4$ pour tout k ($\beta = 1$), car plusieurs poids peuvent dépasser w_{max}. Inversement, il est clair que $\|\mathbf{W}^1\| < 4$ est vérifié si $|w_{ij}| < 4/n$ car $\|\mathbf{W}\| < N w_{max}$. Par ailleurs, la condition d'oubli de Frasconi et al.'s 1992 (forgetting behavior condition, [FRA 92]), i.e., $|w_{jj}| < 1/d$ avec $d = \max_h (g'(h)) = 4$, pour les réseau à bouclage local est aussi un cas particulier du Théorème 2.

Il est aussi intéressant de noter que l'existence de $[\mathbf{I} - \mathbf{G'}_k \mathbf{W}^0]^{-1}$, et donc de la *stabilité* des points fixes est aussi garantie. Il suffit de poser $D = 0$, le Corollaire 1 implique $\|\mathbf{W}^0\| \cdot \|\mathbf{G'}_k\| < 1$, ce qui est clairement une condition suffisante pour l' existence de $[\mathbf{I} - \mathbf{G'}_k \mathbf{W}^0]^{-1}$.

Lien avec la régularisation - Pour garantir la convergence de l'GEBF, on souhaite contraindre le modèle en résolvant

$$w = \arg\min_w E(w) \quad \text{sous les contraintes} \quad \sum_{d=0}^{D} \|\mathbf{W}^d\| \leq \frac{1}{\mu} \tag{4.10}$$

La contrainte définie une structure des sous-ensembles des fonction paramétrées. En fait, $R(w) = \sum_{d=0}^{D} \|\mathbf{W}^d\|$ est un terme de *réguralisation* puisque R est continue, positive, une solution au problème d'optimisation existe, et $\forall c \geq 0$, $\{w \mid R(w) \leq c\}$ sont des ensembles compacts [GOU 97]. Selon Khun-Tucker[1], il y a une équivalence implicite entre la résolution de (4.10) et la minimisation d'une version modifiée du coût :

$$w = \arg\min_w \left(E(w) + \xi \sum_{d=0}^{D} \|\mathbf{W}^d\| \right) \tag{4.11}$$

[1] prendre une norme différentiable dans (4.10), e.g., la norme de Frobenius

Le second terme défavorise indirectement les fonctions aux variations abruptes. Il agit comme le *weight decay* car il provoque une décroissante exponentielle des poids vers zéro. Le paramètre de *régularisation* , ξ, appelé *hyper paramètre*, instaure un compromis entre la nécessité de minimiser l'erreur et celle de rester dans un sous-ensemble compact. Toutefois, Il n'existe pas de méthode universelle pour déterminer le bon niveau de régularisation, i.e. la valeur de ξ [BIS 95] . Un certain nombre de méthodes existent basées sur l'estimation de l'erreur de généralisation (techniques basées sur les ensembles de validation ou estimation algébrique) [GOU 97]. Les liens entre la régularisation et l'inférence Bayésienne sont bien établis. Dans le cas gaussien, minimiser le coût régularisé revient à maximiser la probabilité *a posteriori* des poids car

$$-\ln P(w \mid \mathcal{D}, \alpha) \propto E(w) - \frac{\sigma^2}{N} \ln P(w \mid \alpha) + cte \tag{4.12}$$

où w est le vecteur de tous les poids concaténés, $\mathcal{D}$ est l'ensemble d'apprentissage, σ est la variance des données et N le nombre de neurones. Ainsi, minimiser (4.11) revient à to maximiser la solution *a posteriori* en supposant une probabilité *a piori* des poids suivante :

$$P(w \mid \alpha) \propto \prod_{d=0}^{D} \exp\left(-\alpha \|\mathbf{W}^d\|\right) \tag{4.13}$$

où l'hyper-paramètre, α, parametrise la distribution. Le *prior* résume l'état de nos connaissances sur les poids : ici, les poids faibles sont privilégiés. Ainsi, l'analyse de la convergence de l'apprentissage par descente du gradient et l'analyse statistique des modèles des RN concordent : les faibles poids sont préférables. Quelque soit le *prior* (e.g., Gaussien, Laplace [GOU 97]), la régularisation dite formelle *stabilise* la descente du gradient en forçant indirectement la *décroissance* du gradient dans l'adjoint déplié.

4.3. La troncature de BPTT

Il est dorénavant possible d'estimer le nombre de rétro-propagations requisent par la BPTT pour approximer le gradient d'erreur avec une précision donnée et réduire ainsi significativement les calculs.

Si D est le délai synaptique maximum, supposons que l'adjoint, à l'itération n, est déplié sD fois où s est un entier positif. Donc $\Delta\hat{\mathbf{W}}^d_n(s) = \eta \sum_{k=n-sD}^{n} \boldsymbol{\delta}^n_k \mathbf{v}^T_{k-d}$ estime la vraie adaptation de la matrice de poids $\Delta\mathbf{W}^d_n(s) = \eta \sum_{k=d}^{n} \boldsymbol{\delta}^n_k \mathbf{v}^T_{k-d}$ à l'instant $n > sD + d$.

On cherche une condition sur les paramètres du réseau pour que $\|\Delta\hat{\mathbf{W}}_n^d(s) - \Delta\mathbf{W}_n^d\| < \epsilon$, pour tout $\epsilon > 0$. Le théorème suivant [AUS 02b] exprime s en fonction de ϵ, la précision requise sur les ajustements des poids :

Théorème 5 - *En batch mode, il suffit, à l'itération n, de rétro-propager sD l'adjoint, où s est le plus petit entier vérifiant*

$$s > \frac{1}{\ln \rho} \cdot \ln \left[\frac{\epsilon(1-\rho)(1-\mu\|\mathbf{W}^0\|)}{\eta\mu ND\|\mathbf{e}_n\|} \right] \tag{4.14}$$

pour garantir que les poids sont ajustés avec une précision ϵ.

Preuve : Soit n l'itération courante. Soit $\Delta\hat{\mathbf{W}}_n^d(s) = \eta \sum_{k=n-sD}^{n} \boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T$, pour $n > sD + d$, l'estimateur de la matrice d'ajustement des poids àl'instant n obtenue après sD duplications de l'adjoint, tel que $\Delta\hat{\mathbf{W}}_{sD+d}^d(s) = \Delta\mathbf{W}_{sD+d}^d$. D'après (4.2), on a $\forall d \leq D$, $\forall s > 0$,

$$\|\Delta\hat{\mathbf{W}}_n^d(s) - \Delta\mathbf{W}_n^d\| = \eta \sum_{k=d}^{n-sD-1} \boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T \tag{4.15}$$

Pour $n > sD + d$, il vient

$$\begin{aligned}
\sum_{k=d}^{n-sD-1} \boldsymbol{\delta}_k^n \mathbf{v}_{k-d}^T &\leq \sum_{k=d}^{n-sD-1} \|\mathbf{G'}_k\| \, \|\mathbf{y}_k^n\| \, \|\mathbf{v}_{k-d}\| \\
&\leq \mu \sum_{k=d}^{n-sD-1} \|\mathbf{y}_k^n\| \, \|\mathbf{v}_{k-d}\| \\
&\leq \mu N \sum_{k=d}^{n-sD-1} \|\mathbf{y}_k^n\| \\
&\leq \mu ND\|\mathbf{y}_n^n\| \frac{\rho^{s+1}}{1-\rho}
\end{aligned} \tag{4.16}$$

Les inégalités découlent du Théorème 4 et de $\|\mathbf{v}_{k-d}\|_p < N$ pour $p = 1, 2, \infty$. Le Théorème 6 est obtenu facilement de ce qui précède et de l'inégalité suivant

$$\|\mathbf{y}_n^n\| \leq \|\mathbf{I} - \mathbf{W}_k^0 \mathbf{G'}_k\|^{-1} \|\mathbf{e}_n\| \leq \frac{\|\mathbf{e}_n\|}{1 - \mu\|\mathbf{W}^0\|} \tag{4.17}$$

□

Notons de plus que $\|\mathbf{W}^0\| < Nw_{max}$; la dépendance sur n (i.e., l'instant auquel le gradient est rétro-propagé) peut être éliminée en observant que

$$\|\mathbf{e}_n\| \leq N_{out} \cdot \max_j(e_n(j)) \leq N_{out} \tag{4.18}$$

où N_{out} est le nombre de sorties.

La formule (4.14) permet d'estimer un majorant du nombre de rétro-propagations requisent par la procédure pour maintenir une erreur négligeable dans l'évaluation du gradient d'erreur.

Remarques - Remarquons que la dépendance de s à ρ prend la forme $\ln(1-\rho)/\ln\rho$ (voir Fig. 4.1), et donc diverge rapidement pour des grandes valeurs $\rho > 0.9$. $s \to \infty$ quand $\rho \to 1$, et $s \to 1$ quand $\rho \to 0$. Notons aussi le nombre de rétro-propagations, pour garantir une précision donnée sur l'ajustement des poids, varie en $\ln ND$, lorsque ρ et $\|\mathbf{W}^0\|$ sont fixés.

4.3.1. *Complexité de BPTT tronqué*

A la lumière de la résultats précédent, il est légitime d'exploiter la décroissance exponentielle de l'GEBF pour se limiter à sD dépliements du réseau adjoint, tout en garantissant une borne sur l'erreur d'approximation du gradient d'erreur. Supposons ρ constant, s varie alors en $O(\ln(DN))$. Il faut donc de l'ordre de $N\ln(DN)$ pour mémoriser les états antérieurs du réseau et DN^2 pour stocker les poids. A chaque itération, l'ajustement d'un paramètre exige de l'ordre de $O(DN^2)$ opérations, soit $O(DN^2\ln(DN))$ opérations pour tous les paramètres. En contraste, l'approche *forward* nécessite un stockage en $O(DN^3)$ et la complexité en temporelle varie en $O(D^2N^4)$, et l'approche par blocs (BU) - détaillée au paragraphe 2 -, réduit ce nombre d'opérations à $O(D^2N^3)$, au détriment de la localité des opérations.

Il faut cependant insister sur un point important : les algorithmes que nous avons évoqués, **BPTT tronqué** en particulier, **capitulent sur le problème des dépendances longue portée**. Pire, BPTT exploite le déclin du gradient (*vanishing gradient*) pour réduire le nombre d'opérations.

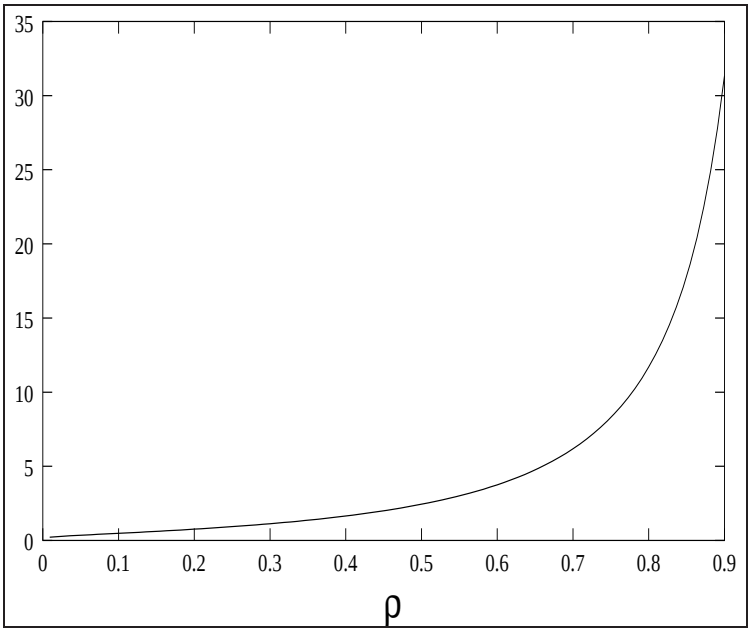


Figure 4.1. *Accroissement typique du nombre de rétro-propagations (i.e., sD), selon le théorème 6, pour garantir une précision donnée sur l'ajustement des poids, en fonction de ρ. Voir Eq. 4.14.*

4.3.2. *Applications numériques*

Pour illustrer l'analyse menée dans ce paragraphe, des applications numériques sont menées avec quelques réseaux bouclés standards.

L'architecture de Williams et Zipser : considérons de prime abord le réseau bouclé de Williams et Zipser (1989) obtenu en imposant $\mathbf{W}^0 = 0$ et $D = 1$ (toutes les synapses sont retardées d'une unité de temps). Des valeurs usuelles comme $N = 10, D = 1, \mu = 1/4, \eta = 10^{-2}, \epsilon = 10^{-6}, \rho = 10^{-1}$, avec une seule sortie, donne un résultat cohérent de s égal à 5 dans la formule ci-dessus. En d'autres termes, il suffit de rétro-propager 5 fois l'adjoint pour assurer que l'erreur absolue de l'ajustement des poids à chaque itération est inférieure à 10^{-6} (i.e., $\|\Delta\hat{\mathbf{W}}_n^d(s) - \Delta\mathbf{W}_n^d\| < 10^{-6}$). Maintenant, supposons que ρ vale 0.5, nous obtenons $s = 17$ et pour $\rho = 0.9$ $s = 128$.

NARMAX : ces réseaux bouclés peuvent être vus comme des réseaux non bouclés (statiques) avec une seule sortie $z(t)$ et M entrées additionnelles qui sont les sorties retardées $z(t-1), \ldots, z(t-D)$. Les réseaux NARMAX sont des cas particuliers de de DRNN pour lesquels des synapses à délai $(d = 1, \ldots, D)$ connectent la sortie aux neurones cachés. Des valeurs comme $N = 10, D = 5, \mu = 1/4, \eta = 10^{-2}, \epsilon = 10^{-6}, \|\mathbf{W}^0\| = 1, \rho = 10^{-1}$, donnent s égal à 6. Pour $\rho = 0.5$ et $\rho = 0.9$, on obtient $s = 19$ et $s = 144$ respectivement.

DRNN : considérons un DRNN totalement récurrent de taille $N = 100, D = 1$ avec 10 sorties ($N_{out} = 10$) et une précision de $\epsilon = 10^{-8}$ et $\mu = 1/4, \eta = 10^{-2}, \|\mathbf{W}^0\| = 1$. Pour $\rho = 10^{-1}$, on obtient $s = 10$. Pour $\rho = 0.5$ et $\rho = 0.9$, on obtient $s = 32$ et $s = 233$ respectivement.

Comme prévu, une augmentation de ρ se traduit par une augmentations drastique de s, rendant le Corollaire 2 inutile dans certains cas. Aussi, l'utilité de ρ doit être confirmée expérimentalement.

4.4. Expérimentations

Dans ce chapitre, quelques expérimentations sont menées pour valider les résultats analytiques obtenus concernant le taux de décroissance du gradient et l'erreur de troncature. Une architecture générique $1 \times N \times 1$, décrite Fig. 4.2, est employée. Les N neurones cachés sont totalement bouclés à l'aide de connexions auto-régressives d'ordre D.

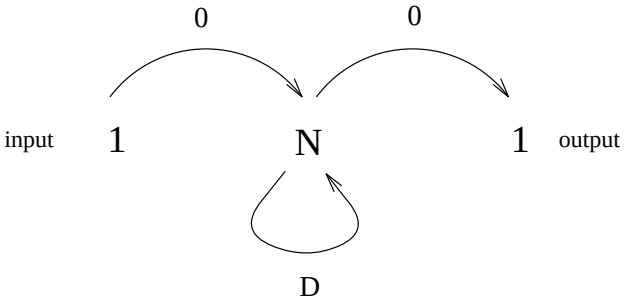


Figure 4.2. *Représentation schématique de la configuration du DRNN utilisé pour la prédiction des sunspots. Les flèches représentent une connectivité totale entre les* N *neurones cachés et* D *désigne l'ordre des filtres.*

4.4.1. *Déclin du gradient*

La suite des sunspots figure parmi les benchmarks incontournables sur lesquels tester son modèle sur des prédictions à un pas en avant (voir les références dans [AUS 95a]. Dans ce qui suit, cependant, nous cherchons une confirmation expérimentale du déclin de l'GEBF pour des configurations variables du DRNN. C'est à dessein que nous nous limitons à de petits modèles contraints d'exploiter leur mémoire interne pour pallier le manque d'information présentée en entrée. L'accent est mis sur le codage et la capacité de mémorisation plutôt que sur la performance pure.

Mode opératoire - Une architecture générique $1 \times N \times 1$, décrite Fig. 4.2, est employée. Les N neurones cachés sont totalement bouclés à l'aide de connexions auto-régressives d'ordre D. Une connexion de délai 1 est ajoutée entre la sortie vers tous les neurones cachés. $\mathbf{W}^0$ est quelconque, la relaxation vers un point fixe est nécessaire. L'initialisation des poids est aléatoire mais vérifie les conditions (4.9). C'est à dessein que nous nous limitons à de petits modèles contraints d'exploiter leur mémoire interne pour pallier le manque d'information présentée en entrée. L'accent est mis sur le codage et la capacité de mémorisation plutôt que sur la performance pure. Les réseaux sont entraînés à prédire la suite à un pas en avant sur les moyennes annuelles normalisées de 1720 à 1979. La version tronquée de BPTT pour le DRNN est utilisée pour le calcul du gradient. L'algorithme du gradient stochastique est utilisé. La version tronquée de BPTT est utilisée pour le calcul du gradient. L'apprentissage est semi-dirigé. L'algorithme opère *en ligne*. La méthode de *weight élimination* [WEI 90] est utilisée. L'initialisation des poids est aléatoire entre -1.0 et $+1.0$

Résultats - Fig. 4.3 illustre le comportement des termes $\|\mathbf{y}_{n-j}^n\|_\infty$ moyennés sur la base d'apprentissage pour $n = 1, \ldots, K$. Une représentation logarithmique sied mieux à la décroissance observée en fonction du temps rétrograde j dans le réseau adjoint déplié. Typiquement, la norme du gradient décroît linéairement dès la première passe en oscillant légèrement autour d'une droite jusqu'à $-\infty$.

Le critère d'arrêt défini en (4.14) est approprié et l'erreur d'estimation du gradient peut se lire sur la courbe de décroissance. Les simulations corroborent la présente étude. Le nombre de propagations rétrogrades dans l'adjoint varient de 5 à 45, 15 en moyenne avec $\epsilon = 10^{-3}$, confirmant en cela l'allégation antérieure que le nombre de passes s, défini dans (4.14), est de l'ordre de D

A titre illustratif, l'histogramme du nombre moyen de rétro-propagations dans le réseau adjoint associé au réseau $N = 2, D = 5$ est tracé Fig. 4.4. La courbe fluctue de 15 à 45, centrée autour de 30, c'est-à-dire, 6 fois l'ordre des filtres $D = 5$. 45 (i.e. $9 \times D$) vecteurs d'état passés du réseau suffisent à estimer correctement le gradient d'erreur avec une précision de 10^{-3}. De même, l'histogramme de la *variation moyenne* du nombre de rétro-propagations dans le réseau adjoint est tracé Fig. 4.5, indiquant que Δs entre 2 itérations se situe le plus souvent dans l'intervalle $[-D, +D]$. Le taux de décroissance du GEBF varie relativement peu au fil des itérations.

Le modèle présente de 31 paramètres présente un compromis taille/performance intéressant [AUS 95b]. A titre illustratif, sa représentation d'état est explicitée ci-dessous sous la forme d'un couple équations à point fixe :

$$x_{t+1} = g(6.129 z_t^1 - 3.641 z_t^2 - 3.627) \tag{4.19}$$

où z_t^1 et z_t^2 sont les activations des deux neurones cachés. Le modèle est régi par le système d'équations

$$\begin{aligned}
z_t^1 &= g(2.440 z_t^1 - 1.152 z_{t-1}^1 + 0.350 z_{t-2}^1 - 0.115 z_{t-3}^1 + 0.041 z_{t-4}^1 - 0.067 z_{t-5}^1 \\
&\quad -0.066 z_t^2 - 0.355 z_{t-1}^2 + 0.097 z_{t-2}^2 - 0.678 z_{t-3}^2 + 0.094 z_{t-4}^2 - 0.118 z_{t-5}^2 \\
&\quad +3.816 x_t - 0.072) \\
z_t^2 &= g(+1.286 z_t^2 + 0.813 z_{t-1}^2 - 0.171 z_{t-2}^2 - 0.120 z_{t-3}^2 - 0.491 z_{t-4}^2 - 0.180 z_{t-5}^2 \\
&\quad -2.296 z_t^1 - 2.948 z_{t-1}^1 + 0.583 z_{t-2}^1 - 0.342 z_{t-3}^1 - 0.112 z_{t-5}^1 \\
&\quad -0.940 x_t - 0.405)
\end{aligned} \tag{4.20}$$

Ce système à point fixe, constitué de 2 neurones cachés et d'une seule entrée, présente une NMSE de 0.091 sur la période 1921-1955 et une NMSE de 0.091 sur la période 1955-1979 (des comparaisons de performance sur ce benchmark

classique sont faites dans ma thèse [AUS 96]). La dernière valeur x_{t-1} associée au deux variables d'état résument tout le passé du processus.

4.4.2. *Erreur de troncature de BPTT*

Dans ce paragraphe, la précision de la borne ρ sur la décroissante de l'GEBF est évaluée empiriquement en fonction de N et D. A cette fin, une mesure appelée le *taux d'erreur de troncature* (TLR) (voir Fig. 4.6) est introduite.

Taux d'erreur de troncature - Supposons donnée l'erreur ϵ sur calcul du gradient. Soit k_1 le véritable nombre minimum de rétro-propagations nécessaire pour que $\|\mathbf{y}_{n-j}^n\| < \epsilon$. Soit, $k_2 = s_2 D$, où s_2 est le plus petit entier vérifiant l'inégalité 4.14. Le *truncation length error rate* (TLR) est défini par

$$\mathrm{TLR} = \frac{k_2 - k_1}{k_1}. \tag{4.21}$$

Son interprétation est aisée : un TLR de valeur n se traduit par une propagation de l'EBBF n fois trop loin dans l'adjoint déplié.

Mode opératoire - Une architecture générique $1 \times N \times 1$, décrite Fig. 4.2, est employée. Les N neurones cachés sont totalement bouclés à l'aide de connexions auto-régressives d'ordre D. Les poids sont choisis aléatoirement entre $-\xi$ et $+\xi$ avec $\xi = 10e-2$ tel que $\sum_{d=0}^{D} \|\mathbf{W}^d\| < 1/\mu$ pour assurer la convergence de l'apprentissage (voire le Corollaire 1). La norme matricielle de Frobenius est choisie, elle est compatible avec la norme Euclidienne . 100 couples entrée/sortie sont tirés aléatoirement. A chaque instant, la sortie du DRNN est calculée pour cette entrée, le gradient d'erreur est rétro-propagé dans l'adjoint et le *véritable* taux de décroissance du gradient est mesuré. La pratique montre que des valeurs de poids élevées accentuent la décroissance de gradient. Afin de couvrir tout l'intervalle, la procédure a été répétée ; à chaque instant la valeur de $+\xi$ est incrémentée de $10e-2$. Le taux de décroissance du gradient, moyenné sur 100 essais, augmente également. On stoppe la procédure lorsque le taux se rapproche de 1.

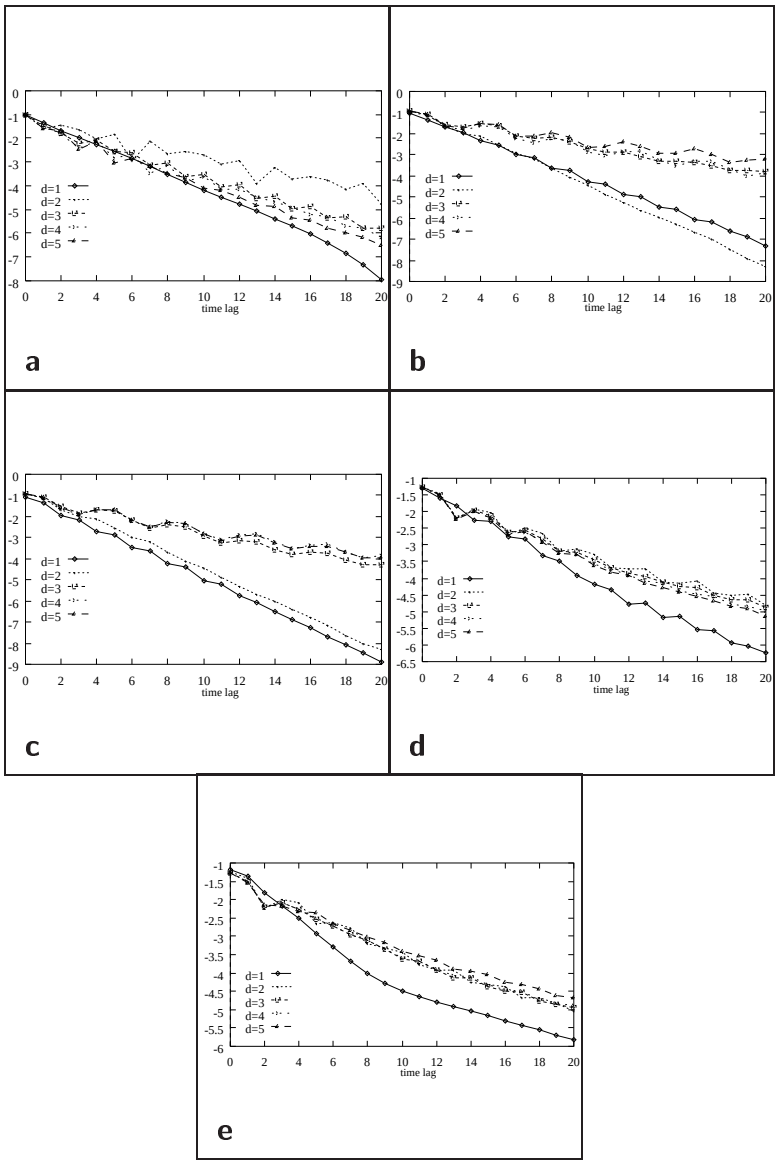


Figure 4.3. *Sunspots :* $< Log_{10}(\|y^n_{n-j}\|_\infty >$*, moyenné sur* n *en fonction du nombre de rétro-propagations* j *dans le réseau adjoint pour plusieurs modèles DRNN de la forme* $1 \times N \times 1$ *avec des connexions d'ordre* $D = 1, \ldots, 5$*. (a)* $N = 1$*, (b)* $N = 2$*, (c)* $N = 3$*, (d)* $N = 4$*, (e)* $N = 5$*.*

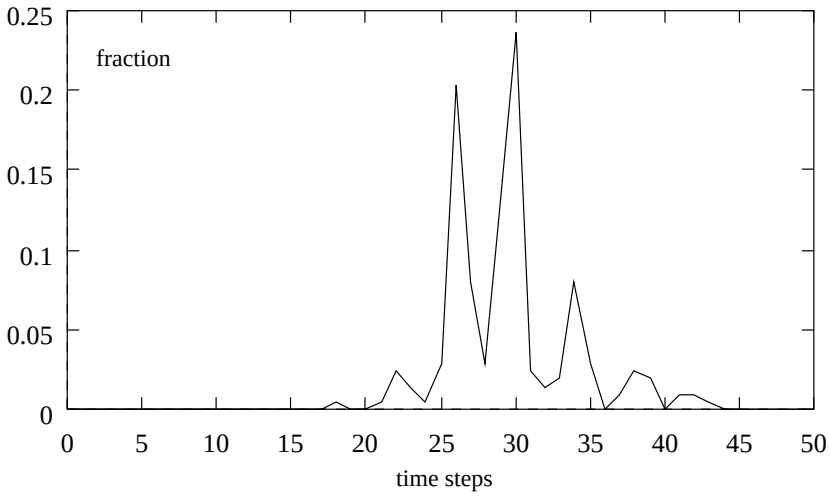


Figure 4.4. *Sunspots : histogramme du nombre moyen de rétro-propagations nécessaires dans le réseau adjoint associé au DRNN* $N = 2, D = 5$.

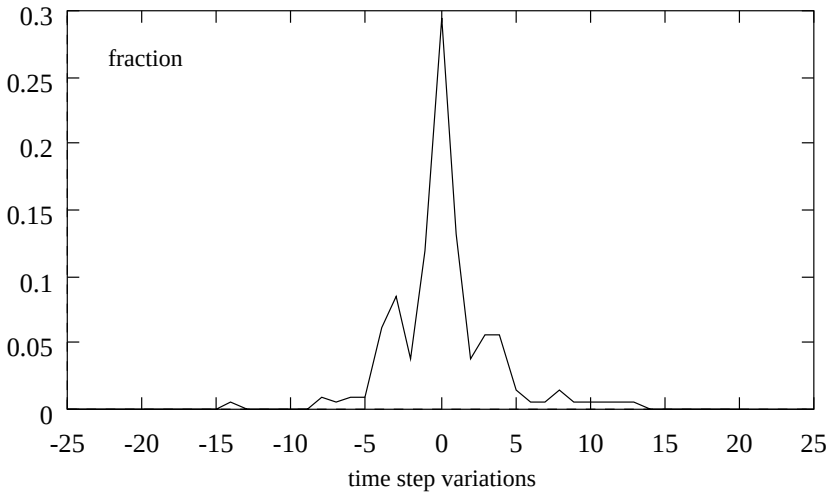


Figure 4.5. *Histogramme de la variation du nombre moyen de rétro-propagations nécessaires dans le réseau adjoint associé au DRNN* $N = 2, D = 5$ *entre deux itérations successives.*

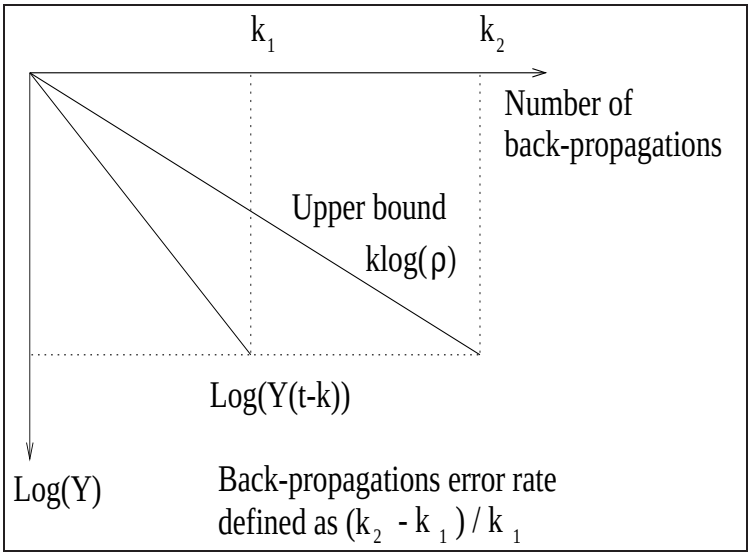


Figure 4.6. *Définition du taux d'erreur de troncature (TLR). Soit* $Y(t-k) = \|\mathbf{y}_{t-k}^{t}\|$, $k_1 = s_1 D$ *le nombre de rétro-propagations suffisant pour que* $Y(t-k_1) < \epsilon$, *où* $\epsilon << 1$. $k_2 = s_2 D$, *où* s_2 *est le plus petit entier vérifiant l'inégalité (4.14). un TLR de valeur n se traduit par une propagation de l'EBBF n fois trop loin dans l'adjoint déplié.*

Résultats - Plusieurs expérimentations ont été menées avec $N = 1, 5$ et 10, $D = 1, 5$ et 10, voire Fig. 4.7. Les courbes d'erreur sont très similaires à Fig. 4.1. Il apparaît que la taux analytique de convergence, ρ, est précis pour de faibles valeurs. La précision se dégrade avec la taille du réseau. Le taux diverge comme pouvait s'y attendre pour ρ proche de 1. Pour autant, des erreurs acceptables sont obtenues pour des valeurs de ρ inférieure à 0.5 : le TLR est inférieure à 5. Pour être complet, la Figure 4.4.2 illustre la variation du TLR pour :

1. une architecture de Williams et Zipser (réseau bouclé standard) avec $N = 10$ neurones cachés inter-connectés, une entrée et une sortie,
2. un réseau (non-bouclé) NARMAX avec $N = 10$ neurones cachés, une entrée exogène, une sortie et $D = 5$ sorties additionnelles réinjectées en entrée. Là encore, l'erreur de troncature diverge pour des valeurs de ρ proche de 1.

Finalement, les expérimentations confirment les résultats analytiques obtenus concernant le taux de décroissance du gradient et l'erreur de troncature. Au paragraphe suivant, des alternatives à la descente du gradient sont énumérées, suivi de quelques cas d'école issue de la littérature de difficulté variable présentant des dépendances à longue portée,

4.5. Quelques alternatives à la descente du gradient

Quelles sont les alternatives à la descente du gradient ? De nombreuses méthodes heuristiques plutôt sophistiquées ont été proposées pour contourner la difficulté d'apprendre des dépendances à long terme. *Sans exhaustivité aucune*, nous en listons ci-dessous quelques-unes extraites de la littérature :

- le **recuit simulé** est un algorithme de recherche global éminemment lent. Appliqués aux réseaux neuronaux, la sélection des nouveaux points dans l'espace des poids est généralement effectuée dans hypercube autour du point courant. Il est également utilisé pour l'initialisation et l'estimation des paramètres [RYN 00].
- la **recherche aléatoire multi-grilles**. Cette méthode est plus simple que le recuit ; une recherche aléatoire est menée dans un hypercube de volume décroissant autour de la meilleure solution trouvée [BEN 94b].
- les **algorithmes pseudo Newton** ont l'avantage de remettre le gradient à l'échelle en rapport à la courbure de la fonction coût, et donc potentiellement de réduire le problèmes de plateaux.
- le **filtrage de Kalman étendu**, (EKF) (voir par ex. [SIN 89]) étendu découle de la linéarisation des équations du réseau. Les calculs de la matrice de gain sont exécutés en ligne mais reposent sur une inversion matricielle

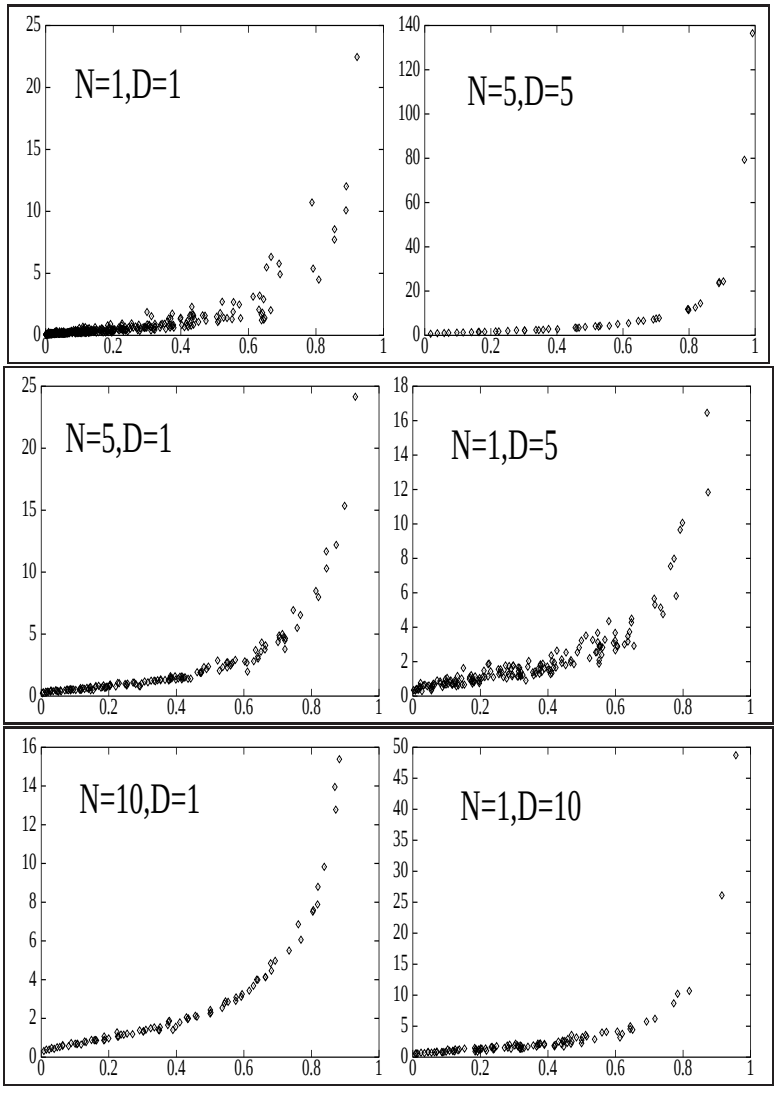


Figure 4.7. *Taux d'erreur de troncature versus le vrai taux de de décroissance du gradient observé pour différentes architectures DRNN génériques.*

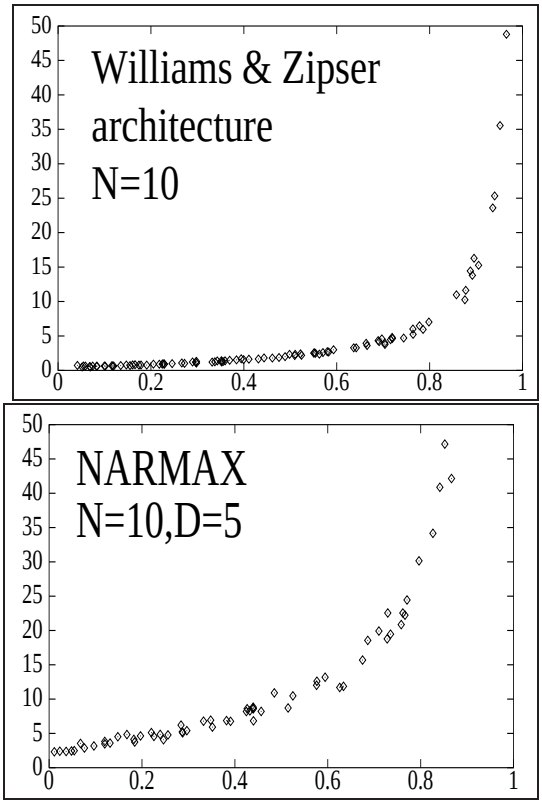


Figure 4.8. *Taux d'erreur de troncature versus le vrai taux de de décroissance du gradient observé. (Haut) Une réseau récurrent standard (de Williams and Zipzer) avec 10 neurones. (Bas) Un réseau NARMAX avec 10 neurones cachés et 5 entrées additionnelles constituées des valeurs retardées de la sortie.*

coûteuse. Le manque de stabilité et l'importance de la linéarisation sont les inconvénients importants de cette méthode.
- l'**algorithme des moindres carrés récursifs étendus** (ERLS) présenté dans [MAN 01] est assez proche de l'EKF. Les deux reposent sur une représentation d'état de Kalman avec équation d'état (état = vecteur de paramètres) du type marche aléatoire ($w(k+1) = w(k) + u(k)$).
- l'**algorithme EM** appliqué au réseaux bouclés requiert un certain nombre d'approximations. A titre d'exemple, Ma et Ji [MA 98] prennent les activations des neurones cachés comme variables cachées du schéma EM. La distribution des variables cachées conditionnellement aux entrés/sorties est supposée gaussienne. Une approximation des *champs moyens* est nécessaire durant l'étape E.

De nombreuses comparaisons - plus ou moins rigoureuses - entre ces algorithmes ont été effectuées sur des critères de performance, de stabilité numérique, de complexité etc. Beaucoup visent à démontrer la primauté d'un algorithme sur les autres sur une classe de problème très spécifique [KRE 01]. Aucun n'apporte une réponse définitive au problème de dépendance à longue portée ; la recherche purement aléatoire [SCH 96] dans l'espace des paramètres donne parfois des meilleurs résultats !

A titre d'information, Mandic et Chambers, dans un ouvrage récent exclusivement consacré aux réseaux récurrents pour la prédiction [MAN 01], proposent de limiter les effets du déclin de l'GEBF en présence de longues dépendances, à l'aide de plusieurs réseaux bouclés en cascade (Pipelined RNN) entraînés par RTRL. Dans le même esprit, ils préconisent également le recuit simulé, l'EKF et l'inclusion de délais dans l'architecture pour palier le problème.

4.6. Quelques problèmes types de dépendance à longue portée

Extraire et mémoriser de l'information sur des intervalles de plusieurs centaines d'unités de temps a été un enjeu majeur de la communauté connexioniste depuis cette dernière décennie. Voici ci-dessous un certain nombre de cas d'école de difficulté variable mettant en jeu des dépendance à longue portée.

Problème des 2 séquences - C'est le premier problème historiquement étudie dans [BEN 94b, BEN 94a, HIH 96, LIN 96]. Deux classes de suites de nombres réels de longueur variable sont générées. Seuls les N premiers termes apportent l'information sur la classe : ils sont tous égaux à 1 ou à -1 selon la classe. Les termes suivant sont tirés au hasard selon une gaussienne centrée de variance 0.2. Le signal d'erreur, $+1$ ou -1 selon le cas, n'est délivré qu'à la fin. Il faut de l'ordre de 6000 exemples à BPTT pour apprendre des suites de longueur

aux alentour de 50 [LIN 96]. Pour les suites plus longues de 50 à 100 unités, seules la recherche aléatoire multi-grilles et l'approche EM tombent en deçà d'une erreur acceptable après la présentation de 3000 exemples. La recherche aléatoire multi-grilles est plus simple que le recuit ; une recherche aléatoire est menée dans un hypercube de volume décroissant autour de la meilleure solution trouvée.

Une recherche purement aléatoire dans l'espace des paramètres d'un petit réseau bouclé fait même encore mieux en performance avec moins d'exemples [SCH 96] alors que la descente du gradient ne parvient pas à converger [BEN 94b].

Problème de parité - Ce problème qui vise à déterminer la parité de suites binaires de longueur variable, entre 25 et 50 seulement, a été étudié par [BEN 94b, BEN 94a, LIN 96]. Il faut de l'ordre de 50 000 présentations d'exemples pour que BPTT converge. 3000 suffisent à la recherche purement aléatoire dans l'espace des paramètres d'un petit réseau bouclé pour résoudre ce problème. Le problème est lié aux minimas dits "plats" de la courbe d'erreur [HOC 97a].

L'apprentissage des automates à états finis - L'apprentissage des automates à états finis a fait couler beaucoup d'encre [GIL 92, ZEN 94, ROD 99, ROD 01, SCH 96]. L'inférence de grammaires déterministes, avec ou sans contexte, a été entreprise avec des réseaux bouclés discrets du second ordre dans [GIL 92, ZEN 94]. Les exemples typiques sont des langages de Tomita (voir [ZEN 94]). Or la recherche purement aléatoire dans l'espace des paramètres d'un petit réseau bouclé permet d'obtenir de meilleurs résultats que BPTT. Typiquement, les réseaux sont entraînés à prédire pas à pas le symbole suivant d'un mot, ou le dernier symbole, x ou y, est uniquement déterminé par le premier symbole du mot, e.g., $(x, a_1, \ldots, a_{n-1}, x)$ ou $(y, a_1, \ldots, a_{n-1}, y)$ où les a_j sont des symboles tirés au sort et jouent le rôle de distracteurs. Les symboles sont codés sur $n+1$ bits. Au delà de $n = 10$, la descente du gradient échoue quelque soit la technique (BPTT, FD, etc) employée.

Apprendre à compter - Il s'agit de mémoriser précisément des nombres réels sur une période de temps indéfini. Les entrées sont constituées de paires de valeurs. La première est tirée au hasard dans $[-1, 1]$. La seconde est un *marqueur* qui vaut -1, 0 ou $+1$. Le but est, par exemple, de fournir en sortie la somme des entrées dont la seconde composante a été marquée par $+1$. Les exemples sont des séquences de longueur comprise entre T et $T + T/10$, avec $T = 100, 500, 1000$ et le signal d'erreur n'est donné qu'à la fin de la séquence. Seul LSTM permet de résoudre ce type de problème avec de l'ordre de 10^6 exemples pour $T = 1000$.

Apprendre à compter sans marqueur temporels - Gers, Schmidhuber et Cummins [GER 00] ont montré qu'une version améliorée de LSTM pouvait résoudre le problème précédent sans l'aide de marqueurs. Le modèle entraîné par une technique de gradient apprend à compter les intervalles de temps discrets séparant l'apparition de symboles identiques dans la séquence d'entrée. Cette tâche est réputée extrêmement ardue pour des réseaux bouclés classiques [ROD 99, ROD 01].

Reconnaître l'ordre temporel - Le but est de classer des séquences de symboles codés sur N bits (où N est la taille de l'alphabet) de longueur variant entre 100 et 110. Seuls les symboles aux instants t_1 et $t_2 > t_1$ et $t_3 > t_2$ décident du symbole final, les autres sont tirés aux hasard dans un alphabet donné. Seul LSTM (présenté ci-dessous) permet de résoudre ce type de problème avec de l'ordre de 10^6 exemples pour 8 classes.

4.7. Vers des modèles d'ordre supérieur

Bien que la descente du gradient soit inadaptée pour capturer des dépendances à long terme, l'effet est réduit dans le DRNN par un facteur 1/D [LIN 96]. Ni les algorithmes ajustant les délais [DUR 99], ni les heuristiques constructives pour ajouter de nouvelles connexions retardées [BON 00] ne peuvent remédier à ce problème. A titre d'information, l'heuristique constructive de [BON 00] en $0(N^4)$ vise à ajouter incrémentalement une connexion retardée de d entre i et j selon

$$(i^\star, j^\star, d^\star) = \arg\max_{(i,j,d)} \left\{ |\sum_{k=1}^{K} \boldsymbol{\delta}_k \mathbf{v}_{k-d}^T| \right\}. \tag{4.22}$$

On retrouve à droite la formule d'ajustement du poids w_{ij}^d selon BPTT si la connexion (ij) existait et était retardée de d.

Une architecture spécifique opérant à plusieurs échelles a été proposée par El Hihi et Bengio (1996) [HIH 96]. Mais la méthode achoppe dès lors qu'il s'agit de mémoriser, par exemple, des valeurs numériques sur de intervalles de temps de plusieurs centaines d'unités de temps.

Pour remédier à cette faiblesse, Hochreiter et Schmidhuber (1995) [HOC 97b, GER 00, SCH 02] ont proposé une architecture prometteuse appelée Long Short Term Memory (LSTM), basée sur des neurones capables de mémoriser des valeurs numériques sur de longs intervalles de temps. Ces *memory cells*, maintiennent constant le flot d'erreur dans le neurone. A la différence des réseaux

standards, le signal d'erreur est piégé dans la cellule et sa valeur ne change que lorsque il est libéré par un porte d'entrée. La *cellule* est régie par les équations

$$\begin{aligned} v_j^{in}(t) &= g^{in}(\sum_i w_{ij}^{in} v_i^{out}(t-1)), \\ v_j^{out}(t) &= g^{out}(\sum_i w_{ij}^{out} v_i^{out}(t-1)), \\ s_j(t) &= s_j(t-1) + v_j^{in}(t) \cdot g(\sum_i w_{ij} v_i^{out}(t-1)), \\ v_j(t) &= v_j^{out}(t) \cdot h(s_j(t)). \end{aligned} \tag{4.23}$$

Des portes v_j^{in}, v_j^{out} sont positionnées en entrée et en sortie du neurone j de manière à autoriser ou refuser l'accès au flot du gradient d'erreur dans la cellule. Une connexion localement récurrente de poids fixé à 1.0 garantit un flot constant dans la cellule $s_j(t)$ (3ème équation ci-dessus). En d'autres termes, c'est une façon de forcer $\rho = 1$ et de rester à la bifurcation entre l'explosion et la décroissance exponentielles.

Une version de BPTT tronquée à une seule rétro-propagation est employée pour entraîner le réseau. Les auteurs jugent inutile de rétro-propager plus en amont l'GEBF puisqu'il decroît exponentiellement vers zéro. Ainsi, il est nullement besoin de mémoriser les vecteurs d'état passés.

Ce faisant, LSTM peut extraire et mémoriser des valeurs numériques sur de intervalles de temps potentiellement infinis. Pour autant, un surcroît de connexions est dédié au maintient d'un l'GEBF localement constant, au détriment de la taille du modèle.

4.8. Conclusion

Nous avons établies des conditions suffisantes pour garantir : 1) la convergence de l'GEBF, 2) l'unicité et la stabilité asymptotique du point fixe du réseau à chaque instant et 3) la convergence asymptotique globale vers un point fixe du réseau en bouclage fermé. Celles-ci s'expriment explicitement en fonction de la matrice de poids et s'appliquent à de nombreux les réseaux bouclés introduits dans la littérature ces dernières années [KRE 01, TSO 94]. Une borne supérieure sur le nombre de rétro-propagations dans le temps est a été établie pour limiter l'erreur de (BPTT). Des simulations ont confirmé ces résultats théoriques et montré que l'erreur de troncature est fiable dès lors que le coefficient de décroissance du gradient est petit.

Ainsi, extraire et mémoriser de l'information sur des intervalles de plusieurs centaines d'unités de temps est une tâche ardue avec des réseaux bouclés standards entraînes par une technique basée sur le gradient d'erreur. L'adjonction de *neurones d'ordre supérieur* semble porter remède à ce problème épineux, sans nuire à la simplicité de BPTT. Nous verrons au chapitre 6 consacré à la multirésolutions qu'il est possible de contourner le difficile problème de l'apprentissage des dépendances à longue portée présentent dans le signal d'entrée à l'aide d'un ban de filtres *ad hoc*.

Chapitre 5

PREDICTIONS ENVIRONNEMENTALES

5.1. Introduction

Je dresse dans ce chapitre un bref panorama des applications des réseaux discrets bouclés à délais aux sciences environnementales, dans une acception très large, menées dans le cadre de plusieurs contrats de recherche s'échelonnant de 1997 à 2001, principalement avec le *European Southern Observatory* (ESO) à Garching en Allemagne, et le *Marine Environment Unit* du *Joint Research Center* (JRC) situé à ISPRA en Italie.

Après quelques rappels sur les propriétés des réseaux de neurones, la modélisation du type boîte noire et le mode opératoire, des applications des réseaux bouclés à délais sur des données synthétiques engendrées par des processus chaotiques déterministes sont présentées. Ces processus bien connus caractérisent certains processus physiques tels que les systèmes physiologiques, les fluctuations d'un laser infrarouge, ou un fluide soumis à des échanges thermiques. L'objectif est de reconstruire, en itérant les réseaux en bouclage fermé, les attracteurs chaotiques à géométrie fractale à partir d'un vecteur d'état tronqué.

Ensuite, nous présentons une étude réalisée pour le JRC en 1999 et 2000, visant à prédire la température à la surface de la mer (SST) sous forme de cartes 2D, dans une zone maritime où l'on observe des mouvement ascendants d'eau froide, que l'on désigne par phénomène d'*upwelling*, e.g. au large de la Namibie. Les lignes de contour de la SST peuvent être construites à moindre coût, sans l'aide directe des modèles numériques (de circulation océanique) lourds à mettre en oeuvre, grâce à la collaboration de plusieurs RN en charge

de la prédiction de la SST en plusieurs points d'une grille. Les cartes 2D sont alors reconstruites par interpolation des SST fournies par les modèles.

Enfin, nous présentons les prédictions des fluctuations du *seeing* astronomique à partir de mesures météorologiques prises au sol, réalisées pour ESO [AUS 00d] en 2001, dans le cadre d'un contrat de recherche de 16 mois. Le *seeing* est une mesure de la diffraction des ondes lumineuses dûe aux perturbations atmosphériques. Quand les astronomes finissent une observation, le calibrage des instruments est effectué pour l'observation suivante. La prédiction d'une mesure de variabilité du *seing* - de 5 à 60 minutes en avance - permettrait l'ordonnancement optimale des tâches de calibration et d'observation.

Ce paragraphe applicatif s'achève sur l'esquisse d'un travail mené avec David Hill (LIMOS), dans le cadre d'un projet LIFE "Control of the spread of the *Caulerpa Taxifolia* in the Mediterranean" (programme DG XI) consacré à la prévision de la surface contaminée par la caulerpe après plusieurs années dans la bassin méditerranéen.

Le principe d'entraîner un réseau de neurones à l'aide d'un simulateur stochastique, mis en oeuvre à plusieurs reprises au cours de mes applications, est parfois qualifié de *méta-modélisation* [KIL 94]. Son principal intérêt est de permettre un gain en temps de calcul considérable puisque l'évolution du système stochastique complexe (le modèle), est approximée par un processus déterministe plus simple (le méta-modèle). Dans tous les cas, la modélisation est du type boîte noire et l'apprentissage est réalisé à l'aide d'un historique de valeurs mesurées ; la physique du phénomène - souvent méconnue - n'est pas prise en compte.

5.2. Régression linéaire/non-linéaire : quelques rappels

Les réseaux connexionistes occupent une place centrale parmi les outils statistiques de prévision. Ils sont aussi le thème d'un courant de recherche vivant et fécond, à la confluence de l'analyse des séries temporelles et de la théorie des systèmes dynamiques. Nous présentons ici quelques rappels sur la régression linéaire et non-linéaire, avant de rappeler quelques propriétés phares importantes des réseaux de neurones dans le cadre de l'approximation de fonction.

Limites des modèles linéaires - Les débuts de l'analyse "moderne" des séries temporelles remonte à 1927 lorsque Yule inventa la technique autorégressive dans l'intention de prédire le nombre annuel de taches sombres du soleil, les sunspots. Le paradigme des modèles linéaires alimentés par un bruit blanc additif a prévalu durant la moitié XXième siècle. Toutefois, il s'avère inadéquat pour certaines suites déterministes parfois très simples. Par exemple,

la suite obtenue par itération de la fonction logistique possède un spectre de puissance impossible à modéliser avec un modèle linéaire. L'évolution "libre" des processus ARMA ne permet pas un comportement périodique stable indépendant des valeurs initiales. Le signal généré est caractérisé par un nombre *fini* de fréquences. Par ailleurs, l'évaluation des paramètres des modèles AR, MA, et ARMA reposent uniquement sur la fonction d'autocorrelation de la suite observée. Or l'autocorrelation de X_t avec X_{t-j} ($j \in \mathbf{Z}$) ne présente qu'un aspect de la distribution des couple (X_t, X_{t-j}). D'autres aspects vitaux peuvent être omis. Ainsi, la classe des modèles linéaires se prête à la modélisation des suites temporelles pour lesquelles seule la fonction d'autocorrelation (ou indifféremment le spectre de puissance) caractérise la suite.

Exemples - Prenons deux exemples classiques afin de mettre à jour les limites des modèles linéaires. Considérons la suite générée par la fonction logistique $x_{t+1} = rx_t(1 - x_t)$. Popularisée dans le cadre des modèles écologiques idéalisés, ce modèle simple possède une dynamique complexe [OTT 93]. Cette équation décrit par ailleurs un certain nombres de réactions chimiques et de flots hydrodynamiques. Le paramètre r agit de manière qualitative sur le comportement de la suite, variant du point fixe au chaos déterministe. Par exemple pour $r = 4$, chaque itération détruit un bit d'information ; du fait de la non-inversibilité de la fonction logistique et la symétrie de la *densité naturelle invariante*, les prédécesseurs de x_t au nombre de deux sont équiprobables. En d'autres termes, la connaissance de x_t à ϵ près ne permet la connaissance de son successeur, x_{t+1}, qu'avec une précision de 2ϵ. La croissance exponentielle de l'incertitude est une caractéristique du chaos déterministe.

Le deuxième exemple est tout aussi simple. Considérons la suite générée par $x_{t+1} = 2x_t (\mathrm{mod}\ 1)$. L'action du modulo se comprend aisément en considérant x_t sous forme binaire (i.e. $x_t = 0d_1d_2\ldots = (d_1.2^{-1}) + (d_2.2^{-2}) + \ldots$) ; chaque itération décale la décomposition binaire d'un rang vers la gauche ($d_i \leftarrow d_{i+1}$). Il résulte que le bit le plus significatif d_1 est éliminé, cédant la place à d_2. L'implémentation physique de la fonction est la boule de billard évoluant sans dissipation d'énergie ; x_t peut être vu comme les positions successives sur une droite quelconque dans le plan du billard que la boule croise après chaque rebond.

L'évolution de ces deux systèmes unidimensionnels déterministes est entierement déterminée par la condition initiale x_0 comme l'est le processus AR piégé sur une orbite périodique stable. Cependant, ces systèmes élémentaires (une parabole et une droite) génèrent chacun une suite de valeurs possédant une *densité* spectrale *inhérente au modèle lui même*, par opposition à un spectre discrétisé. Dans le contexte des modèles ARMA, c'est le bruit externe en entrée du système qui pourvoit le modèle d'une densité spectrale, alors que dans le cas présent, cette composante est inhérente à ces systèmes. Il faut conclure de

cela que la non-linéarité est essentielle dans les systèmes déterministes pour produire un comportement d'apparence aléatoire.

5.2.1. *Quelques propriétés du MLP*

Les réseaux discrets bouclés à délais héritent des certaines propriétés des MLP, en particulier les propriétés d'approximation et de robustesse, que nous énumérons rapidement ci-dessous. On trouvera toutes les références dans, par exemple, la thèse de M. Mangeas [MAN 95].

Propriétés d'approximation - Le problème de l'approximation de fonctions par des MLP à une couche cachée a suscité une abondante littérature. Le théorème de Hornik stipule par exemple que toute fonction continue à support compact de $\mathbb{R}^n$ peut être approchée avec une précision arbitraire (au sens de la norme de $\mathbb{R}^n$) par un MLP à une couche cachée, munis de fonctions sigmoïdes, et une sortie linéaire. Ce théorème ne fournit pas le vitesse de convergence toutefois.

Vitesse de convergence - Sous certains conditions portant sur le module de la transformée de Fourier de la fonction, Barron a établi que l'erreur d'approximation de la fonction par un MLP est bornée par un terme en $O(\frac{1}{C})$, où C est le nombre de neurones cachés. Il est cependant difficile de relier les propriétés classiques de la fonction (e.g. continuité, dérivabilité) avec ce critère de complexité dans l'espace complexe.

Correspondance avec les modèles ARMA - Enfin, notons qu'il existe une correspondance formelle entre les modèles ARMA et neuronaux, pour la prévision à un pas, en raison de la linéarité au voisinage de zéro des fonctions sigmoïde. La sortie d'un processus ARMA peut être approchée avec une précision arbitraire par un MLP. On est donc en droit de penser que si le modèle linéaire est plus performant, l'apprentissage amènera le MLP à ne travailler que dans sa partie linéaire.

Robustesse - Outre les propriétés d'approximation, n'oublions pas d'évoquer les particularités de robustesse de ce type de modèle paramétrique, à savoir la robustesse à la détérioration (e.g. supprimer une connexion, un neurone) en raison de la redondance de l'information dans le réseau, mais également la résistance aux variables d'entrée aberrantes du fait de la saturation des fonctions d'activation, et ce à la différence des modèles linéaires pour lesquels la sortie est linéairement proportionnelle à l'entrée.

Estimation des paramètres - Notons enfin que l'estimateur des moindres carrés d'un MLP à une couche est *fortement consistant* [MAN 95]. En d'autres

termes, si l'on suppose que les données ont été engendrées par un MLP, muni d'un vecteur de paramètre inconnu, la minimisation du risque empirique garantit la convergence *presque sûre* des paramètres estimés vers leur vraie valeur. On se place pour cela dans un sous-ensemble quotient des paramètres assurant l'*identifiabilité* du MLP [MAN 95].

Identification du modèle - L'identification du modèle, qui revient à sélectionner sa structure au sein d'une famille, est une procédure généralement en aval de l'estimation des paramètres. Nombreux sont les cas où la minimisation du risque empirique ne garantit pas la minimisation du risque théorique (e.g. les polynômes de degré quelconque, les fonctions sinusoïdales). C'est pourquoi Vapnik a étudié la distance du risque empirique au risque théorique, et ce indépendamment de la distribution conjointe $P(\mathbf{x}, y)$, et en a déduit le principe de minimisation du risque structurel (SRM). Or les bornes établies par Vapnik ne sont valables que pour des variables i.i.d., hypothèse fausse dans le cadre des séries temporelles et des modèles auto-régressifs, pour lesquels les variables explicatives sont le passé du processus. De plus, seules des bornes supérieurs de la VC dimension des MLP existent. Aussi, pour déterminer la bonne architecture, certains statisticiens substituent un critère d'information aux bornes de Vapnik, sous la forme d'une fonction objectif pénalisée par un terme additif du type $P \ln n/n$, où P est le nombre de paramètres [MAN 95]. Il suffit alors à partir d'un modèle dominant d'affiner le modèle en éliminant les connexions superflues ; on parle alors de *régularisation structurelle* [GOU 97] par opposition à une *régularisation formelle* du type *weight decay*. Outre sa consistance, l'estimateur des moindres carrés pour les MLP vérifie les propriétés de normalité asymptotique et une loi du logarithme itéré (sous l'hypothèse que le bruit a un moment d'ordre suffisant). Il est donc possible d'éliminer les connexions superflues par un test de Student.

Mais en pratique, la mise en application du principe SRM où du contraste pénalisé pour identifier le modèle est laborieuse. Dans cette étude, on lui préfère une approche *moins rigoureuse*, certes, fondée sur la régularisation (au sens de Tikhonov) d'un modèle surparamétré de façon à réduire la *variance* du modèle (au sens de la décomposition biais/variance).

Enfin, mentionnons les méthodes empiriques de sélection de modèle, telles que celles fondées sur la validation croisée et ses variantes, ainsi que les nouvelles méthodes de pénalisation empiriques comme les méthodes basées sur les métriques de Schuurmans (voir [BEN 03]).

5.2.2. *La modélisation dynamique "boîte noire"*

Nous nous sommes attachés à souligner les traits saillants des modèles linéaires, dans un langage àmi-chemin entre le formalisme pur et la compréhension intuitive. Le moment est venu de mettre l'accent sur les principes de la modélisation "boîte noire', ainsi que sur les problèmes qui demeurent encore ouverts à ce jour.

Revenons un instant au problème originel de l'analyse statistique des séries temporelles : prédire, c'est trouver une relation fonctionnelle, probablement non-linéaire, entre les valeurs passées et la valeur courante de telle sorte que l'erreur d'estimation se réduise à un bruit blanc. Un modèle causal idéal, que symbolise la fonction $h()$, pour la séquence $\{X_t\}$ s'écrit

$$X_t - h(X_{t-1}, X_{t-2}, \ldots) = \epsilon_t \tag{5.1}$$

où ϵ_t est un processus blanc centré. Si la séquence des résidus $\{\epsilon_t\}$ ne formait pas un bruit blanc, il y aurait une structure dans la relation entre X_t et son passé dont le modèle ne tiendrait pas compte. Posé en ces termes, le problème est trop général pour être résolu car pour un nombre fini de données, il est impossible de spécifier $h()$ parmi toutes les fonctions non-linéaires possibles. Considérons l'extension non-linéaire des modèles ARMA,

$$X_t = h(X_{t-1}, \ldots, X_{t-p}; \epsilon_{t-1}, \ldots, \epsilon_{t-q}) + \epsilon_t \tag{5.2}$$

La nouveauté par rapport à (5.1) réside dans le fait que x_t ne dépend pas seulement des perturbations aléatoires au même instant, représenté par le bruit blanc ϵ_t, mais garde une mémoire des perturbations antérieures jusqu'à $t - q$. On compense ainsi le manque d'observations passées par une mémoire des perturbations. Si le modèle est inversible, i.e. $\epsilon_t = f(X_t, \ldots)$, alors le prédicteur optimal est

$$\hat{X}_t = h(X_{t-1}, \ldots, X_{t-p}; \epsilon_{t-1}, \ldots, \epsilon_{t-q}) \tag{5.3}$$

Il est donc envisageable d'approximer $h()$ par un modèle neuronal, dans la mesure où l'entrée est maintenant de taille finie. Or le problème qui surgit maintenant a trait à la nature même des entrées : on ne connaˆpas le bruit, ϵ_t. Il nous faut donc se restreindre à une approximation $\mathcal{N}$ du prédicteur optimal $h()$ donnée par

$$\mathcal{N}^\star = h(X_{t-1}, \ldots, X_{t-p}; e_{t-1}, \ldots, e_{t-q}) \tag{5.4}$$

où $e_j = X_j - \hat{X}_j$. Une représentation équivalente est

$$\hat{x}_t = \mathcal{N}(x_{t-1}, x_{t-2}, \ldots, x_{t-p}; \hat{x}_{t-1}, \hat{x}_{t-2}, \ldots, \hat{x}_{t-q}) \tag{5.5}$$

Une représentation d'état est plus parcimonieuse et plus générale qu'un représentation entrée-sortie qui n'utilise que les entrées et les sorties retardées du système pour estimer la sortie du système. De la même manière que le modèle neuronal NARMA étend le modèle ARMA au domaine non-linéaire, le réseau bouclés généralise la représentation d'état linéaire de Kalman. Au lieu de disposer la mémoire spatialement en entrée de la "boite noire", le vecteur d'état confère une mémoire interne au réseau. Tout réseau bouclé peut sous une forme d'état minimale, dite forme "canonique" (voir [DRE 02]). La transposition du DRNN (sans cycle de délai nul) vers une forme canonique au sens de Nerrand et al. est immédiate. Je lui préfère la représentation suivante où les délais sont représentés explicitement,

$$\begin{aligned} \mathbf{z}_t &= \Phi(\mathbf{z}_{t-1}, \ldots, \mathbf{z}_{t-D}; \mathbf{x}_{t-D}^t; \hat{\mathbf{x}}_{t-D}^t) \\ \hat{x}_{t+1} &= \psi(\mathbf{z}_t, \ldots, \mathbf{z}_{t-D}; \mathbf{x}_{t-D}^t; \hat{\mathbf{x}}_{t-D}^t) \end{aligned} \tag{5.6}$$

où $\mathbf{x}_{t-D}^t = \{x_{t-D}, \ldots, x_t\}$ et $\hat{\mathbf{x}}_{t-D}^t = \{\hat{x}_{t-D}, \ldots, \hat{x}_t\}$. En concaténant dans le vecteur $\tilde{\mathbf{z}}_t$, l'activation de tous les neurones aux instants $t, t-1, \ldots, t-D$, soit $N(D+1)$ valeurs, on retrouve la forme canonique de Nerrand et al.

5.2.3. *Quelques problèmes ouverts*

Enumérons à présent un certain nombre de problèmes ouverts que se pose concrêtement l'expérimentateur dans l'optique de construire un modèle prédictif neuronal.

- Est-il possible de reconstituer le vecteur d'état du système au moyen des observations passées ? C'est la généralisation aux système dynamiques non-linéaires de la notion d' *observabilité* des systèmes. Quand bien même $\psi()$ et $\Phi()$ seraient linéaires, l'observabilité n'est pas garantie.
- Il n'existe pas de *critère statistique de linéarité* fiable à l'heure actuelle pour évaluer la linéarité d'un processus,
- Sauf avis contraire, les suites sont supposées être stationnaires et ergodiques (au moins au second ordre). En d'autres termes les propriétés statistiques du processus sous-jacent peuvent être obtenu à partir de l'observation d'une trajectoire unique sur $]-\infty, \infty[$. Dans le cadre non-linéaire, il est difficile de mener une analyse de la stationnarité qui peut prendre de multiples formes (e.g. stationnarité par morceaux).

- Dans le cadre linéaire, la meilleure prévision à plusieurs pas est obtenue en substituant la anciennes prévisions aux valeurs manquantes. Cette propriété n'est plus vraie dans le cadre non-linéaire. Il n'existe pas de résultats d'optimalité de la méthodologie de prévision à plusieurs pas en avant.
- Enfin, il est légitime de se demander si les modèles stochastiques constituent la meilleure alternative pour modéliser des processus (chaotiques ou non) *déterministes*?

5.3. Mode opératoire

Avant de nous tourner vers les expérimentions à proprement parler, évoquons les techniques empiriques classiques - mais assez arbitraires - qui nous ont permis d'accélérer l'apprentissage et réduire l'effet néfaste des minima locaux.

Les modifications des poids des réseaux bouclés ont été réalisées dans cette étude par l'algorithme de descente du **gradient stochastique**. L'apprentissage opère en mode non-dirigé lorsqu'il s'agit de réaliser des prévisions à plusieurs pas en bouclage fermé, et semi-dirigé pour réaliser des prédicteur à un pas. Le gradient d'erreur est calculé selon la version tronquée de BPTT évoquée au Chapitre 3. Le pas d'apprentissage η est adapté au fil de l'apprentissage en vérifiant si l'adaptation de la matrice de poids décroît effectivement la fonction d'erreur. A l'opposé, si plusieurs déplacement réduisent l'erreur consécutivement, il s'agit peut être d'une vallée en pente auquel cas il est judicieux d'accroître la valeur du pas d'apprentissage. Il est préférable, au regard de la littérature [HER 91], d'acccroître la valeur de η par un coefficient constant et de le décroître géométriquement pour permettre une chute rapide lorsque cela s'avère nécessaire. Cela conduit au schéma suivant,

$$\eta = \begin{cases} +\alpha & \text{si } \Delta E \leq 0 \\ -\beta\eta & \text{si } \Delta E > 0 \end{cases} \tag{5.7}$$

Dans la pratique, α est fixée à 10^{-3} pour les premières 100 époques et ensuite réduit à 5×10^{-4} ; $\beta = 0.9$ reste constant. Une moyenne mobile (**momentum**) de coefficient 0.9 dans l'adaptation des poids est employée pour améliorer la robustesse de la descente gradient et augmenter la vitesse de convergence. En outre, pour prévenir le risque potentiel d'*overfitting*, la technique **weight elimination** [BIS 95] est employée. Inspîrée du *weight decay*, la méthode consiste à élaguer les connexions jugées inutiles au cours de l'apprentissage [HER 91, WEI 90]. Pour cela, chaque poids est systématiquement diminué par l'entremise d'un coefficient multiplicatif, $\epsilon < 1$,

$$w_{ij}^{new} = (1 - \epsilon_{ij}) w_{ij}^{old} \tag{5.8}$$

si bien que les connexions pénalisées s'effacent devant les autres lorsqu'elles ne sont plus renforcées. Pour que les w_{ij} grands ne décroissent pas plus rapidement que les petits, ϵ_{ij} est ajusté selon

$$\epsilon_{ij} = \frac{\gamma \eta}{(1 + w_{ij}^2)^2} \tag{5.9}$$

où γ est varié au fil de l'apprentissage. Habituellement, nous commençons avec $\gamma = 10^{-3}$ pour garantir l'élimination des connexions superflues et, lorsque l'erreur stagne, sa valeur est réduite à 10^{-4} pour permettre une décroissance à plus long terme. Ces valeurs ont été choisies heuristiquement et appliquées à tous les réseaux de neurones pour garantir un bon fonctionnement de la procédure d'apprentissage.

Enfin, la progression de l'apprentissage est jaugée à l'aune de l'erreur quadratique normalisée (NMSE), ou *normalized mean squared error* :

$$\text{NMSE} = \frac{1}{\sigma^2 T} \sum_{k=1}^{T} (x(k) - \hat{x}(k))^2 \tag{5.10}$$

$x(k)$ désigne la valeur observée de la suite, et $\hat{x}(k)$, la prédiction fournie en sortie du réseau de neurones ; σ^2 est la variance de la suite et T le nombre d'exemples dans la base de test. Remarquons que NMSE = 1 correspond à la prédiction de la moyenne non conditionnée.

5.4. La modélisation des processus chaotiques

L'observation et l'analyse des processus chaotiques remontent au début du siècle lorsque Henri Pointcarré s'intéressa aux orbites "complexes" des astres sous l'effet mutuel de la gravitation. Toutefois, l'attrait pour le chaos est très récent et il y a à cela deux raisons majeures. D'une part les ouvrages mathématiques sont souvent abscons [OTT 93], voire ésotériques, de sorte que les chercheurs dans d'autres domaines n'ont pas facilement accès aux fruits de la recherche mathématique. D'autre part, les résultats concrets et applicables n'ont vu le jour que très récemment. Il existe, en effet, un lien étroit entre la fenêtre temporelle et la dynamique du processus piégé dans un attracteur. Ce lien a été mis en évidence par Ruelle, Packard et Takens dans les années 1980 (voir références dans [AUS 98c]). En effet, il s'avère qu'un vecteur de délai de

longueur suffisante permet de recouvrer la structure géométrique du système étudié ; en d'autres termes, il est possible d'anticiper exactement l'évolution du système grâce à un vecteur de délais

$$[x(t-\tau), x(t-2\tau), \ldots, x(t-M\tau)] \tag{5.11}$$

Le résultat établi par Takens stipule que pour un grand nombre de systèmes déterministes exempts de bruit, il existe un *difféomorphisme* (i.e. fonction bijective et différentiable) entre $\mathbf{x}(t)$ et le vecteur ci-dessus, dès lors que M est deux fois plus grand que la *dimension fractale*[1] de l'attracteur. On pourra consulter l'ouvrage de Ott [OTT 93] pour avoir plus de détails sur ce sujet.

Dans les exemples qui suivent, des modèles DRNN aux dimensions variables sont entraînés sur des suites temporelles générées par des processus chaotiques. Les résultats des prédictions à court et moyen terme sont exhibés puis analysés. Les attracteurs chaotiques à géométrie fractale sont reconstruits en itérant les réseaux en bouclage fermé. Les conditions suffisantes de convergence vers un *point fixe* du réseau opérant en bouclage fermé ne sont bien entendu pas être satisfaites. Nonobstant le caractère assez "classique" de la reconstruction d'attracteurs chaotiques, elle suscite encore aujourd'hui beaucoup d'intérêt comme en témoigne les travaux similaires très récents [BAK 00, ISH 01].

Avant de plonger dans le grand bain de la prédiction des processus chaotiques, il est bon de rappeler brièvement les traits caractéristiques des processus chaotiques déterministes.

Le chaos déterministe - L'approche traditionnelle pour anticiper l'évolution d'un système dynamique consiste à établir un modèle mathématique *ad hoc* et à l'intégrer dans l'avenir. Or le nombre de degrés de liberté parfois exorbitant d'un système non-linéaire (e.g. un écoulement turbulent, la météorologie, les grandeurs économiques, etc.) constitue un écueil rédhibitoire à la résolution des équations. Cependant, l'étude des systèmes dynamiques révèle que la dissipation (e.g. la viscosité) réduit le nombre de degrés de liberté effectifs à un nombre restreint. Lorsque le cas se présente, l'évolution du système après un certain temps est piégé dans un sous-espace appelé un *attracteur*. L'attracteur est souvent un objet *fractal* dont la dimension n'est pas nécessairement entière [OTT 93].

L'attribut des processus chaotiques, qui au demeurant définit le chaos, est la dépendance exponentielle aux conditions initiales. Restreignons cette discussions à un système unidimensionnel discret. L'écart infinitésimal entre deux

[1] La dimension fractale est une mesure du nombre effectif de degrés de liberté du système dynamique [OTT 93].

conditions initiales x_0 et $x_0 + \epsilon$ typiquement diverge exponentiellement lorsque le système est itéré en avant. Aussi, pour de grande valeur de n, on a

$$dx_n \sim \exp(hn) dx_0 \tag{5.12}$$

où dx_n désigne l'écart après n itérations. Un indicateur commode de la sensibilité des perturbations des orbites infiniment proches est l'*exposant de Lyapounov*, h, défini par

$$h = \lim_{n\to\infty} \frac{1}{n} \log \mid \frac{dx_n}{dx_0} \mid \tag{5.13}$$

Si $x_{n+1} = M(x_n)$, h s'écrit

$$h = \lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} \log \mid M'(x_k) \mid \tag{5.14}$$

Compte tenu de (5.14), un exposant de Lyapounov positif $h > 0$ indique le chaos.

Aussi, une perturbation de l'état du système, quel qu'en soit l'origine (e.g. erreur de quantification, de modélisation, etc.) provoque la divergence exponentielle de l'état du système par rapport à sa trajectoire sans perturbations. Ce résultat pose une limite fondamentale à la qualité des prédictions dans un futur éloigné.

En revanche, la prédiction à court terme bénéficie de la réduction dimensionnelle. Mais comment trouver la combinaison des degrés de liberté effectifs qui déterminent l'évolution du système dans l'attracteur ? En fait il ne s'agit pas tant de savoir quelle variables choisir mais plutôt combien. En effet, le théorème de Takens affirme qu'un ensemble adéquat d'observations passées permet la reconstruction univoque de l'état du système piégé dans l'attracteur.

5.4.1. *La suite de Mackey-Glass*

Dans ce premier exemple, considérons l'équation différentielle à délais de Mackey-Glass [MAC 77]

$$\frac{dx(t)}{dt} = -0.1x(t) + \frac{0.2x(t-\Delta)}{1 + x(t-\Delta)^{10}} \tag{5.15}$$

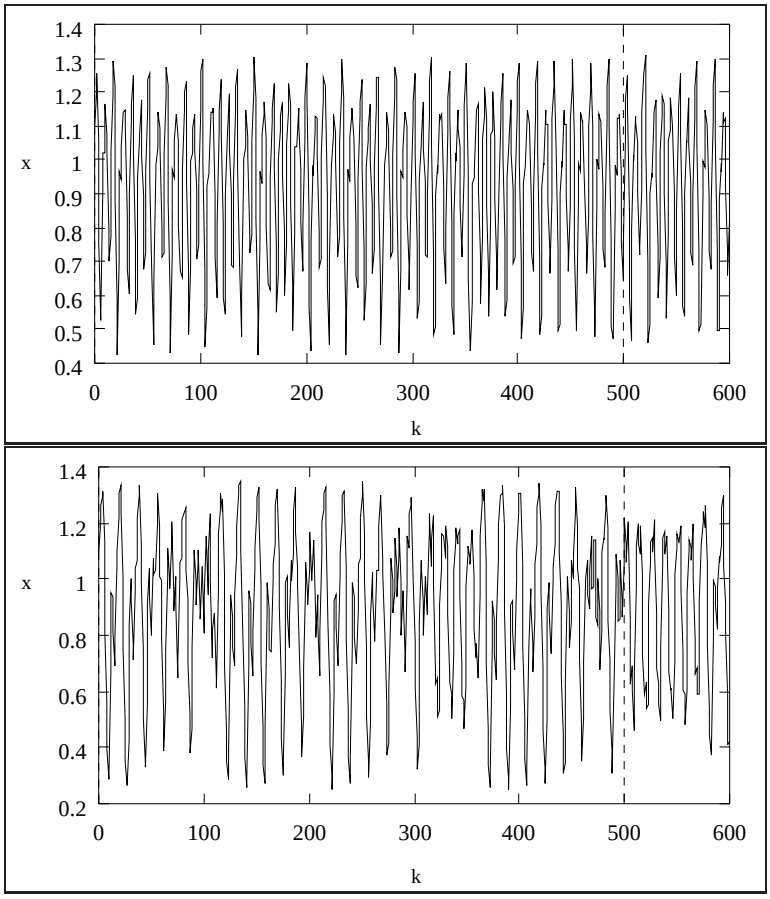


Figure 5.1. *Suite de Mackey Glass avec* $\Delta = 17$ *et* $\Delta = 30$.

Ce modèle est sensé caractériser certains systèmes physiologiques. Sa dynamique, paramétrée par Δ, revêt des formes très diverses variant du point fixe au chaos déterministe en passant par un comportement périodique. L'espace des phases est infini en raison du délai $x(t-\Delta)$. Toutefois, deux valeurs de Δ se distinguent, $\Delta = 17$ et $\Delta = 30$, conduisant à un comportement chaotique avec des attracteurs de dimension finie qualifiés d'*étranges* [OTT 93] de dimension 2.1 et 3.5 respectivement. Pour $\Delta = 17$, la densité spectrale de puissance de $x(t)$ présente de nombreux pics témoignant de la quasi-périodicité de la suite obtenue.

La technique de Runge-Kutta du 4^{ieme} ordre à été employée pour simuler

	NMSE	
modèle	$\Delta = 17$	$\Delta = 30$
Modèle linéaire	-0.57	-0.49
Modèle polynomial	-1.95	-1.40
Modèle rationnel	-1.14	-1.33
loc(1)	-1.48	-1.24
loc(2)	-1.89	-0.42
Fonctions radiales	-1.97	-1.60
RN non-bouclé statique	-2.00	-1.50
RN FIR non-bouclé	-2.31	-1.79
DRNN	-2.33	-2.12

Tableau 5.1. *Erreur normalisée logarithmique des prédictions de la suite de Mackey-Glass à un pas en avance selon différentes approches : linéaire, polynomiale, rationnelle, reconstruction locale, fonctions radiales et réseaux de neurones (voir [AUS 98a]).*

le système. Deux bases d'exemples ont été générées pour $\Delta = 17$ et $\Delta = 30$ avec les conditions initiales $x(t) = 0.9$ pour $0 \leq t \leq \Delta$ puis échantillonnées à une fréquence $f = 6$ (Fig. 5.1). Des études similaires (voir références dans [AUS 95a]) nous ont guidé dans le choix de ces paramètres. Un modèle DRNN de configuration $1 \times 7 \times 1$ avec des filtres d'ordre $4 : 2 : 0$ (2 représente l'ordre des filtres employés pour les connexions bouclées des 7 neurones cachés) a été entraîné sur les premiers 500 points de la suite. La sélection de ces paramètres a été faite sur la base du nombre de degrés de liberté pour faciliter la comparaison avec un réseau FIR de configuration $1 \times 15 \times 1$ avec des filtres d'ordre $8 : 2$ [WAN 93]. Le DRNN compte 197 poids ajustables et le réseau FIR 196. La simplicité a prévalu dans le choix de la topologie. En dépit du nombre équivalent de degrés de liberté, les deux réseaux se distinguent par une fenêtre temporelle de taille distincte. Le FIR dispose des 9 dernières valeurs de la suite pour fournir une estimation en sortie tandis que le DRNN se contente de 5 valeurs. L'ordre des connexions retardées en entrée du DRNN est réduit au profit du nombre de connexions récurrentes internes.

Titre illustratif, les résultats de la prédiction à un pas en avance sont illustrés Table 5.1 pour différents modèles [AUS 98a, WAN 93]. Observons les prédictions itérées. Le DRNN est comparé au réseau FIR au-delà de la 500^{ieme} itération. Comme le révèle Fig. 5.2, la prédiction itérée sur les 20 points suivant est remarquablement précise puis se dégrade au-delà.

Une caractéristique fondamentale du chaos est l'impossibilité de délivrer des prédictions précises dans le futur éloigné, indépendamment du modèle. En fait, les trajectoires infiniment proches divergent exponentiellement en raison

du bruit de quantifications. En moyenne, les prédictions itérées doivent donc diverger exponentiellement de la trajectoire observée. Pour illustrer cela, des prédictions itérées sont fournies au départ de 10 points équidistants de la base de prédiction. L'erreur normalisée *moyenne* est tracée Fig. 5.3 en échelle logarithmique : la linéarité est patente, corroborant l'analyse théorique. On remarquera une pente plus abrupte pour le DRNN par rapport au modèle FIR ; les deux courbes se croisent à l'itération 15.

Fig. 5.3 illustre le tracé de la NMSE moyennée en fonction des itérations dans le futur. On remarque une pente linéaire caractéristique des systèmes chaotiques. Il est aussi à noter que passé un certain délai, la NMSE dépasse 0, indiquant que la prédiction est pire que la prédiction constante de la moyenne de la suite. Cela s'explique par le fait que après un certain nombre d'itérations, la prédiction $\hat{x}$ devient statistiquement *indépendante* de la suite observée x. L'erreur mesurée est le double de la variance de la suite et la prédiction linéaire devient "meilleure" au sens statistique, même si la modèle linéaire est incapable de reproduire la dynamique de la suite. Cela s'explique simplement par

$$\begin{aligned} \mathbf{E}[(x-\hat{x})^2] &= \mathbf{E}[(x-m_x)^2] + \mathbf{E}[(m_x-\hat{x})^2] + 2\mathbf{E}[(x-m_x)(m_x-\hat{x})] \\ &= 2\sigma_x^2 \end{aligned} \tag{5.16}$$

Compte tenu de la taille différente des fenêtres temporelles et sous l'hypothèse du même nombre de connexions dans chaque réseaux, il apparaît que le DRNN délivre de meilleures prédictions à court terme que le réseau FIR, aux dépends de la précision à long terme. Ce résultat s'interprète intuitivement : dès la 6^{ieme} itération dans le future, le DRNN opère en bouclage fermé. Ainsi toutes les 6 itérations le vecteur d'entrée est entièrement renouvelé par les anciennes sorties du réseau, alors que cette période est de 10 pour le FIR. Il n'est pas étonnant que la qualité des prédictions itérées se dégrade plus rapidement avec le DRNN puisque les erreurs se répercutent en cascade à une fréquence plus élevée. Mais il ne tient qu'à moi d'avoir choisi d'utiliser un fenêtre temporelle plus petite que celle du réseau FIR non-bouclé, alors que ce choix est subordonné à la structure temporelle de la série pour les réseaux statiques.

5.4.2. *La suite de Hénon*

Dans cet exemple, considérons la suite obtenu en itérant le système bidimensionnel

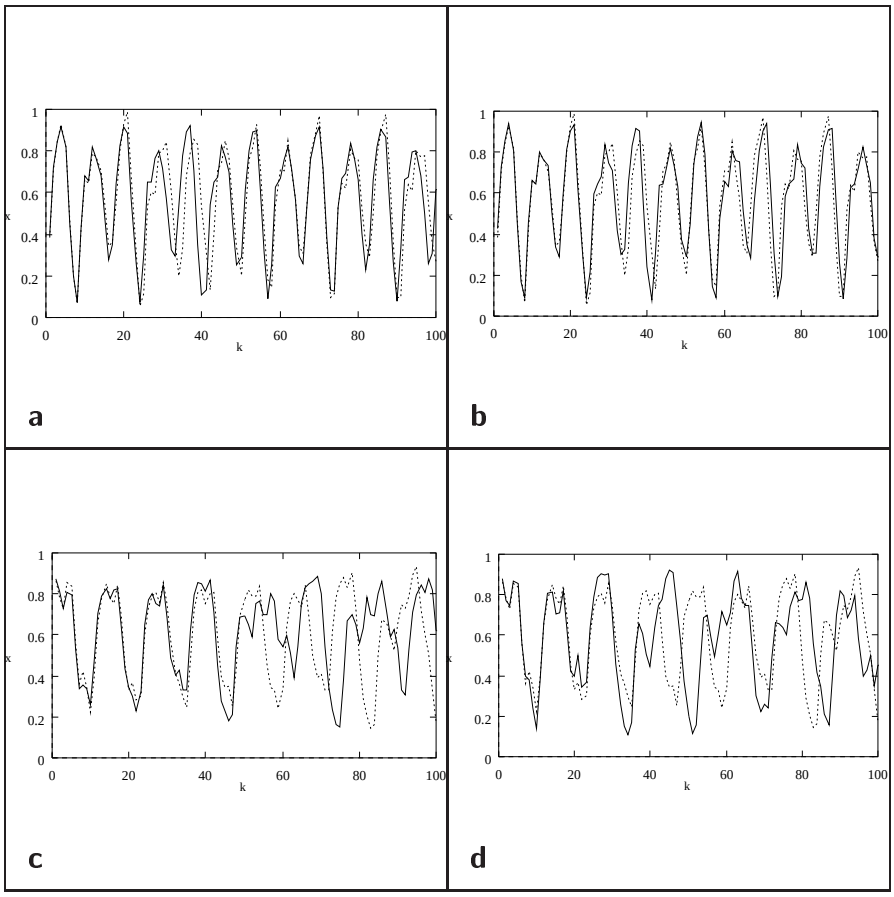


Figure 5.2. *Prédiction itérée de la suite de Mackey-Glass : a) réseau FIR, $\Delta = 17$, b) réseau DRNN, $\Delta = 17$, c) réseau FIR, $\Delta = 30$, d) réseau DRNN, $\Delta = 30$. En trait plein, la prédiction ; en pointillé, la suite simulée.*

$$\begin{aligned} x(k+1) &= 1.0 - 1.4x^2(k) + 0.3y(k) \\ y(k+1) &= x(k) \end{aligned} \tag{5.17}$$

Le comportement de la suite est illustré Fig. 5.4.2. L'image de l'attracteur dans l'espace de phases de x en fonction de y est représentée Fig. 5.5. Un agrandissement de l'attracteur révélerait une structure à l'échelle locale constituée d'une multitude de lignes parallèles, et ce quelque soit l'échelle. En fait, l'attracteur de la suite de Hénon est constitué d'un ensemble *non-dénombrable* de

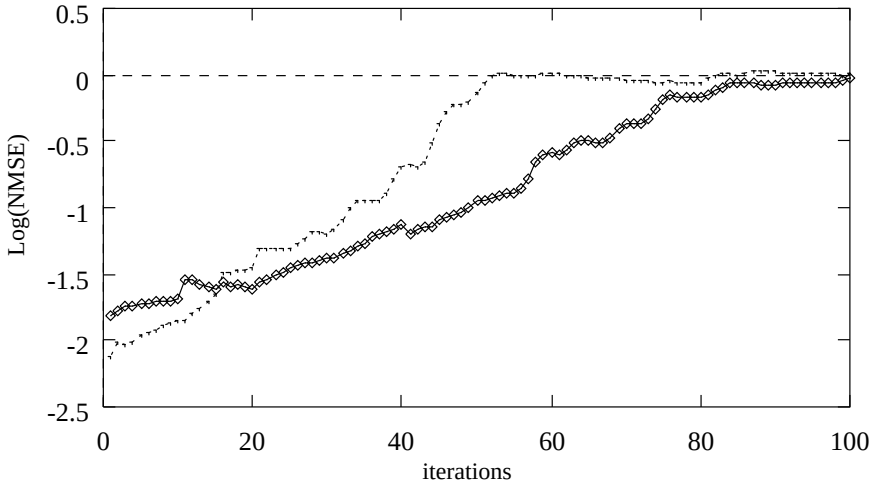


Figure 5.3. *Mackey-Glass : logarithme de l'erreur quadratique normalisée des prédictions itérées. La courbe en pointillé représente le DRNN, en trait plein, le réseau FIR.*

modèle	NMSE
Modèle linéaire	0.874
RN FIR	0.0017
DRNN	0.0012

Tableau 5.2. *Erreur normalisée des prédictions de la suite de Hénon à un pas en avance.*

lignes contiguës : il s'agit d'un attracteur *étrange*. Le calcul numérique montre que la dimension fractale de l'attracteur Fig. 5.5 est de dimension $D = 1.26$, confirmant le qualificatif d'attracteur *étrange* [OTT 93]. Un réseau FIR de dimension $1 \times 12 \times 12 \times 1$ avec des filtres d'ordre $3:1:1$ est entraîné ainsi qu'un modèle DRNN de dimension plus modeste : $1 \times 10 \times 1$ avec des filtres d'ordre $3:1:0$. L'apprentissage est effectué sur 5000 points de la suite et les résultats de la prédiction à un pas en avance sont présentés Table 5.2.

La prédiction itérée, non représentée, est remarquablement exacte jusqu'à l'itération 10 puis diverge au-delà. La divergence est une conséquence incontournable de la nature chaotique de la suite. Mais la suite issue du réseau itéré en bouclage fermé conserve les propriétés inhérentes au système comme le montre le tracé de l'attracteur dans l'espace des phases Fig. 5.5. Le réseau de neurones a capturé au cours de l'apprentissage la dynamique sous-jacente du système original représenté par (5.17). Il est intéressant de remarquer que le

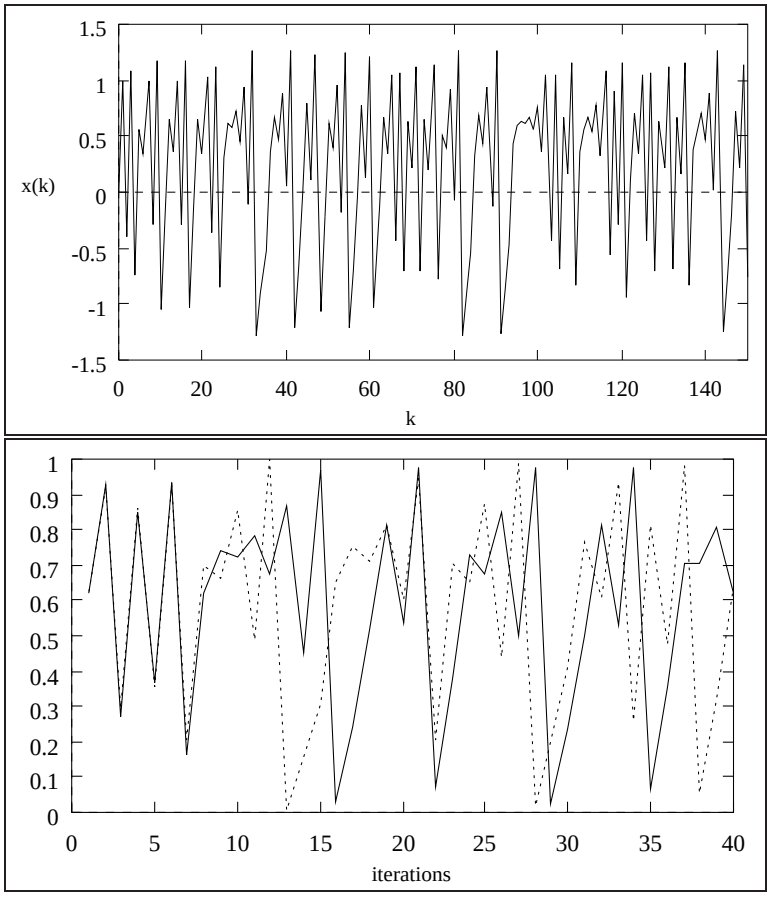


Figure 5.4. *Suite de Hénon (en haut), et prédictions itérées (en bas).*

bruit ne vient pas corrompre la dynamique de réseau et donc que l'attracteur est stable. Wan [WAN 93] a montré à quel point un réseau FIR était robuste en bruitant la suite avec un bruit blanc Gaussien d'un rapport signal/bruit de 7.47 dB. Le réseau entraîné sur la base d'apprentissage bruitée est en mesure de reconstituer la forme précise de l'attracteur original.

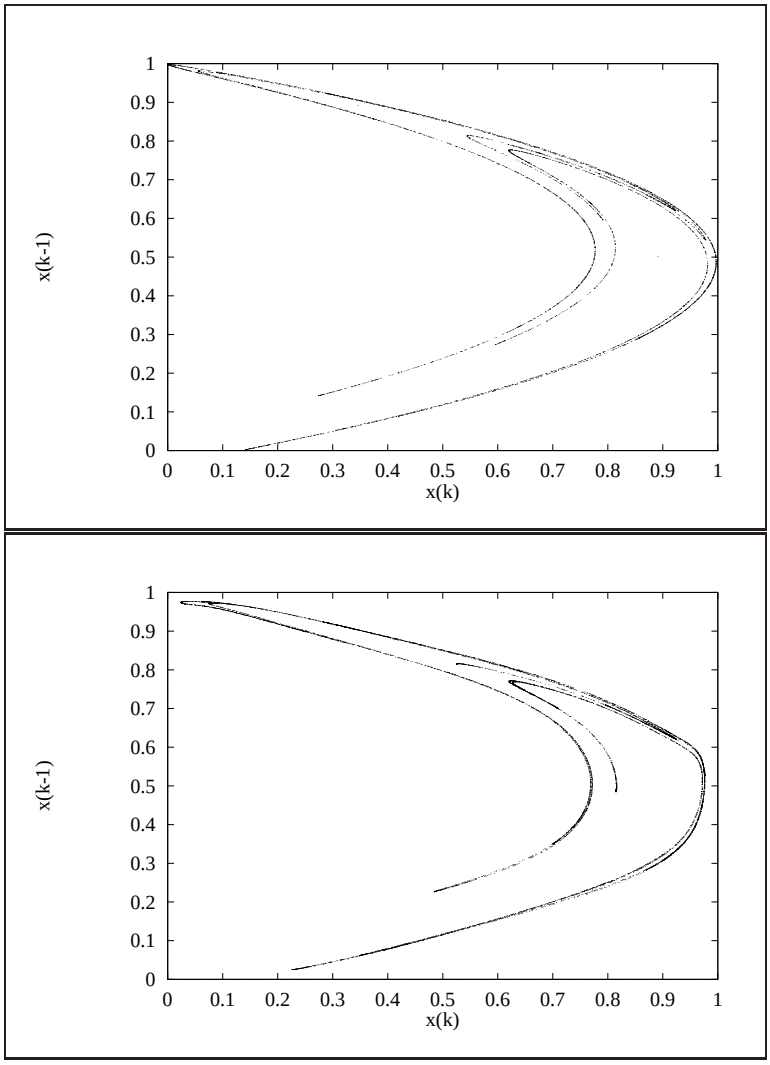


Figure 5.5. *Attracteur de Hénon : (en haut) espace des phases de* $x(k-1)$ *en fonction de* $x(k)$*, (en bas) attracteur reconstruit par un DRNN* $1 \times 10 \times 1$ *d'ordre* $3:1:0$.

5.4.3. *Les équations de Lorenz*

Le système de Lorenz est une représentation simplifiée des équations du mouvement d'un fluide en deux dimensions soumis à des échanges thermiques.

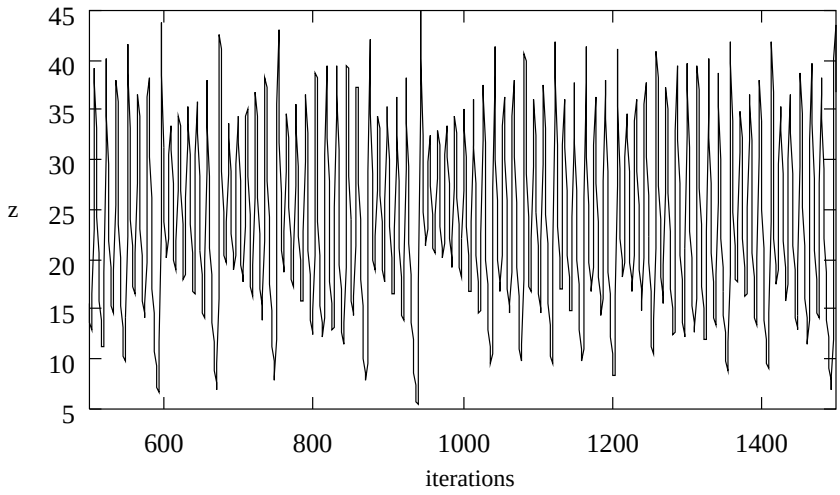


Figure 5.6. *Equations de Lorenz : évolution de z.*

Ce système d'équations décrit aussi les fluctuations d'un laser infrarouge. Les équations différentielles du système tridimensionnel sont les suivantes :

$$\begin{aligned} \frac{dx}{dt} &= -\sigma x + \sigma y \\ \frac{dy}{dt} &= -xz + rx - y \\ \frac{dz}{dt} &= xy - bz \end{aligned} \tag{5.18}$$

Les paramètres prennent leur valeur standard : $\sigma = 10, r = 28, b = 8/3$. Le système est émulé au moyen de la technique Runge-Kutta du 4^{ieme} ordre avec un pas de 0.01. Une suite est constituée en sélectionnant des échantillons avec une période 0.05s. Les variables x et z sont représentées Fig. 5.6. La nature chaotique du processus est manifeste au vu du tracé de z en fonction de x, Fig. 5.4.3. Le réseau de neurones est alimenté par $x(k)$ et fournit en sortie les estimations $\hat{x}(k+1)$ et $\hat{z}(k+1)$,

$$[\hat{x}(k+1), \hat{z}(k+1)] = \mathcal{N}_k(x(k)) \tag{5.19}$$

où $\mathcal{N}_k$ désigne le DRNN à l'instant k. En délivrant une information tronquée de l'état du système, il incombe au réseau de neurones la lourde tâche d'inférer les variables d'état manquantes du système à chaque instant.

Un DRNN de dimension $1 \times 10 \times 2$ avec des filtres d'ordre $30 : 1 : 0$ est entraîné. L'apprentissage est effectué sur 4000 points de la suite et les résultats de la prédiction à un pas en avance sont présentés Table 5.3. Les prédictions du réseau itéré Fig. 5.7 souffrent inévitablement de la divergence des trajectoires.

A la différence des simulations précédentes, le DRNN est ici entraîné à inférer la structure temporelle du processus chaotique, n'ayant à disposition qu'une seule variable d'état. La géométrie fractale de l'attracteur de Lorenz est illustré Fig. 5.4.3. Le tracé de $\hat{z}$ en fonction de $\hat{x}$, également visualisé, témoigne de la capacité du modèle neuronal à capturer la relation mutuelle entre les variables du vecteur d'état, nécessaire à la modélisation de la dynamique du système.

	NMSE	
modèle	x	z
Modèle linéaire	0.036	0.090
RN FIR	0.0070	0.0095
DRNN	0.0055	0.0078

Tableau 5.3. *Erreur normalisée logarithmique des prédictions de la suite de Lorenz à un pas en avance.*

5.4.4. *La suite d'Ikeda*

Pour le dernier exemple, nous considérons une équation dans le plan complexe qui émerge de l'étude des interactions d'une onde plane dans un laser. L'équation s'écrit

$$z(k+1) = a + bz(k)\exp(i[\phi - c/(1+|z(k)|^2)]) \tag{5.20}$$

avec $a = 1.0, b = 0.9, c = 6.0$ et $\phi = 0.4$. La partie réelle et imaginaire de la suite est illustrée Fig. 5.11 et le tracé de l'espace des phases est représenté Fig. 5.12. A l'instar de l'approche précédente, la tâche dévolue au réseau consiste à reconstruire le vecteur d'état en injectant en entrée du réseau la partie réelle uniquement. Le réseau fournit en sortie les estimations de la partie réelle et imaginaire,

$$(\mathrm{Re}[\hat{z}(k+1)], \mathrm{Im}[\hat{z}(k+1)]) = \mathcal{N}_k(\mathbf{w}, \mathrm{Re}[z(k)]) \tag{5.21}$$

où $\mathcal{N}_k$ désigne le DRNN à l'instant k.

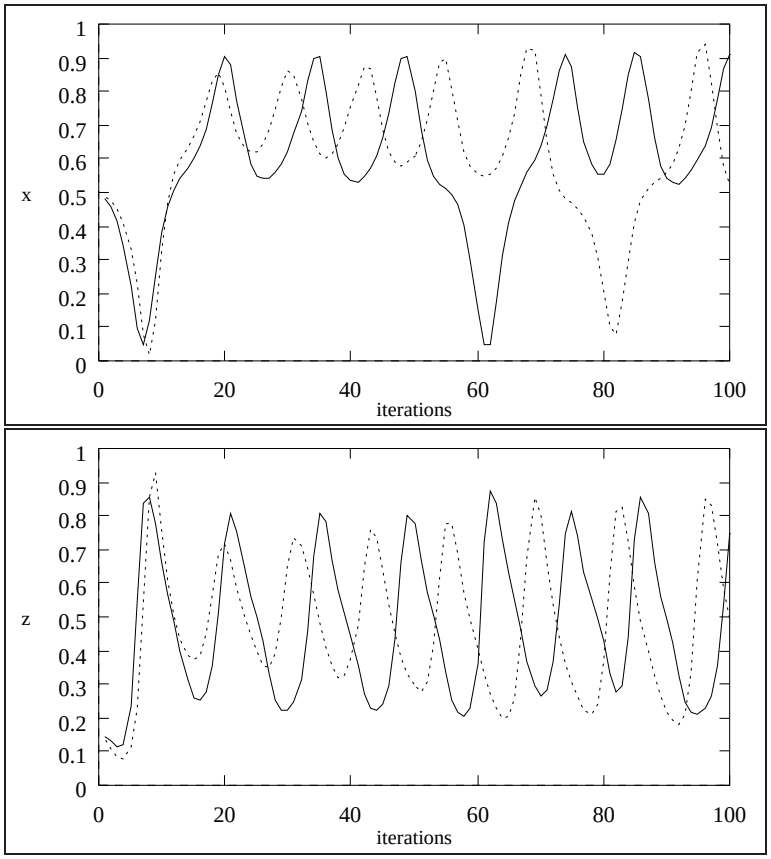


Figure 5.7. *Prédictions itérées de la suite de Lorenz.*

Un DRNN de dimension $1 \times 15 \times 2$ avec des filtres d'ordre $5 : 1 : 0$ est entraîné sur 10,000 points. La NMSE de la prédiction à un pas en avance, à l'issue de l'apprentissage, est affichée Table 5.4. La reconstruction de l'attracteur par le réseau est fidèle à Fig. 5.12. Une fois encore, le réseau est en mesure de capturer la relation mutuelle entre les variables d'état qu'exige la modélisation de la dynamique du système.

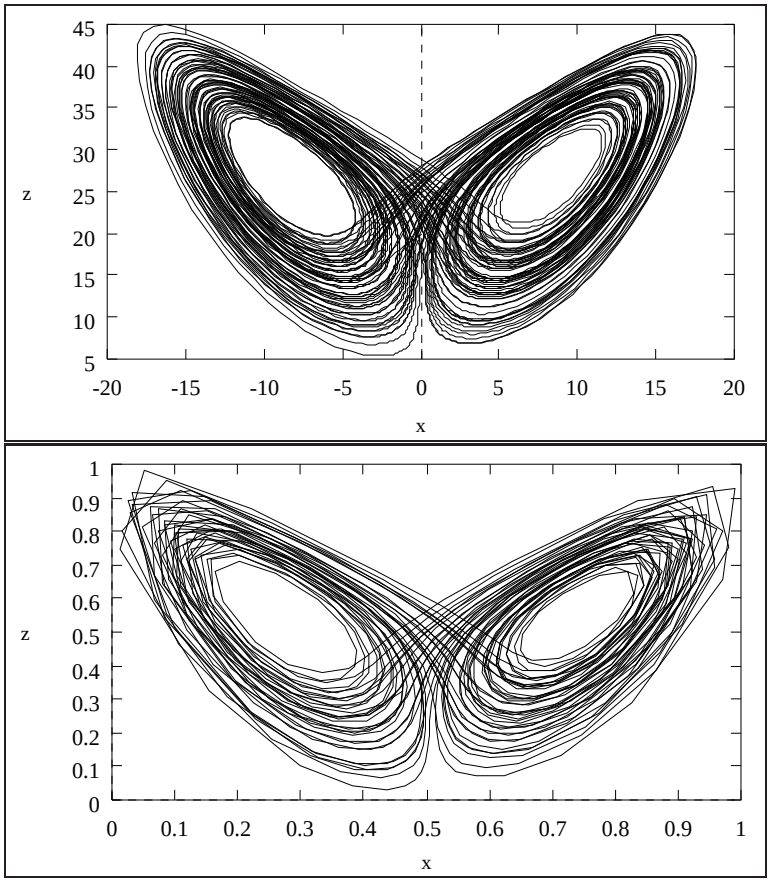


Figure 5.8. *Attracteur des équations de Lorenz : (en haut) espace des phases de* z *en fonction de* x *calculé sur 10000 points de la séquence originale obtenus par simulation avec la méthode de Runge-Kutta* 4^{ieme} *ordre. En bas : attracteur de l'équation de Lorenz reconstruit avec le DRNN.*

	NMSE	
modèle	x	z
Modèle linéaire	0.640	0.715
RN FIR	0.0080	0.0150
DRNN	0.0063	0.0134

Tableau 5.4. *Erreur normalisée logarithmique des prédictions de la suite d'Ikeda à un pas en avance.*

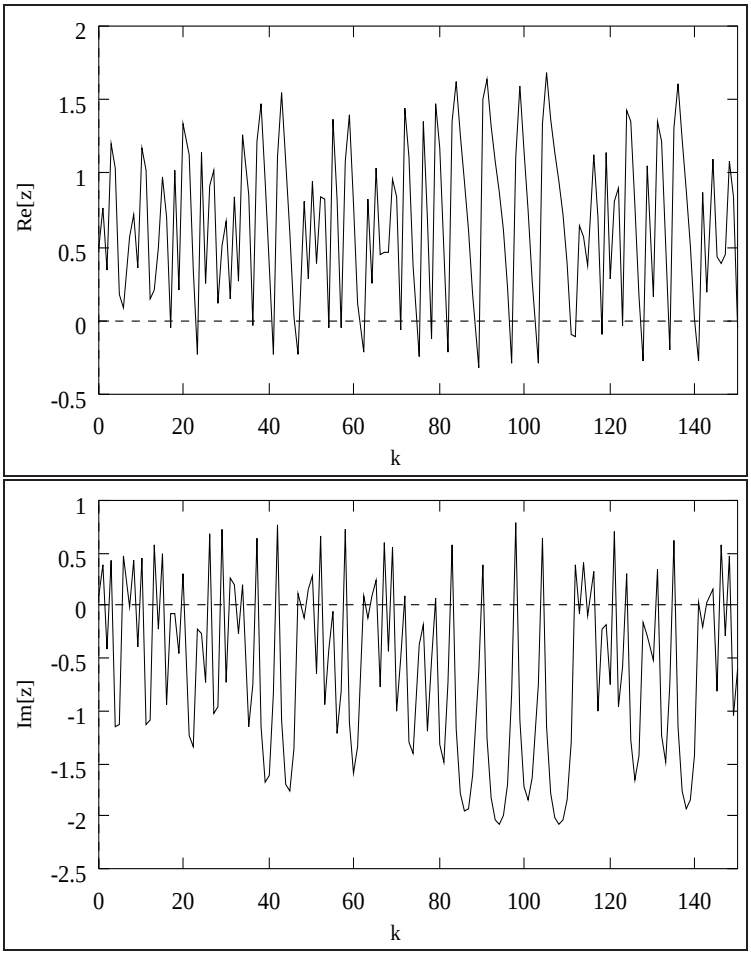


Figure 5.9. *Partie réelle et partie imaginaire du nombre complexe issu de l'équation d'Ikeda.*

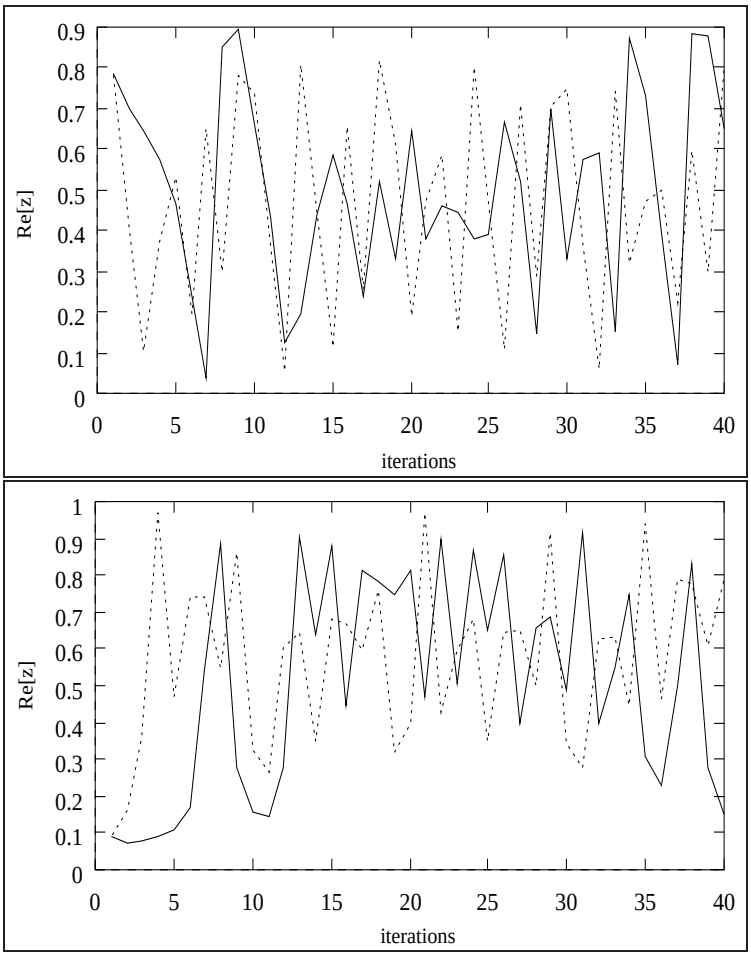


Figure 5.10. *Prédictions itérées des équations d'Ikeda : a) partie réelle, b) partie imaginaire.*

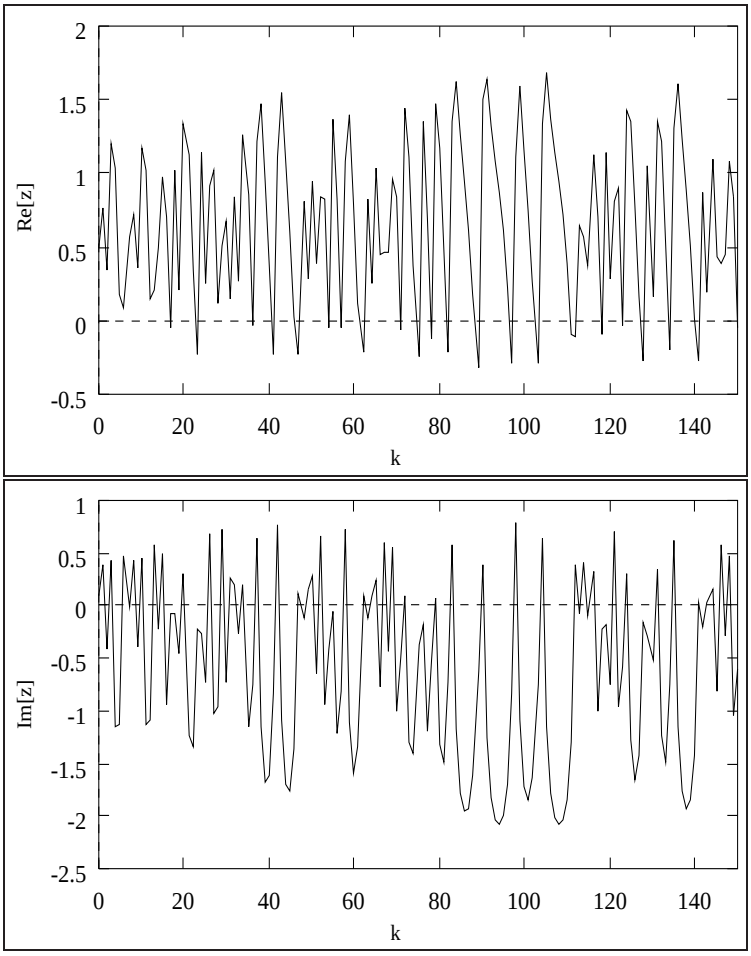


Figure 5.11. *Partie réelle et partie imaginaire du nombre complexe issu de l'équation d'Ikeda.*

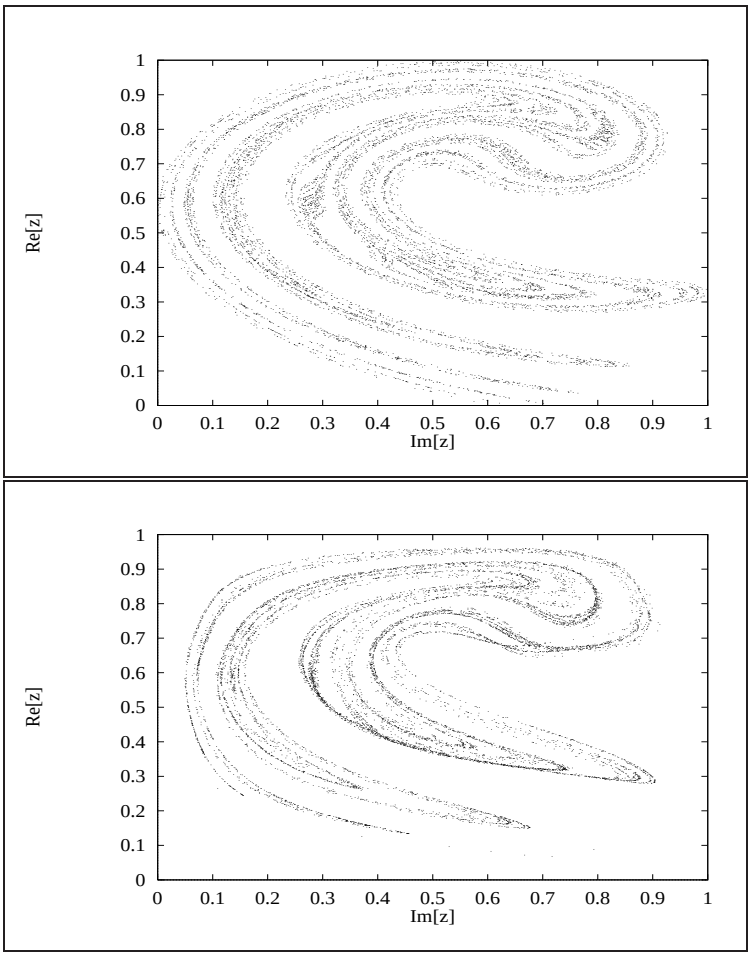


Figure 5.12. *En haut, attracteur de la suite d'Ikeda : $Re[z]$ en fonction de $Im[z]$. En bas, attracteur de l'équation d'Ikeda reconstruit avec le DRNN.*

5.5. Prédictions de température à la surface de la mer

Dans le cadre d'un projet réalisé en 1999 et 2000 pour le *Marine Environment Unit* du *Joint Research Center* (JRC) à ISPRA (Italie), ce paragraphe présente brièvement la méthodologie mise en oeuvre pour prédire la température à la surface de la mer (SST) sous forme de cartes 2D, dans une zone maritime où l'on observe des mouvement ascendants d'eau froide, que l'on désigne par phénomène d'*upwelling*, e.g. au large de la Namibie.

La gestion des zones de pêche côtières exige des outils fiables de modélisation et de prédiction des phénomènes maritimes (e.g., courants, SST). Prédire un courant froid permettrait de prévoir le mouvement des flux de plancton et donc également les bancs de poissons. En effet, l'eau froide en provenance des profondeurs charrie le plancton dont les poissons se nourrissent.

Actuellement, seuls les modèles numériques à grande échelle sont fiables pour des prédictions à plusieurs jours de la SST. Sur des zones locales, une modélisation physique plus fine est nécessaire car des phénomènes nouveaux apparaissent (e.g. l'upwelling). Le Marine Environment Unit développe des outils d'analyse et de gestion en milieu marin, en particulier le modèle de simulation numérique de la circulation océanique ISPRAMIX. Ce dernier est appliqué à l'étude locale du phénomène d'upwelling. ISPRAMIX fournit la valeur de SST à quelques jours à l'avance en chaque point d'une grille après de lourds calculs. Des prédictions météorologiques globales (e.g. vent, température de l'air) sont fournies en entrée du simulateur.

Nous allons montrer que la distribution de la SST et les lignes de contour peuvent également être construites à moindre coût, sans l'aide directe d'ISPRAMIX, grâce à la collaboration de plusieurs RN en charge de la prédiction de la SST en chaque point. Il suffit d'interpoler les valeurs pour reconstituer la carte 2D. ISPRAMIX servira de tuteur au RN, on dira que le réseau de neurones est un *meta-modèle* du modèle de circulation océanique.

Le phénomène d'upwelling - En océanographie, l'upwelling est un terme decrivant le processus maritime, produisant un mouvement ascendant des eaux profondes vers la surface. Comme il dépend d'un régime hydrologique particulier, on le retrouve sous des latitudes et des conditions souvent très différentes : en Antarctique, dans les zones équatoriales, etc. Cependant, l'upwelling le plus intensif est habituellement observé prés des côtes Est des océans.

D'après la théorie d'Ekman (1905), il a été démontré que la dynamique du phénomène d'upwelling est due à la circulation des vents dominants. Ceux-ci, par l'intermédiaire du *stress* (i.e., projection du vent sur un axe) qu'ils exercent sur la surface de l'océan, engendent un flot dans la couche supérieure de ce dernier, qu'on désigne souvent comme transport d'Ekman, et qui est toujours

dirigé de 90° vers la droite par rapport à la direction du vent dominant dans l'hémisphère nord et de 90° vers la gauche dans l'hémisphère sud, ce que l'on peut s'expliquer en partie grâce à la force d'inertie de Coriolis. Lorsque ce flot est divergent, on observe alors une remontée des eaux profondes qui équilibre le bilan de conservation de la masse d'eau dans la couche de surface.

Le modèle ISPRAMIX - Le modèle de simulation hydrodynamique ISPRAMIX, est un modèle numérique tridimensionnel variationnel, intégrant l'assimilation de données satellitaires à l'aide de méthodes variationnelles, pour des zones côtières océaniques. Les méthodes numériques de ce dernier sont basées sur l'intégration, par une méthode d'éléments finis volumiques, des équations fondamentales de l'hydrodynamique utilisant l'approximation de Boussinesq. ISPRAMIX, et ses extensions, est écrit en FORTRAN 77 - environ 10000 et fonctionne sur Cray J916 (6 processeurs, 1Gb de mémoire partagée) du CCR et une autre version fonctionnant sur Dec Alpha. ISPRAMIX a été calibré sur les phénomène d'upwelling du Nord-Est de l'Afrique. La zone étudiée ici est la Namibie.

5.5.1. *Les données de SST*

Pour les valeurs de la SST, on utilise les résultats d'une simulation d'ISPRAMIX sur le site de la Namibie. La durée totale de la simulation est de deux ans, de janvier 1992 à décembre 1993. Le modèle a ainsi calculé, une valeur de température par jour, pour tous les points de la fenêtre d'étude (de $-29°N$ $9°E$ à $-15°N$ $16°E$), sous la forme d'une matrice de taille 86x70, où la côte terrestre est représentée grâce à un masque de valeur constante. On dispose donc de valeurs de SST, à raison d'une matrice par jour et par fichier, convention aussi usité pour les autres grandeurs physiques calculés par le modèle.

Concernant les différentes grandeurs d'origine météorologiques (stress éolien, transfert de chaleur..), elles sont fournies par la sortie du module SeaFlux intégré à ISPRAMIX, qui utilise les données du Centre Européen des prédictions météorologiques(ECWF), et des modèles de dérivation empiriques pour calculer le forçage océan-atmosphère,

Pré-traitement - Dans le but de faciliter la manipulation de ces données, une mise en forme a été nécessaire :

- extraction : les réseaux sont entraînés avec l'historique de la température en des points précis de la fenêtre d'étude, puis normalisées pour gommer

les disparités entre les diverses variables météo selon la transformation classique :

$$x_i \mapsto \sigma_x^{-1}(x_i - \bar{x}) \tag{5.22}$$

on mémorise la moyenne et l'ecart-type pour permettre la transformation inverse en sortie.

– Le vecteur d'entrée est constitué d'un historique de la SST sur quelques jours, accompagné de quelques variables météorologiques *prédites* par modèle numérique ISPRAMIX. Par exemple, pour une prédiction à 3 jours et un historique de 3 jours, la composante $WindSpeed_x$ ou *stress* (projection du vecteur sur l'axe x) est une donnée météorologique additionnelle. Le vecteur d'entrée est donc

  – $SST(j-2), SST(j-1), SST(j)$,
  – $WindSpeed_x(j-2), WindSpeed_x(j-1), WindSpeed_x(j)$,
  – $WindSpeed_x(j+1), WindSpeed_x(j+2), WindSpeed_x(j+3)$,

  En sortie du modèle est une estimation de la SST à 3 jours $SST(j+3)$. Encore une fois, les valeurs futures de $WindSpeed_x(j+1), \ldots$ sont fournies par ISPRAMIX.

5.5.2. *Résultats*

Pour mesurer l'influence des paramètres du RN et de l'historique d'entrée, on a effectué plusieurs apprentissages en un point arbitraire (20,20). Les meilleures configurations au sens de la NMSE ont été regroupées. Une première série d'apprentissages a d'abord été réalisée pour laquelle la taille de l'historique des vents comblait le fossé temporel entre l'historique des températures et la température prédite. Cette méthode de prédiction est plus fidèle à la réalité du modèle ISPRAMIX. Car celui-ci utilise les données météorologiques jusqu'à l'instant où l'on souhaite faire la prédiction.

Pour évaluer l'utilité du modèle numérique ISPRAMIX sur les performances, une deuxième série d'apprentissages a été menée en ignorant les sortie du modèle ISPRAMIX. L'historique des vents s'arrête au même instant que l'historique de la SST passée. Il s'agit d'une utilisation du type "boîte noire" standard sans information exogène sur le processus observé. Comme le montre la table suivante, une dégradation sensible des performances est observée pour le DRNN.

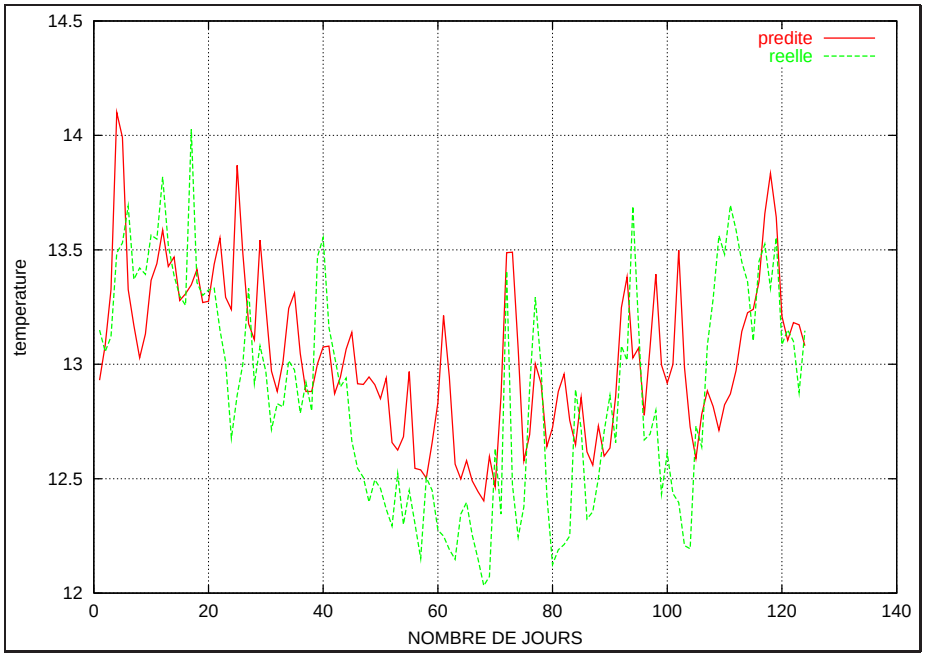


Figure 5.13. *Courbe de SST en un point sur une base de test.*

Modèle	histo T	histo V	futur	$NMSE_{app}$	$NMSE_{test}$
CC	-	-	3 jrs	0.0625	0.7606
MLP	3 jrs	6 jrs	3 jrs	0.0435	0.7381
DRNN ($D = 2$)	3 jrs	6 jrs	3 jrs	0.0488	0.6771
CC	-	-	6 jrs	0.1023	0.9153
MLP	6 jrs	12 jrs	6 jrs	0.0652	2.5527

Figure 5.14. *Comparaison de NMSE pour la prédiction future de la SST (CC = Carbon Copy).*

5.5.3. *Reconstruction de cartes météorologiques*

La prédiction de SST, montre tout son intérêt, lorsqu'elle permet de prédire l'ensemble d'une carte. On est alors à même d'assimiler beaucoup plus d'informations, absentes des prédictions ponctuelles : comme par exemple, la répartition spatiale en zones de dynamique similaire, l'influence de la proximité côtiére.

Plusieurs méthodes de construction de cartes peuvent être envisagées [FUE 00].

Modèle	histo T	histo V	futur	$NMSE_{app}$	$NMSE_{test}$
CC	-	-	3 jrs	0.0625	0.7606
MLP	3 jrs	3 jrs	3 jrs	0.0544	0.8302
DRNN ($D = 2$)	3 jrs	3 jrs	3 jrs	0.0592	0.8891
CC	-	-	6 jrs	0.1023	0.9153
MLP	6 jrs	6 jrs	6 jrs	0.0789	1.8493

Figure 5.15. *Comparaison de NMSE pour la prédiction future de la SST (CC = Carbon Copy). Les sortie du modèle ISPRAMIX sont ignorées). Une sensible dégradation des performances est observée pour le DRNN.*

Pour conserver la localité des opérations, un RN est d'entraîné en chaque point d'un maillage, la température au point(x,y) est supposée ne dépendre que de l'historique précédent des températures au même point. Chaque RN est relaxé *indépendamment* sur chacun des points de la carte.

Outre le côté illustratif, la carte des prédictions, Figure 5.16, met en évidence l'allure générale des fronts à fort gradient de température, et révèle ainsi en avance les courants froids, malgré des erreurs de température ponctuels. La visualisation agréable des cartes a été possible grâce au logiciel de visualisation Grads.

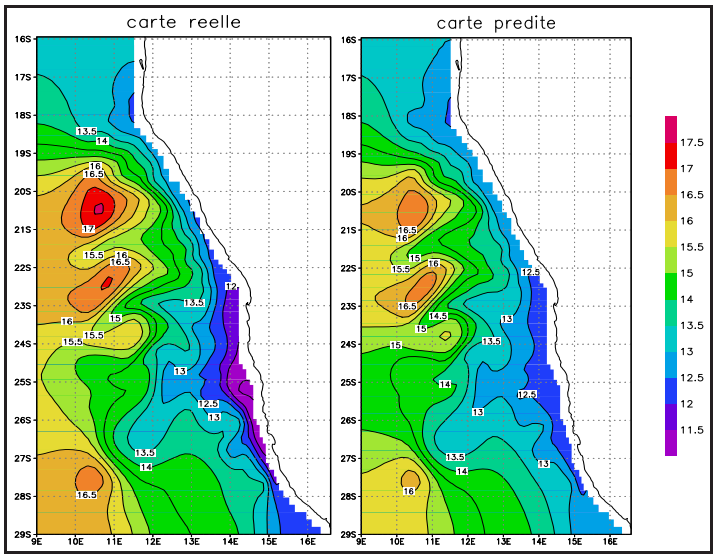


Figure 5.16. *Prédiction à 3 jours de la SST le 25/08/93*

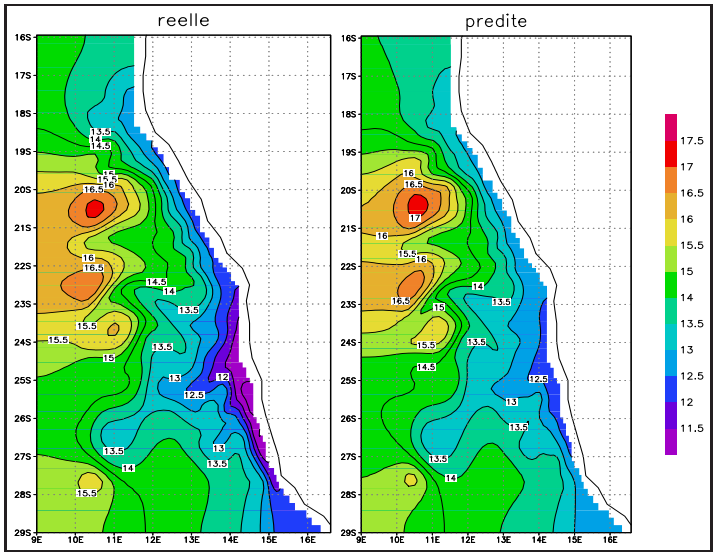


Figure 5.17. *Prédiction à 3 jours de la SST le 06/09/93*

Lors d'une prédiction pour un seul jour, au niveau de l'évaluation de l'erreur, le temps n'intervient et on peut alors se ramener à mesurer une MSE entre les deux cartes. L'erreur quadratique moyenne de d'approximation est

$$MSE_{th} = \frac{1}{m(\mathcal{D}_x \times \mathcal{D}_y)} \int_{\mathcal{D}_x} \int_{\mathcal{D}_y} (pred(x,y) - reel(x,y))^2 \, dx \, dy \tag{5.23}$$

On l'approxime naturellement par

$$MSE_{emp} = \frac{1}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} (pred(i,j) - reel(i,j))^2 \tag{5.24}$$

Les cartes sont construites par une interpolation cubique des points de la grille. L'erreur a seulement été calculée en chaque point de la grille. Le minimun, le maximum et l'écart type sur toute la carte. L'évaluation numérique des caractéristiques de l'erreur commise lors de la prédiction d'une carte ,à l'aide de grandeurs statistiques - moyenne, écart-type, minimum, maximum - donne des résultats satisfaisants comme le montre la table et les figures suivantes. Fig. 5.5.3 illustre les cartes obtenues au large de la Namibie en Novembre 1993, sur un horizon de prévision 6 jours, par interpolation cubique d'une grille de

MLP. Les résultats obtenus ont été jugés très satisfaisants par l'équipe MEU du JRC. Les fronts d'onde sont manifestement bien anticipés.

Modèle	histo T	histo V	futur	NMSE	écart-type	min
PMC - délai 0	3 jrs	6 jrs	3 jrs	0.19	0.21	1.17e-9
DRNN - délai 2	3 jrs	6 jrs	3 jrs	0.14	0.32	1.24e-8
PMC - délai 0	6 jrs	12 jrs	6 jrs	0.82	1.11	2.12e-8
PMC - délai 0	6 jrs	6 jrs	6 jrs	0.47	0.63	2.92e-7

Figure 5.18. *Statistiques des erreurs quadratiques moyennes normalisées (NMSE) lors de la reconstruction de carte 2D : moyenne, écart-type et minimum de la NMSE.*

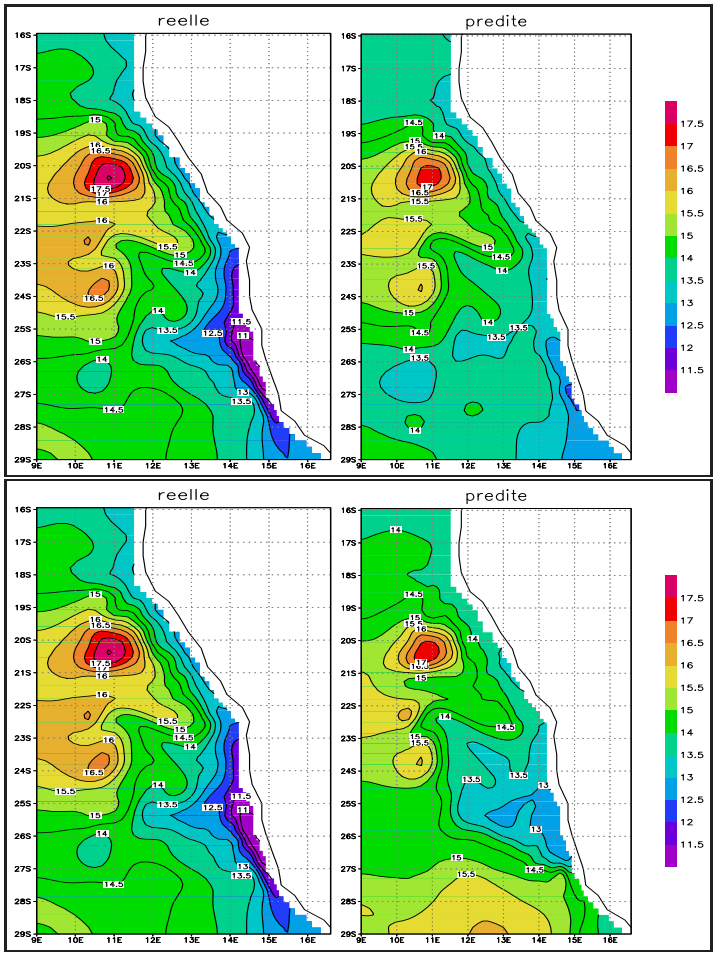


Figure 5.19. *Prédiction à 6 jours de la SST le 25/09/93 (en haut) sans données météo futures, (en bas) avec données météo futures.*

5.6. Prédiction des fluctuations du *seeing* astronomique

Ce paragraphe retrace brièvement les grandes étapes d'un projet de recherche de 16 mois mené pour le *European Southern Observatory* (ESO) [AUS 00d], dédié à la prévision en ligne des fluctuations du *seeing* astronomique à partir de mesures météorologiques prises au sol, sur des horizons de $10, 20, \ldots, 60$ minutes. Situé à Munich, ESO conçoit, entretient et exploite les télescopes (terrestres) des observatoires astronomiques dans l'hémisphère austral. Plusieurs ingénieurs ISIMA ont contribué significativement à ces travaux, dans le cadre de projets-ingénieur et stages DEA, en particulier Germain Tran pour la réalisation des scripts en PERL pour la partie temps-réel [TRA 01], ainsi que G. Stauffer et F. Moerel pour l'analyse de données [STA 00].

Le **seeing** ou **FWHM** - par essence un coefficient de visibilité -, est défini comme la pleine largeur d'une image stellaire longue pause , au demi maximum à la longueur de 50 nm au zénith (FWHM, 'full width at half maximum'). Le FWHM est une mesure de la diffraction des ondes lumineuses dûe aux perturbations atmosphériques. Ses fluctuations sont d'une grande importance pour les astronomes parce que les opération de calibrage sont menées quelque temps avant l'observation. Or le *seeing* ne doit pas trop varié entre temps. Quand les astronomes finissent une observation, le calibrage des instruments est mis en place pour l'observation suivante. La prédiction d'une mesure de variabilité du *seeing* - de 10 à 60 minutes en avance - permettrait l'ordonnancement optimale des tâches de calibration et d'observation. C'est l'un des rôles de l'Astronomical Site Monitor (ASM) de ESO pour le Very Large Telescope à Cerro Paranal au Chili.

Des années d'observation du *seeing* en différents sites a montré que le *seeing* n'est pas stationnaire et motivé la définition d'une mesure de variabilité, le FSC, i.e. le "fractional seeing change" (voir références dans [AUS 00d]) , dont l'impact négatif sur les performances des systèmes d'optique adaptative est avéré. Devant l'importance d'anticiper les fluctuations du *seeing*, des mesures ont été réalisées. Ces fluctuations induisent des erreurs de calibration optique. A partir de mesures météorologiques prises au sol toutes les minutes, il s'agit d'anticiper la variabilité du *seeing* de 10 à 60 minutes en avance. Les observations courtes sont préférables lorsque le *seeing* est très variable et vice-versa. Les variabilités sont observées à plusieurs échelles de temps, jusqu'à plusieurs heures. Un historique de valeurs de septembre 1999 a été utilisé pour ajuster le modèle.

La modélisation est du type boîte noire ; la physique du phénomène n'est pas prise en compte. Le processus sous-jacent est supposé obéir à une représentation d'état générale avec un bruit de d'état et bruit de mesure [DRE 02]. Les variables météo sont la commande du processus et le *seeing* est l'observation.

5.6.1. *Variabilité du* seeing

Une façon pratique et intuitive *aux yeux des astronomes* de caractériser la variabilité du *seeing* est le *Finite Exposure Fractional Seeing Change* (FEFSC). Cet indice exprime le changement *relatif moyen* du *seeing* sur une plage de temps donnée, car les valeurs instantanées du *seeing* sont sans importance. Il est défini ainsi

$$FEFSC(t,\tau) = 2\frac{\sum_{i=0}^{\tau} FWHM(t+\Delta t-i) - FWHM(t-i)}{\sum_{i=0}^{\tau} FWHM(t+\Delta t-i) + FWHM(t-i)} \tag{5.25}$$

FWHM désigne la valeur du *seeing* à l'instant t. Une valeur nulle du FEFSC est idéale. τ vaut typiquement 10 minutes.

Sélection des données et critères de performance - Les données météorologiques sont enregistrées toutes les 2 secondes. Elles sont agrégées à la minute. Une analyse en composantes principales (PCA) a été menée pour sélectionner les variables d'entrée qui influent sur la valeur du *seeing* à 10 minute [STA 00, TRA 01]. Sur les 17 variables initialement disponibles, 8 ont été retenues par les physiciens ESO pour la PCA, et à l'issue des nombreuses expérimentations, 4 variables ont été sélectionnées :

- TA1S : la température à 30m au dessus du sol,
- TA1S - TA2S : la température à 30m moins celle à 2m (au dessus du sol),
- W1S : la vitesse du vent à 30m au dessus du sol,
- FWHM : la valeur courante du *seeing*.

Les critères de performance choisis sont le MSE et le taux de réussite (hit rate) pour la tendance.

5.6.2. *Apprentissage en temps réel*

Le système alimente en temps réel la base d'exemples avec de nouvelles données et élimine les plus anciennes. Le principe est de rendre le système réactif. Intuitivement, il doit être prêt à s'adapter à des variations rapides tout en conservant de l'information du passé. Plusieurs tailles de base d'apprentissages ont été jaugées à l'aune de l'erreur sur la base de test [STA 00, TRA 01, AUS 00d]. Une fenêtre mouvante de 6 jours (3000 couples E/S en apprentissage) à été choisie. Après d'innombrables essais, nous avons choisi de décaler la base d'apprentissage de 500 valeurs chaque jour. L'apprentissage est lancé

une fois par jour. Ce compromis s'est montré satisfaisant pour appréhender la non-stationnarité supposée du signal.

Les DRNN pour les prédiction à $10, \ldots, 60$ minutes sont constitués de 4 entrées, 6 unités cachées et 1 sortie, le FEFSC prédit. Les délais des entrées vers les unités cachées varient de 30 à 60 minutes. Les sorties retardées sont réinjectées en entrée. Chaque modèle contient aux alentours de 650 paramètres. Des simulations extensives ont été conduites sur les données 1999. Les critères de performances sont résumés dans les table ci-dessous, en phase opérationnelle et comparée au prédicteur de référence des physiciens : le 'Carbon Copy', i.e. $\hat{(x_t)} = x_{t-1}$.

	10 min	20 min	30 min	40 min	50 min	60 min
MSE train	0.014	0.032	0.043	0.047	0.055	0.049
MSE test	0.014	0.034	0.039	0.057	0.057	0.085
Taux de réussite	67%	62%	64%	68%	69%	71%

Tableau 5.5. *Performances de la prédiction du FEFSC de 10 à 60 minutes en avance moyennée sur 10 réplications sur la base de test.*

Prédiction Cible	Positif	Négatif
Positif	30 .66	19.34
Négatif	18.54	31.46

Tableau 5.6. *Table des contingences des prévisions moyenne obtenue avec le DRNN en phase opérationnelle. Hit rate = 62.11, Loss rate = 37.89.*

Prédiction Cible	Positif	Négatif
Positif	24.40	26.40
Négatif	26.40	22.80

Tableau 5.7. *Tables des contingences des prévisions obtenues avec le prédicteur CC (en bas) en phase opérationnelle. Hit rate = 47.2, Loss rate = 52.8.*

En conclusion, nous obtenons un Hit rate variant entre 55% et 68% en phase opérationnelle sur l'horizon 10 minutes. Les autres horizons ne sont pas encore opérationnels, et devraient l'être courant 2003. Le lecteur désireux de voire le système fonctionner en temps continu pourra se connecter sur le site :

www.eso.org/gen-fac/pubs/astclim/forecast/meteo/neurop/

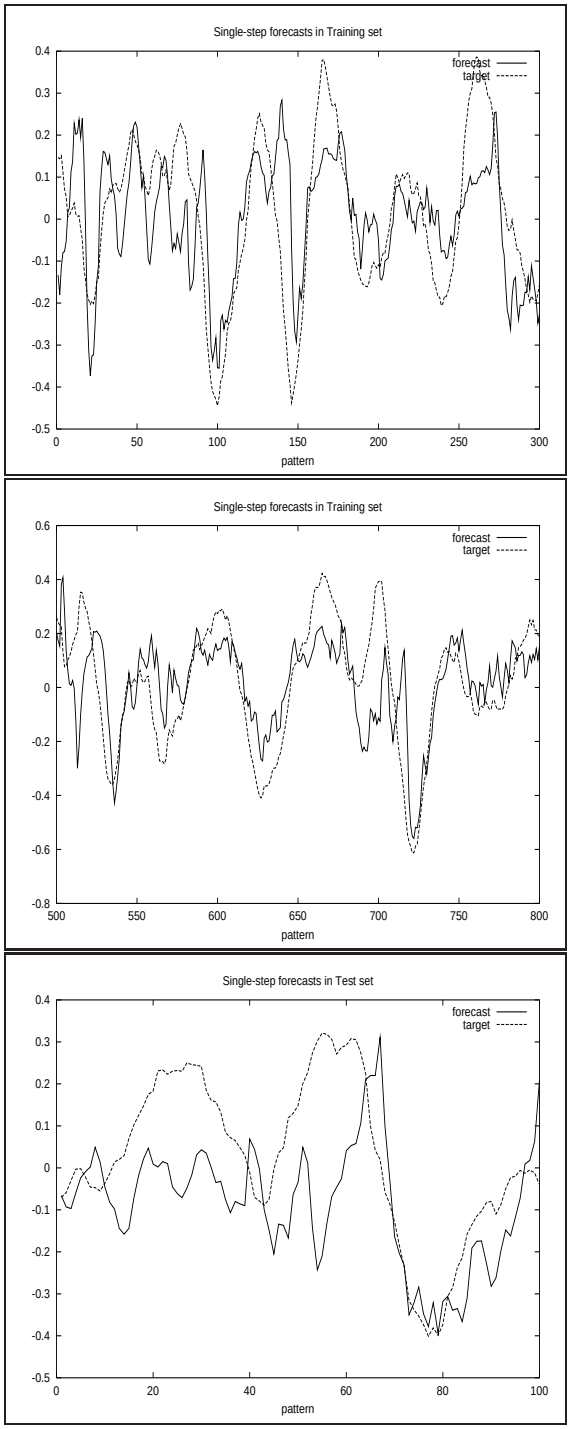


Figure 5.20. *Illustration des prévisions obtenues avec le DRNN en phase de test après un apprentissage sur une fenêtre glissante des 3000 valeurs passées. Prédictions à 20 minutes du* seeing *sur 2 extraits de la base d'apprentissage (en haut) et sur un extrait de la base de test (en bas).*

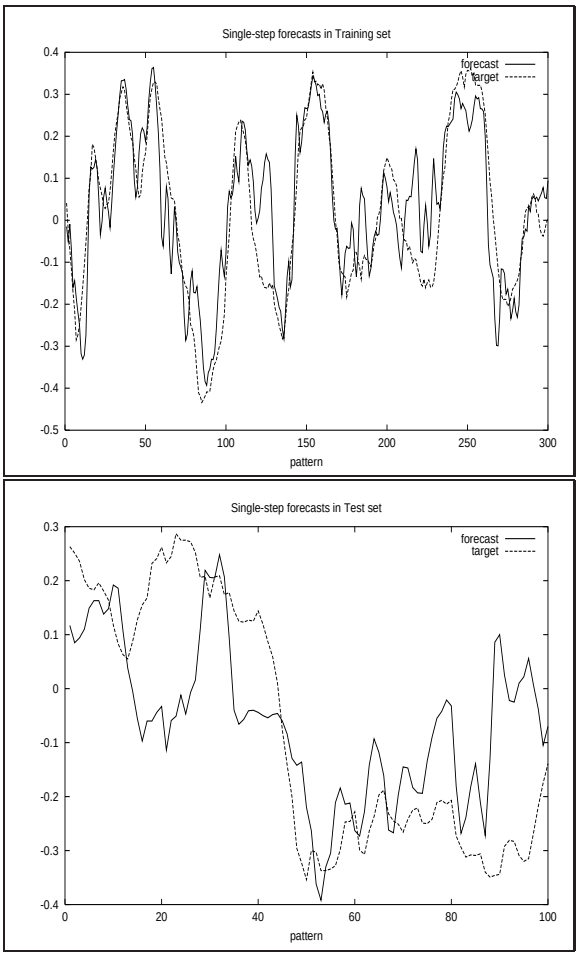


Figure 5.21. *Illustration des prévisions obtenues avec le DRNN en phase de test sur 500 minutes après un apprentissage sur une fenêtre glissante des 3000 valeurs passées. Prédictions à 30 minutes du* seeing *sur 2 extraits de la base d'apprentissage (en haut) et sur un extrait de la base de test (en bas).*

5.7. Le principe de la méta-modélisation

Nombreux sont les phénomènes complexes modélisés par des simulateurs à événements discrets complexes, parfois couplées à des Systèmes d'Information Géographique, lesquels fournissent en sortie des grandeurs ou des séries de nombres se prêtant à l'analyse anticipative du phénomène physique modélisé. Toutefois, l'obtention de ces traces est en général très gourmande en temps de calcul. Il est donc fréquent d'entraîner un réseau de neurones grâce aux traces issues des réplications des simulations stochastiques afin de fournir des prédictions portant sur le processus physique observé. Les articles parus dans la littérature parlent de *méta-modélisation* [KIL 94]. Le réseau de neurones permet, au terme de l'apprentissage, un gain en temps de calcul considérable puisqu'il est possible d'anticiper l'évolution d'un système stochastique complexe (le modèle), par un processus déterministe plus simple (le méta-modèle).

Une fois sélectionnés, les paramètres utiles sont injectés en entrée du réseau de neurones lequel se charge de fournir un certain nombre de prédictions portant sur le processus modélisé, assorti des intervalles de confiance correspondant. La technique de méta-modélisation a été mis en oeuvre dans le projet de prévision de la SST, dans le chapitre consacré à la prévision de la QoS dans les réseaux télécom, mais également du projet LIFE "Control of the spread of the *Caulerpa Taxifolia* in the Mediterranean" (programme DG XI) consacré à la prévision de **la surface contaminée par la caulerpe après plusieurs années dans la bassin méditerranéen** avec David Hill. Le lecteur intéressé pourra consulter [AUS 99b, AUS 00b] pour une présentation détaillée du projet.

On retiendra que l'erreur de la méta-modélisation associée à la vraie réponse, Y, du système physique se décompose en trois termes, $Y - \hat{\hat{Y}} = \epsilon_m + \epsilon_s + \gamma$, où γ est le bruit additif de mesure, ϵ_s est le terme d'erreur qui rend compte de l'erreur de simulation, en raison des variables qui ont été omises par le simulateur, et ϵ_m est le terme d'erreur additionnelle responsable de l'inadéquation du modèle neuronal avec le simulateur [AUS 99b].

5.7.1. *Conclusion et perspectives*

J'ai présenté dans ce chapitre les applications des réseaux discrets bouclés à délais que j'ai menées dans le domaine des sciences environnementales de 1995 jusqu'à nos jours. La prévision à un pas, à plusieurs pas, la simulation (en itérant le modèle en bouclage fermé), ainsi que la "méta-modélisation" ont été évoqués sur des exemples tirés de l'observation de phénomènes physiques.

Le projet de recherche avec ESO se poursuivra jusqu'en 2003 dans l'optique

de regrouper sur le cite Web de l'Astronomical Site Monitor (ASM) Neural Network Operational Prognosticator (NEUROP)[2]. Il reste à mettre en oeuvre les prévisions temps-réel sur des horizons de 20 à 60 minutes, la prévision à 10 minutes est déjà opérationnelle.

[2]www.eso.org/gen-fac/pubs/astclim/forecast/meteo/neurop/

Chapitre 6

PREVISION DU TRAFIC TELECOM PAR ANALYSE MULTIRESOLUTION

6.1. Introduction

La multirésolution est une façon élégante de prédire les séries temporelles à mémoire longue par l'utilisation d'une communauté de réseaux opérant sur des échelles temps-fréquence distinctes [AUS 97, MUR 96b, MUR 97, MUR 98a, MUR 98b, AUS 98c, AUS 98b, AUS 01c, SOL 00]. Une décomposition en ondelettes permet une analyse en temps-fréquence du signal. C'est un outil fort utile à la compression d'image, le filtrage, la reconnaissance, la détection et la reproduction de processus en $1/f$ ou des mouvements browniens fractionnaires [ABR 01, DAU 92, STA 94, STA 95, STA 96, STR 96]. C'est aussi l'outil approprié pour l'étude des comportements en lois d'échelles mettant en jeu toutes les échelles temps-fréquence conjointement.

La décomposition opère par projections successives sur des sous-espaces de $L^2(\mathbb{R})$ dont les fonctions de bases sont les dilatées/translatées de la même fonction : l'ondelette mère. Ainsi, le signal d'entrée (i.e., la série temporelle étudiée) est décomposé avec l'algorithme dit "à trous" [STA 96] en une hiérarchie de signaux de détails *stationnaires* exhibant une *mémoire à courte portée*, à laquelle s'ajoute un signal résiduel basse fréquence. Le signal original s'exprime comme la somme des signaux de détails auquel s'ajoute le signal résiduel. La décomposition met à jour des structures de dépendances à des échelles différentes. Par cet artifice, les dépendances à long terme peuvent être exploitées par un banc de réseaux de neurones indépendants dotés d'une mémoire à court terme. La stratégie de prédiction repose sur une subdivision de la tâche globale en tâches élémentaires. Chaque échelle est traitée individuellement et indépendamment

des autres par un réseau de neurones. Les sorties des réseaux sont alors recombinées pour fournir la prédiction du modèle. Les prédictions fournies par cette méthode hybride présentent typiquement une variance plus élevée que celle obtenue par une méthode directe fondée sur l'artifice de la fenêtre temporelle, sans pour autant apporter de gain notable en terme d'erreur quadratique.

Cette technique a été mise en oeuvre sur des données environnementales [AUS 97], télécom [AUS 98a] et financières [AUS 98b]. Dans ce chapitre, nous présentons une application à la prévision, une minute à l'avance, du volume de données télé-chargées, en termes d'octets, sur un serveur Web, grâce à un fichier de logs HTTP. En effet, l'enchevêtrement complexe de flux hétérogènes émanant de sources de nature et de type différents confère au trafic des lois d'échelle de temps dont la disparité est d'une étonnante amplitude (de la milliseconde, temps caractéristique de la technologie de transfert, à l'heure, temps caractéristique d'une sessions Web), soit au moins six décades. Les phénomènes d'invariance d'échelle du télé-trafic, dans son acception au sens large (e.g., ISDN, Ethernet LAN, WAN, TCP, FTP, Telnet, voir par exemple [ABR 01]), ont été observés et documentés au cours de ces dix dernières années [CRO 96, WIL 95]. Les statistiques marginales des arrivées des paquets présentent typiquement des *queues lourdes*, témoignant de structures de dépendance longue portée et la présence de comportement en loi d'échelle.

La description et la modélisation statistique fine du télé-trafic permet d'améliorer la conception et le contrôle des réseaux de télécommunication. La dimensionnement des files d'attente, des routeurs, des capacités des liens repose sur une estimation fiable de la qualité de service (QoS) comme nous le verrons au chapitre 7. Celle-ci est tributaire du contrôle d'accès, du contrôle de congestion, du protocole de routage, mais également du comportement statistique des flux de données. Aussi, la prévision à l'échelle de la minute du télé-trafic - et donc indirectement de la QoS -, permettrait une allocation *dynamique* de la mémoire partagée des files d'attente en entrée des commutateurs. De même, une prédiction fiable à brève échéance de la charge d'un serveur Web autoriserait une gestion automatisée des caches [CRO 96]. A ce titre, c'est un enjeu technologique et économique majeur.

6.2. Analyse multirésolution

Pour comprendre l'intérêt de l'algorithme "à trous" en prévision, il est bon de comprendre au préalable l'esprit de l'analyse multirésolution des signaux. Une *analyse multirésolution* [MAL 89, STR 96, STA 96] de $L^2(\mathbb{R})$ est caractérisée par une suite de sous-espaces V_j, $j \in \mathbb{Z}$, de $L^2(\mathbb{R})$, emboîtés,

$$\ldots \subset V_2 \subset V_1 \subset V_0 \ldots, \tag{6.1}$$

vérifiant les propriétés suivantes :

- $\bigcap_{j\in\mathbb{Z}} V_j = \{0\}, \bigcup_{j\in\mathbb{Z}} V_j$ est dense dans $L^2(\mathbb{R})$,
- $f(t) \in V_j \Longleftrightarrow f(2t) \in V_{j-1}$, $\forall j \in \mathbb{Z}$,
- $f(t) \in V_0 \Longleftrightarrow f(t-k) \in V_0$, $\forall k \in \mathbb{Z}$,
- il existe une fonction $\phi(t) \in V_0$ appelée *fonction d'échelle* telle que l'ensemble $\{\phi(t-k), k \in \mathbb{Z}\}$ constitue une base de Riesz[1] de V_0.

Des propriétés précédentes, il résulte que $\{\phi_{j,k}(t) := 2^{-j}\phi(2^{-j}t-k), k \in \mathbb{Z}\}$ constitue une base de Riesz de l'espace V_j. Le principe de l'analyse multirésolution est d'effectuer des projections successives sur des espaces V_j, réalisant ainsi une approximation de plus en plus grossière de $f(t)$. La projection de $f(t)$ sur V_j est caractérisée par les coefficients $\{c_{j,k}\}_{k\in\mathbb{Z}}$, où $c_{j,k}$ est le produit scalaire dans $L^2(\mathbb{R})$ de $f(t)$ avec la fonction échelle translatée et dilatée,

$$c_{j,k} =< f(t), 2^{-j}\phi(2^{-j}t-k) > . \tag{6.2}$$

D'une approximation à l'autre, lorsqu'on passe de V_{j-1} à V_j, une partie de l'information est perdue. Elle est contenue dans les *détails*. La séquence des détails s'obtient par projection de $f(t)$ sur des sous-epsaces, W_j, supplémentaires de V_j dans V_{j-1}, tels que

$$V_j + W_j = V_{j-1}. \tag{6.3}$$

On peut associer à la fonction échelle $\phi(t)$ une fonction $\psi(t)$ telle que les fonctions $\{\psi_{j,k}(t) := 2^{-j}\psi(2^{-j}t-k), k \in \mathbb{Z}\}$ constituent une base de Riesz de l'espace W_j.

L'espace V_0 est inclus dans V_{-1}, donc $\phi(t)$ est aussi dans V_{-1} et peut donc s'écrire comme une combinaison des fonctions $\phi(2t-k)$. Il en va de même pour W_0. Posons h_k et g_k les coefficients de ces combinaisons, il vient

$$\begin{aligned} \phi(t) &= 2\sum_k h_k \phi(2t-k) \\ \psi(t) &= 2\sum_k g_k \phi(2t-k). \end{aligned} \tag{6.4}$$

[1] base de Riesz = base *stable* en dimension infinie, ou base *inconditionnelle* p.69 [STR 96]

On parle de l'*équation de dilatation* et de l'*équation d'ondelette* respectivement. Ces équations établissent un lien crucial entre les ondelettes et les filtres. En effet, Equation (6.4) permet de calculer immédiatement les coefficients $c_{j+1,k}$ et $w_{j+1,k}$ pour $j > 0$ à partir des $c_{j,k}$, et ce en partant de $c_{0,k}$:

$$\begin{aligned} c_{j+1,k} &= \sum_n h_{n-2k} c_{j,n}, \\ w_{j+1,k} &= \sum_n g_{n-2k} c_{j,n}. \end{aligned} \tag{6.5}$$

Ainsi, les coefficients de la décomposition se calculent itérativement à l'aide deux bans de filtres dits d'*analyse* : un filtre passe-bas et un filtre passe-bande. Ils scindent le signal en bandes de fréquences pour faciliter son traitement (compression, codage, mémorisation). On remarquera au demeurant le sous-échantillonnage effectué d'une échelle à l'autre : on parle de *décimation*. Les coefficients $c_{j,k}$ sont donc calculés aux noeuds d'une grille dite *dyadique*. Ils partagent avec leurs voisins dans le plan temps-fréquence une fraction d'information autour de la date k à une fréquence 2^j. La décomposition présente un propriété remarquable : la suite formée des coefficients de détails est stationnaire et ne contient que de la mémoire à courte portée.

Le premier banc de filtres a été construit. Afin de restaurer le signal original, S. Mallat [MAL 89] utilise des filtres orthogonaux mais la théorie a été généralisée à une large classe de filtres par l'introduction de deux autres filtres $\tilde{h}$ et $\tilde{g}$, les conjugués de h et g. La reconstruction s'effectue par l'opération inverse,

$$c_{j,k} = 2 \sum_n \left[\tilde{h}_{k+2n} c_{j+1,n} + \tilde{g}_{k+2n} w_{j+1,n} \right]. \tag{6.6}$$

Des conditions supplémentaires sont requisent pour une restauration Exacte. Posons $\hat{h}(\nu)$ la transformée de Fourrier de la fonction $\sum_n h_n \delta(t-n)$ (i.e., $\hat{h}(\nu) = \sum_n h_n e^{-2\pi i n \nu}$). Alors

$$\begin{aligned} \hat{h}(\nu + \frac{1}{2}) \hat{\tilde{h}}(\nu) + \hat{g}(\nu + \frac{1}{2}) \hat{\tilde{g}}(\nu) &= 0 \\ \hat{h}(\nu) \hat{\tilde{h}}(\nu) + \hat{g}(\nu) \hat{\tilde{g}}(\nu) &= 1. \end{aligned} \tag{6.7}$$

Les bases d'ondelettes *orthogonales* sont les plus populaires. Elles sont obtenues en prenant des espaces W_j orthogonaux entre eux et orthogonaux aux V_j (pour tout j), et en imposant l'orthogonalité des bases $\psi_{j,k}$, $\forall j, k \in \mathbb{Z}^2$. Dans le domaine complexe, ces conditions se traduisent par [STA 96],

$$\begin{aligned}
\tilde{g}(\nu) &= e^{-2\pi i\nu}\hat{h}^{\star}(\nu+\frac{1}{2}),\\
\hat{\tilde{h}}(\nu) &= \hat{h}^{\star}(\nu),\\
\hat{\tilde{g}}(\nu) &= \hat{g}^{\star}(\nu),
\end{aligned} \tag{6.8}$$

et

$$|\hat{h}(\nu)|^2 + |\hat{h}(\nu+\frac{1}{2})|^2 = 1 \tag{6.9}$$

On montre aisément que cet ensemble de relations vérifient (6.7). Les ondelettes de Daubechies sont les seules solutions de support compact. La propriété d'orthogonalité restreint fortement le choix de l'ondelette mère. Il est possible de relâcher la contrainte d'orthogonalité sur les espaces W_j pour aboutir à une décomposition *bi-orthogonale*, plus souple d'utilisation, au prix de la présence de corrélation entre les coefficients. Quoi qu'il en soit, le choix des coefficients du filtre est guidé par la désir de doté la fonction échelle et l'ondelette mère d'une certaine *régularité*.

Dans le cas d'une transformée orthogonale, chaque fonction $f(t) \in L^2(\mathbb{R})$ peut se décomposer ainsi

$$P_{V_0} f(t) = P_{V_k} f(t) + \sum_{j=0}^{k} P_{W_j} f(t), \tag{6.10}$$

avec

$$\begin{aligned}
P_{V_j} f(t) &= \sum_k c_{j,k}\phi_{j,k}(t),\\
P_{W_j} f(t) &= \sum_k w_{j,k}\psi_{j,k}(t),
\end{aligned} \tag{6.11}$$

6.3. Algorithme à trous

L'algorithme dit "à trous" [SHE 92, STA 96] réalise une transformée en ondelettes discrète dite *stationnaire* ou *redondante* car la décimation n'est pas effectuée. Elle est dite discrète non pas en référence au signal, mais parce que les coefficients sont calculés sur un grille dyadique. Le signal, supposé continu,

n'est connu qu'en certains points x_k. D'ordinaire, on suppose que chaque coefficient $\{c_{0,k}\}$, qui est par définition le produit scalaire à l'instant k de la fonction $f(t)$ (inconnue) avec la fonction échelle $\phi_{0,k}(t) := \phi(t-k)$, est égal à la *valeur* de la fonction en k, c'est-à -dire $c_{0,k} = x_k$. Ce point délicat est discuté dans [ABR 01].

D'après l'équation de dilatation de $\psi(t)$

$$\psi(t) = 2\sum_k g_k \phi(2t-k). \tag{6.12}$$

On sait calculer les coefficients d'ondelettes à partir des coefficients d'approximation,

$$w_{j+1,k} = \sum_n g_{n-2k} c_{j,n}. \tag{6.13}$$

Or, dans le cas de la multirésolution, ils peuvent être obtenus plus simplement grâce à $c_{j+1,n}$, par différences successives,

$$w_{j+1,k} = c_{j,k} - c_{j+1,k}. \tag{6.14}$$

Ainsi, la différence $c_{j,k} - c_{j+1,k}$ est l'information perdue entre les deux échelles j et $j+1$ à l'instant k. L'équation de dilatation de $\psi(t)$ s'écrit donc

$$\psi(t) = 2\phi(2t) - \phi(t), \tag{6.15}$$

La distance entre les coefficients croît d'un facteur 2 entre deux échelles successives. Une convolution avec le filtre passe-bas h donne

$$c_{j+1,k} = \sum_n h_n c_{j,k+2^j n}. \tag{6.16}$$

Remarquez l'accroissement des distances entre les points (i.e. $2^j n$) dû à l'absence de décimation. On aurait obtenu le même résultat en inserrant des "0" dans les coefficients du filtre h, d'où l'origine du nom "à trous". La formule de reconstruction jusqu'à l'échelle J est donnée par

$$c_{0,t} = c_{J,t} + \sum_{j=1}^{J} w_{j,t}. \tag{6.17}$$

On impose donc de reconstruire la série original simplement en *sommant* les séries de détail et la série des approximations résiduelles (les trends) à l'octave J. Comparée à 6.10 et 6.11, on remarque que la forme explicite des fonctions échelle et ondelette intervient nullement dans la synthèse du signal. Cette propriété simplifie considérablement le processus de reconstruction et implique un couplage fort entre l'ondelette mère et la fonction échelle. On comprend aussi intuitivement pourquoi ce couplage induit des contraintes sur le coefficient des filtres,

$$h_0 + g_0 = 1, \text{ et } h_n = -g_n, \ \forall n \neq 0. \tag{6.18}$$

En remplaçant les valeurs de g_n dans (6.13), on retrouve effectivement la formule (6.14). Notons que d'autres contraintes (e.g. de régularité des filtres) peuvent être imposées. Pour conclure très succinctement, on notera qu'il existe plusieurs façon de construire une multirésolution : 1) en partant des V_j, 2) en identifiant la fonction échelle $\phi(t)$, ou 3) directement à partir des coefficients h_k du filtre passe-bas. Par exemple, les filtres les plus simples satisfaisant (6.18) sont les filtres de Haar définis par

$$h_0 = h_1 = \frac{1}{2}, \text{ et } g_0 = -g_1 = \frac{1}{2}. \tag{6.19}$$

Avec l'ondelette de Haar, on obtient donc

$$c_{j+1,t} = \frac{1}{2}(c_{j,t-2^j} + c_{j,t}) \tag{6.20}$$

On obtient par récurrence les relations suivantes [SOL 00], pour tout $j = 1, \ldots, J$.

$$\begin{aligned} c_{J,t} &= \frac{1}{2^J} \sum_{m=0}^{2^J-1} x_{t-m}, \\ w_{j,t} &= \frac{1}{2^J} \left(- \sum_{m=0}^{2^{j-1}-1} x_{t-2^{j-1}-m} + \sum_{m=0}^{2^j-1} x_{t-m} \right) \end{aligned} \tag{6.21}$$

On peut identifier directement la fonction échelle $\phi(t)$. Prenons, par exemple, la fonction triangle

$$\begin{aligned} \phi(t) &= 1 - |t|, \text{ si } t \in [-1, 1] \\ \phi(t) &= 0, \text{ sinon} \end{aligned} \tag{6.22}$$

Cette fonction vérifie

$$\frac{1}{2}\phi(\frac{t}{2}) = \frac{1}{4}\phi(t+1) + \frac{1}{2}\phi(t) + \frac{1}{4}\phi(t-1). \tag{6.23}$$

Par conséquent, on en déduit les coefficients des deux bans de filtres,

$$\begin{aligned} c_{j+1,t} &= \frac{1}{4}c_{j,t-2^j} + \frac{1}{2}c_{j,t} + \frac{1}{4}c_{j,t+2^j} \\ w_{j+1,t} &= -\frac{1}{4}c_{j,t-2^j} + \frac{1}{2}c_{j,t} - \frac{1}{4}c_{j,t+2^j}\,. \end{aligned} \tag{6.24}$$

Causalité - Il faut bien comprendre à ce stade que les effets de bords altèrent le calcul des coefficients. Les filtres utilisés ne sont pas causaux ; $c_{j,t}$ dépend en théorie du futur de la série $x_{t+1}, x_{t+2}, \ldots$. La causalité de l'algorithme est obtenue ici au prix d'une astuce classique [STA 96] : à l'instant t, on fera l'hypothèse que $c_{j,t+k} = c_{j,t-k}$ à toute échelle j.

A l'instant t, on dispose des observations $x(t), x(t-1), \ldots, x(1)$ et on désire estimer $x(t+1)$. L'idée est d'appliquer la transformée en ondelettes discrets sur $x(t), x(t-1), \ldots, x(1)$, pour obtenir les coefficients $\{c_{j,k}\}$ pour $k = t, t-1, \ldots, 1$ et $j = 1, 2, \ldots, J$. Les prédictions sont réalisées indépendamment sur chaque suite de coefficients. A l'instant $t+1$, seuls les nouveaux coefficients $c_{j,t+1}$ sont calculés en posant (effet miroir) $c_{j,t+k} = c_{j,t-k}$ [STA 96] à toute échelle j. Seulement, lorsque x_{t+1} est disponible, il faudrait en toute rigueur recalculer les anciens coefficients dont la valeur doit être mise à jour. Ce nombre se déduit de l'ordre des filtres FIR. Dans les expérimentations suivantes, les coefficients à l'instant courant ne sont pas recalculés dans le futur.

Choix d'ondelette - Pour permettre une analyse locale de la fonction f, la fonction ondelette et la fonction d'échelle doivent être localisées dans le temps et l'espace. C'est le cas par exemple de la famille de Daubechies : elles ont un support compact dans le domaine temporel et leur tranformée de Fourier décroît rapidement. Remarquez que lorsque le filtre passe-bas est de réponse impulsionnelle finie (i.e. nombre fini de coefficients h_n non nuls), alors on peut montrer que la fonction échelle est à support compact (p. 185, [STR 96]). Toutefois, l'allure de ces fonctions est clairement irrégulière. On leur préfère souvent les fonctions splines, polynomiales par morceaux, à support compact, symétriques par rapport à l'origine, et dont les qualités d'approximations sont notoires [STR 96]. En choisissant une fonction échelle du type **spline cubique** (de degré 3), on s'assure de la continuité des dérivées aux points de jonction

ainsi qu'une allure très régulière. Les coefficients du filtre passe-bas s'obtiennent facilement grâce à l'équation de dilatation. Ils valent

$$(\frac{1}{16}, \frac{1}{4}, \frac{3}{8}, \frac{1}{4}, \frac{1}{16}). \tag{6.25}$$

Les expérimentations présentées dans ce chapitre ont été réalisées avec ces coefficients. Les sous-espaces V_j et W_j demeurent orthogonaux entre eux, cependant les bases de ces espaces ne le sont plus. Les projection ne sont dès lors plus orthogonales.

Propriétés de la décomposition - La décomposition à trous est en $O(N)$ si N est la longueur de la suite. Notons qu'une fois calculés les coefficients $c_{j,t}$ à l'instant t, les anciens coefficients ne sont pas mis à jour à mesure que de nouvelles valeurs de la suite sont disponibles par souci de simplicité.

Une nouvelle décomposition biais/variance - La série temporelle étudiée est décomposée en plusieurs signaux de détails, et un signal résiduel. Le signal original s'exprime comme la somme des signaux de détails auquel s'ajoute le signal résiduel. Les sorties des réseaux sont alors simplement *additionnées* pour fournir la prédiction du modèle. On peut donc voir le banc de réseaux de neurones comme un comité d'experts, où chaque expert est "responsable" d'une échelle temps-fréquence. Considérons la relation $x_t = c_{0,t} = \sum_{j=1}^{J} w_{j,t}$ où le résidu $c_{J,t}$ a été renommé en $w_{J,t}$ par concision. Posons $\bar{w}_i = E[w_{i,t}]$ et $\bar{x} = E[x_t]$. On montre que sous certaines hypothèses[2], l'erreur de généralisation peut se décomposer en trois termes [AUS 02c],

$$\begin{aligned} E[(\hat{x}_t - x_t)^2] &= \sum_{i=1}^{N} \frac{\bar{x}}{\bar{w}_i} E[(\hat{w}_{i,t} - w_{i,t})^2] \\ &\quad + \sum_{i=1}^{N} \frac{\bar{w}_i}{\bar{x}} E[(\frac{\bar{x}}{\bar{w}_i} w_{i,t} - x_t)^2] \\ &\quad - \sum_{i=1}^{N} \frac{\bar{w}_i}{\bar{x}} E[(\frac{\bar{x}}{\bar{w}_i} \hat{w}_{i,t} - \hat{x}_t)^2] \end{aligned} \tag{6.26}$$

La formule est analogue à la décomposition bias-variance classique pour les comités d'experts [BIS 95]. Le premier terme est la contribution des erreurs sur chaque échelle, pondérées par le terme $\frac{\bar{x}}{\bar{w}_i}$ qui reflète la contribution de l'expert

[2] les innovations $\hat{w}_{i,t} - w_{i,t}$ sont indépendantes de $x_t - \frac{\bar{x}}{\bar{w}_i} w_{i,t}$, $\bar{x} \neq 0$ et $\bar{w}_i \neq 0, \forall i$.

dans l'estimation finale. Le second terme, indépendant du modèle, rend compte des écarts entre la cible et les coefficients pondérés. Enfin, le dernier terme est le pendant du second puisqu'il rend compte des écarts entre l'estimation finale du comité d'experts et les coefficients estimés par les experts également pondérés. Cette expression met en évidence la nécessité de trouver un compromis entre les deux derniers termes, de signes opposés. Un des problèmes de cette formule tient à l'hypothèse erronée que les $\bar{w}_i$ sont non nuls.

6.4. Application : Prédiction du trafic Web

Le Web est un système d'information distribué à grande échelle basé sur une architecture client-serveur. Aussi, la charge d'un serveur est le nombre de requêtes qui émanent de nombreux clients. Nous examinons ici les requêtes HTTP d'un serveur de Magee College de l'université d'Ulster (Irlande du Nord) de Mars 1996 à Février 1997 ainsi que les requêtes HTTP du European Southern Observatory (ESO) en Juin 1996. Les logs nous renseignent sur chaque requête traîtée par le serveur, à savoir le nom du client, la date, le fichier de l'objet désiré, et sa taille en termes d'octets de la réponse.

Compte tenu de la charge modérée du serveur de Magee College, nous considérons les nombres successifs de requêtes HTTP sur des plages distinctes de 60 minutes. Une suite chronologique de 8118 valeurs horaires normalisée dans l'intervalle unité est constituée. L'inspection des courbes révèle plusieurs régimes dynamiques patents correspondant aux périodes académiques. La suite a été divisée, seules 4755 valeurs ont été conservées pour l'expérimentation. Figure (6.1) illustre le résultat de la décomposition en ondelettes : une représentation partielle de la suite est affichée ainsi que les coefficients $w_1, \ldots, w_4$. Le cycle journalier de la demande est clairement représenté. On observe également le lissage graduel des séries w_j à mesure que l'on passe d'une octave à l'autre.

Les logs ESO reflêtent en revanche une situation de charge plus importante. Les logs ont été agrégés à la minute. 14.6% des valeurs sont nulles ; elles ont simplement été éliminées. Les nombreuses séquences de zéros traduisent une période de dysfonctionnement du réseau de la minute jusqu'à quelques heures. Les 34 726 mesures restantes ont été normalisées sur une échelle logarithmique pour des raisons numériques. Figure (6.2) (trait plein en haut à gauche) montre une représentation partielle de la suite ESO, ainsi que les coefficients $w_1, \ldots, w_4$ en traits pleins. Les fluctuations à l'échelle de la minute sont particulièrement irrégulières.

6.4.1. *Dépendances à longue portée*

La dépendance à longue portée induit l'invariance d'échelle. On parle de dépendance à longue portée ou de mémoire longue lorsque les observations disjoints restent "irrémédiablement" corrélées sur de longs intervalles de temps. Ces processus du second ordre stationnaires, X, sont caractérisés [ABR 01] par le comportement asymptotique en loi de puissance de leur fonction de covariance, $r_X(\tau)$,

$$r_X(\tau) \sim c_1 \tau^{\gamma-1}, \quad \nu \to \infty, \quad 0 < \gamma < 1. \tag{6.27}$$

Sous certaines conditions sur la régularité de $r_X(\tau)$ (e.g. monotonie asymptotique), on montre que le spectre de puissance du processus X , $\Gamma_X(\nu)$, obéit à une loi de puissance à l'origine

$$\Gamma_X(\nu) \sim c_2 |\nu|^{-\gamma}, \quad \nu \to 0, \quad 0 < \gamma < 1. \tag{6.28}$$

La lente décroissance de la fonction de covariance, et son corollaire, la divergence de la somme de la fonction de covariance,

$$\int r_X(\tau) d\tau = +\infty, \tag{6.29}$$

caractérisent la propriété de dépendance à longue portée. Cette propriété d'invariance d'échelle est vérifiée ici dans une gamme de fréquences limitée supérieurement. Son origine est essentiellement grande échelle puisqu'elle concerne les basses fréquences. Tous les processus autosimilaires avec $1/2 < H < 1$ présentent une dépendance à longue portée car

$$r_X(\tau) := E[|\delta X(\tau)|^2] \sim \sigma^2 H(2H-1)\tau^{2(H-1)} \tag{6.30}$$

ainsi, pour $1/2 < H < 1$, le processus X_t présente une dépendance longue portée de paramètre $\gamma = 2H - 1$. Typiquement, la dépendance longue portée s'étudie théoriquement et pratiquement par la technique d'agrégation [ABR 02]. La convergence asymptotique de la variance des suites agrégées vers une loi de puissance, i.e. $Var(X^{(m)}) = O(m^{-\gamma})$ avec $0 < \gamma < 1$, résulte de la dépendance à longue portée.

6.4.2. *Analyse des données*

L'objet de cette étude n'est pas d'établir la dépendance à longue portée des traces Web à notre disposition. On pourra consulter par exemple [ABR 01, ABR 02, WIL 95, CRO 96] pour une caractérisation de l'auto-similarité du trafic Telnet, TCP et HTTP. Pour autant, des techniques graphiques simples existent pour estimer le coefficient de Hurst.

Graphe temps-variance - Cette méthode repose sur le lent déclin de la variance de la suite agrégée lorsque la suite est autosimilaire. En effet, la fonction d'autocorrelation est invariante par agrégation. Ainsi, si X_t désigne une suite stationnaire avec $t = 1, 2, \ldots$, alors $X_t^{(m)}$ est la suite obtenue en sommant les X_t sur des intervalles disjoints de taille m unités de temps. Lorsque la variance de $X_t^{(m)}$ est tracée en fonction de m dans un diagramme *log-log*, une droite avec une pente supérieure à -1 indique l'auto-similarité. Le coefficient de Hurst, H, est estimé par $H = 1 + \gamma/2$.

Graphe LLCD - Cette méthode (*log-log cumulative distribution*, LLCD) repose sur l'étude asymptotique de la densité de probabilité des durées des connexions HTTP. Il s'agit de tracer $P(X^{(m)} > x)$ en échelle logarithmique. Pour tester si les données présentent une variance infinie, il faut inspecter la queue des courbes $P(X^{(m)} > x)$ pour différentes valeurs de m. Lorsque les pentes déclinent avec m, la variance finie est postulée et vice-versa.

Illustrons l'application de ces deux méthodes sur nos traces HTTP. Les graphes temps-variance et LLCD des deux séries sont représentés Figures (6.1) et (6.2).

Ulster - La pente du graphe temps-variance de la Figure (6.1) est estimé à -0.358. Il vient $H = 0.82$, caractéristique de l'auto-similarité, résultat très similaire de celui de Crovella et Bestravos [CRO 96] et de Willinger [WIL 95]. Bien que le tracé LLCD de la Figure (6.1) pour les suites agrégées ($m = 10, 100, 500$) soit typique d'une loi log-normale, la pente de la queue de la distribution ne change avec la valeur de m.

ESO - La pente du graphe temps-variance de la Figure (6.2) est estimé à -0.379. Il vient $H = 0.81$, caractéristique de l'auto-similarité, résultat très similaire au précédent. Le tracé LLCD est hélas peu lisible.

Sans pour autant constituer une preuve, les estimations de H laissent supposer un comportement autosimilaire avec des dépendances à longue portée. Aussi, nous faisons l'hypothèse que des événements éloignés dans le passé ont une influence sur le présent.

6.4.3. *Expérimentations*

La méthode de prévision multi-échelles est illustrée sur les données issues du serveur ESO, le plus chargé. L'application - sans réel succès toutefois - de cette méthode aux données Ulster est exposé dans [AUS 98c]. La sélection du nombre adéquate de neurones, de l'architecture et des délais y est décrite. L'architecture retenue des DRNN est arbitraire : elle consiste en 1 entrée, 1 sortie et 5 neurones cachés entièrement connectés. Les entrées sont connectées aux neurones cachés avec des délais variables.

A l'instant t, chaque réseau à l'échelle j dispose des coefficients $w_{j,k}, k \leq t$, et doit estimer $w_{j,t+1}$. Seuls les délais ont été ajustés intuitivement pour tenir compte du lissage accrû des suites aux échelles supérieures comme le montre les figures (6.1) et (6.2). Pour illustration, les performances présentées pour les DRNN sont comparées à celle d'un MLP sur la base du même nombre de paramètres ajustables pour ne privilégier aucune échelle, ni aucun modèle. Les entrées sont connectées aux neurones cachés avec des délais variables en fonction de l'échelle. Ils sont représentés Table (6.4.3).

	Délais en entrée
w_1	0,2,4,6,8,10
w_2	0,3,6,9,12,15
w_3	0,5,10,15,20,25
w_4	0,7,14,21,28,35
c_4	0,9,18,27,36,45

Tableau 6.1. *Délais de la fenêtre d'entrée du DRNN pour la prédiction à une minute en avant sur l'ensemble de test des coefficients* $w_i, \quad i = 1, \ldots, 4$ *plus* c_4.

	w_1	w_2	w_3	w_4	c_4
DRNN	0.37	0.32	0.27	0.18	0.08
MLP	0.39	0.35	0.29	0.19	0.08
Carbon copy	2.78	1.12	0.39	0.15	0.05

Tableau 6.2. *Log NMSE de la prédiction à une minute en avant sur l'ensemble de test des coefficients* $w_i, \quad i = 1, \ldots, 4$ *plus le résidu* $c4$ *par un DRNN, un MLP, et l'estimateur trivial "carbon copy", i.e.* $\hat{x}_{t+1} = x_t$. *Plus faible est la valeur, meilleure sont les prédictions.*

Un index de performance basique, le "carbon-copy error", fera emploi de benchmark trivial. Il s'agit de prendre comme estimateur la valeur de la suite à l'instant précédent. La prédiction à chaque échelle jusqu'à $J = 4$ est illustrée Figure (6.4). La NMSE de la prédiction à une minute en avant sur l'ensemble

de test des coefficients est représentée Table (6.4.3). L'erreur carbon-copy est plus favorable pour les deux derniers coefficients. L'erreur de prédiction de la recombinaison est $NMSE = 0.7$ en prenant le *meilleur modèle à chaque échelle*. Un gain significatif est obtenu par rapport à l'approche classique de la fenêtre temporelle en entrée d'un seul MLP : on obtient $NMSE = 0.85$ avec un un seul MLP alimenté en entrée par tous les coefficients w_j et le résidu c_5 de l'instant précédent.

Il est intéressant de porter son regard sur la Figure (6.5) dans laquelle les suites sont successivement additionnées les unes aux autres comme le suggère la formule de recombinaison. Visuellement, la performance se dégrade rapidement au fur et à mesure que l'on remonte dans les échelles en partant de c_J ($J = 4$). Par exemple, avec $w_4 + c_5$ on obtient $NMSE = 0.13$, avec $w_3 + w_4 + c_5$ on obtient $NMSE = 0.24$, et avec $w_2 + \ldots + c_5$ on obtient $NMSE = 0.38$. La recombinaison finale donne une NMSE de 0.7. L'erreur finale est supérieure à chacune des erreurs individuelles. Notons enfin que les résidus d'erreurs (non représentés ici) paraissent décorrélés au vu de la fonction d'auto-covariance. La dépendance à long terme est donc une propriété inhérente au modèle.

En conclusion, les résultats obtenus ne sont pas franchement favorables en terme d'erreur quadratique. Pour autant, nous avons observés des performances surprenantes en terme de *tendances* : dans environ 65 % des cas sur les bases de test en validation croisée, le signe des variations prédites et observées coïncident, alors que ce pourcentage plafonne aux alentours de 55% avec la méthode classique de la fenêtre temporelle.

Notons enfin que S. Soltani et al. [SOL 00] ont mené récemment une approche très similaire avec des données synthétiques issues d'un modèle ARfIMA, i.e., dont la différenciation *fractionnaire* du processus est ARMA. L'algorithme à trous est également employé, toutefois les prédictions sont effectuées avec un modèle auto-régressifs multidimensionnel du type $\mathbf{c}_k = \sum_{j=1}^{p} \mathbf{A}_j \mathbf{c}_{k-j}$ où $\mathbf{c}_k = (c_{1,k}, c_{2,k}, \ldots, c_{J,k}, w_{J,k})$ et $\mathbf{A}_j$ sont des matrices estimées par la méthode des moindres carrés récursifs. Les échelles ne sont pas traitées indépendamment. Ils montrent que les résidus d'erreur sur la base de test présentent une densité spectrale de pente quasi nulle à l'origine ainsi qu'une fonction d'auto-covariance "plate". Selon toute apparence, les résidus d'erreur sont indépendants.

Prévisions financières - A titre d'information, d'autres expérimentations plus récentes et pour lesquelles les échelles ne sont pas traitées indépendamment les unes des autres, ont été menées sur des séries financières européennes (actions et indices) échantillonnées toute les 15 minutes avec des réseaux de neurones. Une illustration de la prévision du taux d'accroissement de l'indexe S&P500 sur un horizon de 15 minutes est affichée en figures 6.6 et 6.7. Le lecteur intéressé est invité à consulter [AUS 98b].

6.4.4. *Conclusion et perspectives*

La pré-traitement des données permet de palier le problème du comportement "oublieux" des réseaux de neurones grâce à l'utilisation d'une communauté de réseaux opérant sur des échelles temps-fréquence distinctes. La série originale (potentiellement non-stationnaire) est décomposée en une hiérarchie de signaux de détails stationnaires exhibant une mémoire à courte portée, à laquelle s'ajoute un signal résiduel basse fréquence. Le signal original s'exprime comme la somme des signaux de détails auquel s'ajoute le signal résiduel. Chaque échelle est traitée individuellement et indépendamment des autres par un réseau de neurone.

Cette méthode a été brièvement illustrée sur une trace HTTP. Nos expériences sur des données environnementales [AUS 97], télécom [AUS 98c] et financières [AUS 98b] ont montré que les prédictions fournies par cette méthode hybride présentent typiquement un variance plus élevée que celle obtenue par une méthode directe fondée sur l'artifice de la fenêtre temporelle. De plus le modèle à capturer la dépendance à long terme car les résidus d'erreur sont indépendants.

Depuis septembre 2002, une collaboration avec Patrice Abry (Lab. de Physique, ENS Lyon) spécialiste des lois d'échelles, et Pierre Chainais (LIMOS) est menée dans ce sens pour caractériser et prédire le comportement du télé-trafic. Cette collaboration s'inscrit dans le cadre de l'Action Spécifique 'Métrologie Internet" du CNRS qui a débutté fin 2002. Un certains nombres de questions en suspens seront abordées :

- Quelle ondelette mère choisir ? Quels sont les critères de choix ? De manière équivalente, quels coefficients choisir pour les bans de filtres ?
- Comment traiter les effets de bord ?
- Une décomposition dédiée aux signaux discrets est-elle préférable ?
- Quelles sont les conséquences de l'affectation initiale $c_{0,k} = x_k$?
- Comment exploiter au mieux la redondance de l'information dans l'algorithme "à trous" pour prévoir les coefficients.

La prédiction à une seconde du trafic HTTP pourrait aider à la gestion des caches (*caching and prefetching*) [CRO 96]. La simulation pourrait également permettre d'engendrer des suites synthétiques utiles aux modèles d'analyse de performances. La qualité de services des connexions a un grand impact sur les performances des applications distribuées. Par exemple, FTP, Gopher et le Web souffrent de temps de réponse excessifs au moindre début de congestion du réseau. Pour ces documents, le temps de transfert est directement proportionnel à la bande passante de la connexion. Si la demande était connu un peu en avance, les flux pourraient être régulés de manière appropriée afin de réduire les temps de transfert. C'est le principe du routage adaptatif. Certains auteurs

proposent des mécanismes de contrôle de congestion au niveau application grâce à une estimation future continûement réactualisée de la bande-passante entre le client et le serveur [CRO 96].

13

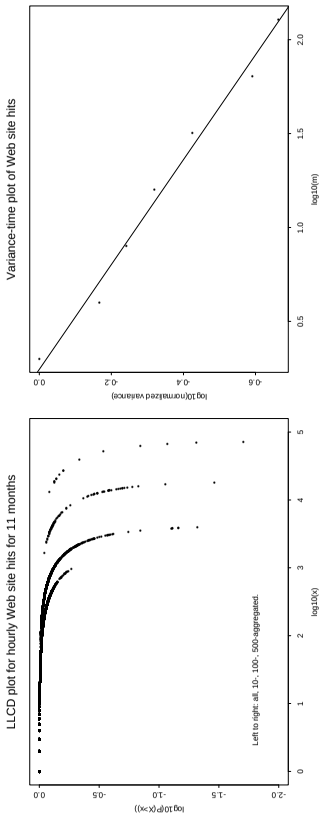


Figure 1: Variance-time plot and LLCD plot of hourly-averaged number of hits.

Figure 6.1. *Serveur HTTP Ulster : Graphe temps-variance et graphe LLCD*

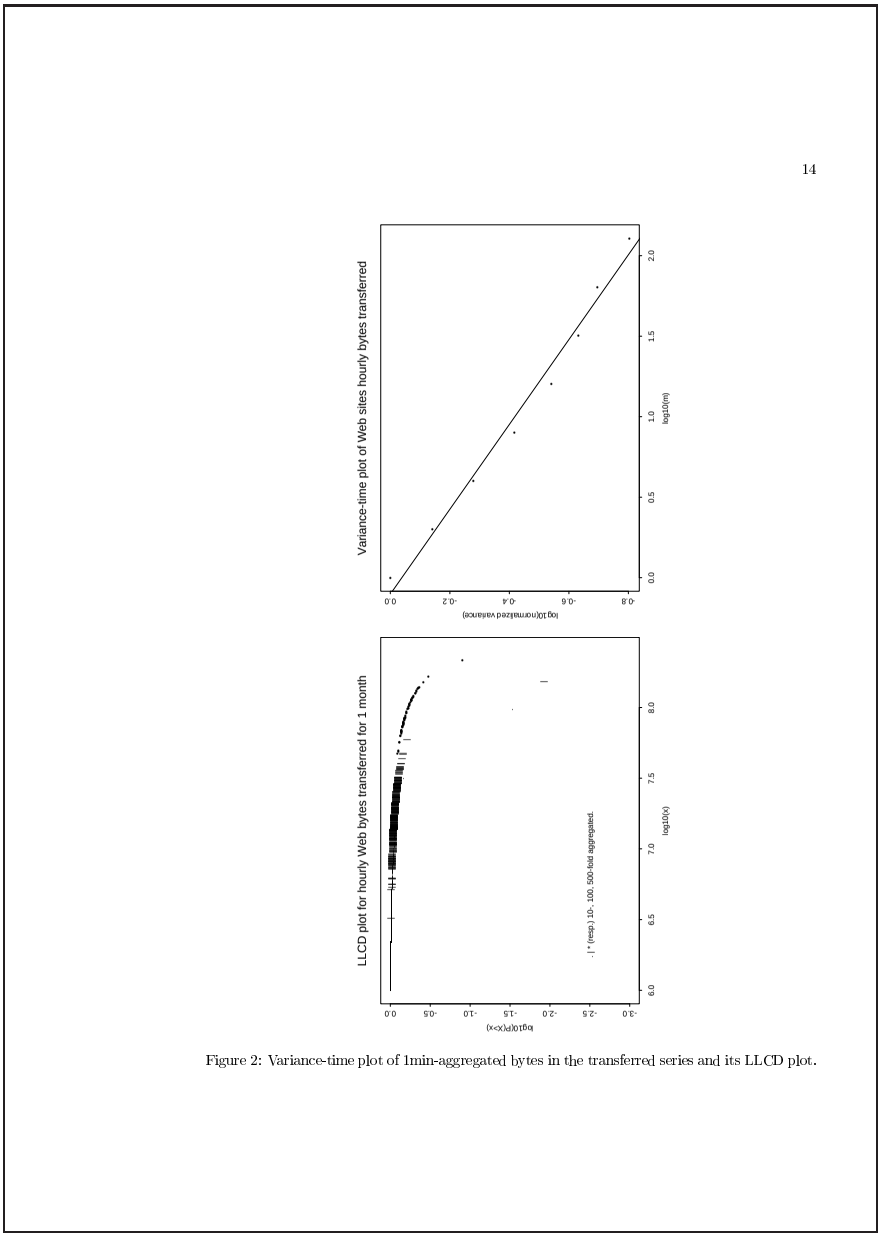


Figure 2: Variance-time plot of 1min-aggregated bytes in the transferred series and its LLCD plot.

Figure 6.2. *Serveur HTTP ESO : Graphe temps-variance et graphe LLCD*

15

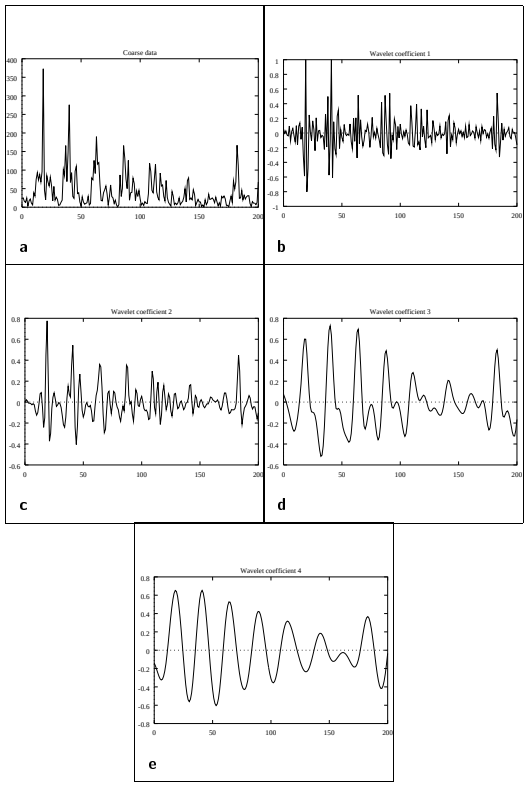


Figure 3: Average number of hits over successive hours is shown in the upper left corner, followed (from top to bottom) by 4 wavelet coefficients rescaled into the unit interval.

Figure 6.3. *Trafic HTTP. En haut à gauche le signal. Ensuite, illustration de la décomposition à trous pour* $J = 4$. c_4 *est omis.*

18

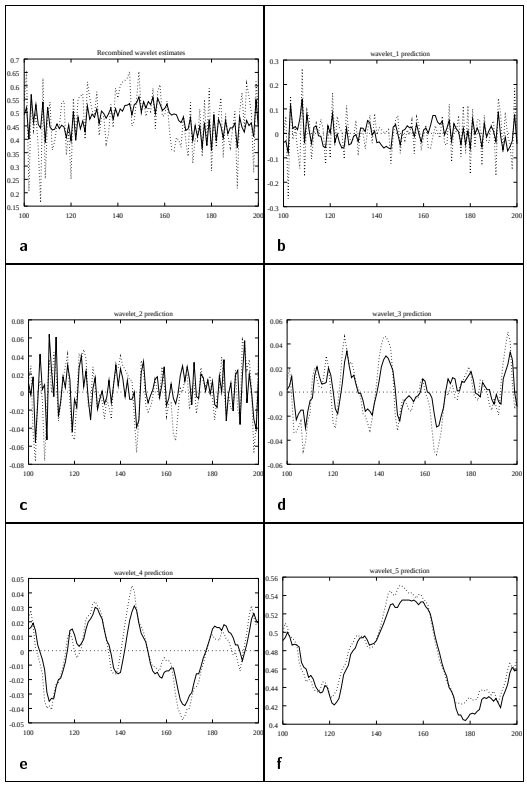


Figure 6: Individual wavelet coefficient on the prediction set forecast by DRNN models. The upper left plots shows the reconstructed forecasts from the 5 remaining wavelets coefficients. Note the change in scale.

Figure 6.4. *Trafic HTTP prédit sur un horizon de 1mn. En haut à gauche, le trafic en pointillé ainsi que la prévision finale en trait plein. Ensuite, viennent les w_j et leur prévision respective (en trait plein) jusqu'à c_4 en bas à droite.*

19

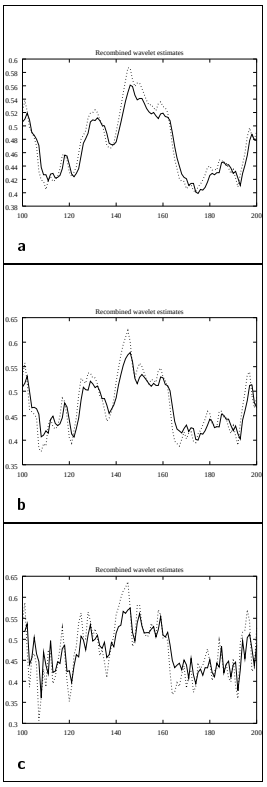


Figure 7: The forecast is gradually reconstructed from the marginal forecasts. From top to bottom: $w_4 + w_5$, $w_3 + w_4 + w_5$, $w_2 + w_3 + w_4 + w_5$. Contribution of errors at distinct time scales are clearly shown.

Figure 6.5. *Trafic HTTP : de haut en bas, les prévisions à 1 minute (en trait plein) recombinées $c_4 + w_4$, $c_4 + w_4 + w_3$, $c_4 + w_4 + w_3 + w_2$ en fonction de leur valeur cible. La contribution des erreurs de chaque échelle est visible.*

16

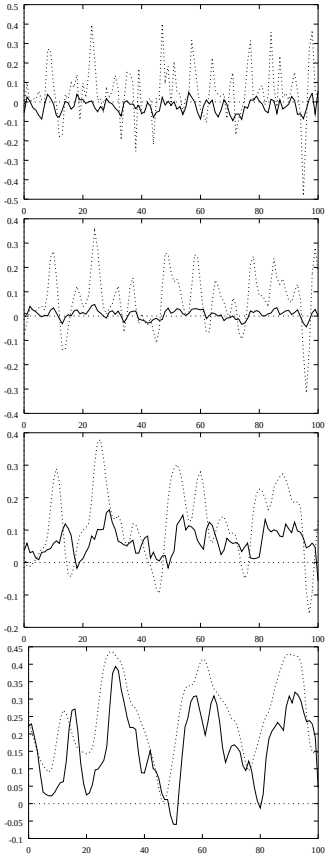


Figure 3: From top to bottom: 5-day ahead forecasts for w1, w2, w3 and w4.

Figure 6.6. *S&P500 : de haut en bas, les prévisions (en trait plein) des w_j pour $j = 1, \ldots, 4$ de la séries des taux d'accroissement : [x(t+5) - x(t)]/ x(t), où x(t)= S&P500(t), la valeur à la clôture le jour t.*

17

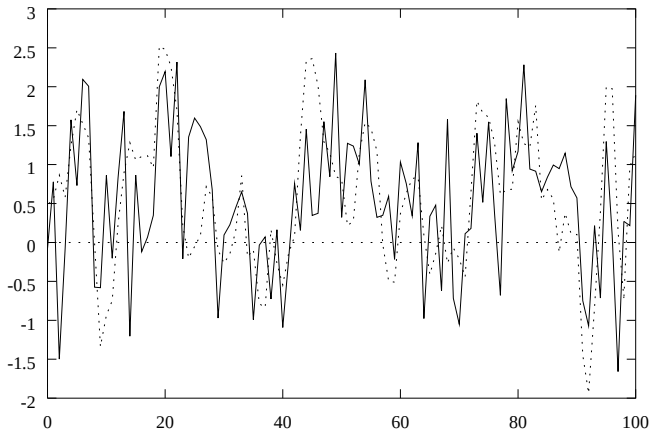


Figure 4: RDP forecasts (in plain line) vs. the true series over the first 100 days in the test set after the individual wavelet forecasts were recombined.

Figure 6.7. *S&P500 : la prévision finale (en trait plein) du taux d'accroissement [x(t+5) - x(t)]/ x(t), où x(t)= S&P500(t), la valeur à la clôture le jour t sur une portion de la base de test.*

Chapitre 7

PREVISION DE LA QUALITE DE SERVICE DANS LES RESEAUX TELECOM

7.1. Introduction

Ce chapitre fait état des derniers développements d'un projet au long court dédié à la gestion des ressources dans un réseau télécom multiservice, initié par Erol Gelenbe et moi-même au milieu des années 90 [AUS 94b], et poursuivi par plusieurs stagiaires de DEA successifs [AUS 94b, AUS 99c] au LIMOS. Ce travail est actuellement poursuivi par Antoine Mahul (en thèse avec moi au LIMOS) dans le cadre du projet RNRT OPIUM (Optimisation de la Planification des Infrastructures des réseaUx Mobiles). Ce projet, dont la partie routage incombe au LIMOS, vise à offrir une solution intégrée pour la planification et l'optimisation de réseaux de télécommunications mobiles. Le travail d'Antoine Mahul vise *in fine* à substituer à la formule M/M/1 classique, dans le code de l'optimiseur, un réseau de neurones entraîné par simulation pour prédire la QoS en chaque noeud en termes de délai de de perte.

La théorie des files d'attente s'applique avec succès à la modélisation du réseau téléphonique parce que les flots de trafic acheminés vers un commutateur peuvent être représentés fidèlement par un processus de Poisson en raison de l'indépendance des trafics[1]. Néanmoins, les files d'attente qui s'établissent dans les files d'attente ont une nature statistique bien différente (au niveau appel) de celles du trafic téléphonique. Parce que les données sont transmises en *rafale*, les arrivées successives sont fortement corrélées et ne peuvent plus par conséquent

[1] Théorème de superposition : la distribution asymptotique obtenue par superposition de N arrivées i.i.d. de taux $\lambda_i \to 0$ et $\sum_N \lambda_i \to$ constante, est une loi de Poisson.

être modélisées par une processus de Poisson. Or les méthodes algorithmiques de planification, de routage, ou de contrôle d'accès des réseaux télécom (e.g. dimensionnement) reposent encore trop souvent sur une estimation grossière des délais de commutation selon la formule standard des files M/M/1 : $(1/\lambda) \cdot \rho/(1-\rho)$, où λ est le taux d'arrivée et ρ désigne le taux d'occupation de la file[2]. Aussi, notre intention est d'inaugurer une approche nouvelle visant *in fine* à substituer à la formule M/M/1, un réseau de neurones en charge de prédire la QoS en termes de délai de transmission, de taux de perte, et de délai en *régime stationnaire*. Une fois l'apprentissage effectué à l'aide d'un simulateur, le RN fournit instantanément la prédiction au vu de quelques descripteurs du trafic incident. La méthode d'apprentissage combine plusieurs réseaux de neurones indépendants distribués sur les commutateurs et inter-connectés via les lignes de communication.

Les RN ont été employés dans la littérature pour : 1) la prédiction court-terme du trafic [HAB 96, LEE 00], par une technique de décomposition en ondelettes [AUS 97, MUR 97, AUS 98c, SOL 00], 2) identifier les paramètres du modèle de source (e.g. processus MMPP) [CAS 98], 3) une caractérisation transitoire du *leaky bucket* [CLé 98], 4) l'estimation du taux de perte d'un multiplexeur pour réaliser un contrôle d' admission (Call Admission Control, CAC) [AUS 94b, NOR 95, HIR 95, SOH 01], le routage et l'allocation dynamique de bande-passante [BOL 98, MOH 95].

Le travail présenté ici fait état des derniers développements d'un projet au long court dédié à la gestion des ressources dans un réseau télécom multiservice dont l'origine remonte au milieu des années 90 [AUS 94b, AUS 99c, AUS 00c, MAH 02]. Les applications numériques présentées dans ce chapitre ont été réalisées par Antoine Mahul.

7.2. Les descripteurs de trafic

Il est essentiel, dans un réseau véhiculant des trafics sporadiques, d'être en mesure de caractériser le processus de génération des cellules pendant un appel, de façon à dimensionner le réseau. Néanmoins, il est excessivement difficile de modéliser, et *a fortiori* de prédire, le comportement statistique d'une kyrielle de services, certains encore méconnus. Il y a eu un travail considérable effectué ces dernières années sur les modèles de source et sur le choix adéquat de descripteurs de trafic permettant d'identifier les propriétés essentielles du trafic et de développer de modèles d'évaluation de performance.

[2] C'est aussi la formule en régime stationnaire du temps de réponse d'une file M/G/1/PS (processor sharing) de temps de service moyen $1/\mu$.

Il existe à ce jour plusieurs choix possibles pour décrire un trafic sporadique. L'ensemble des descripteurs doit être très restreint pour permettre un contrôle simple et efficace, tout en représentant fidèlement les propriété des flots de données. De surcroît, il doit être possible de déduire les caractéristiques de la superposition de trafics élémentaires à l'aide des paramètres individuels. Avec ces contraintes à l'esprit, nous nous limitons délibérément à deux grandeurs statistiques fondamentales

- le taux moyen d'arrivée λ,
- le débit pic, p,
- le carré du coefficient de variation de la distribution des inter-arrivées, $Cv^2 = var(X)/E^2(X)$, où X est le temps d'inter-arrivée des paquet.

Le coefficient de variation de la distribution des inter-arrivées est une mesure standard et intuitive de la variabilité. Pour une loi exponentielle par exemple, $Cv^2 = 1$, valeur de référence. Pour une loi d'Erlang, somme de r variables exponentielles, $Cv^2 = 1/r$. Des durées de traitement hétérogènes peuvent engendrer des Cv^2 très importants, signe d'un risque important de dégradation des performances. A titre d'exemple, la proportion de temps passée en attente sur le temps passée dans le système (attente +service) est une fonction affine de Cv^2 à ρ fixé. Du reste, on parle de loi *hypo* (e.g., loi d'Erlang) et *hyper*-exponentielle suivant le signe de $Cv^2 - 1$.

Par ailleurs, le carré du coefficient de variation de la distribution des inter-arrivées est commode dans notre contexte car il peut être relié à l'*index de dispersion* $I(t)$ représentant un descripteur synthétique approprié lorsque le flot de cellules est vu comme un processus ponctuel stationnaire [COM 96]. Soit $t = 0$ un instant arbitraire, et $N(0,t)$ le nombre des arrivées durant la période $[0,t]$, alors $I(t)$ est définie par

$$I(t) = \frac{\mathrm{var}(N(0,t))}{\mathrm{E}(N(0,t))}. \tag{7.1}$$

Pour un processus de Poisson, $I(t) = 1, \forall t$. De plus, pour une processus de renouvellement $I(t)$ n'est autre que le carré du coefficient de variation de la distribution des inter-arrivées, C_v^2. Plus généralement, il est démontré [COM 96] que $I(t)$ vérifie

$$\lim_{t\to\infty} I(t) = \frac{\mathrm{var}(\tau)}{\mathrm{E}^2(\tau)} = C_v^2, \tag{7.2}$$

où τ est le temps d'inter-arrivée des cellules. En conséquence, on peut facilement calculer et le taux moyen d'arrivée et la variance du nombre des arrivées du flot agrégé en sommant les moyennes et les variances individuelles et déduire le coefficient de variation des inter-arrivées correspondant. Considérons

la superposition de n sources indépendantes et $N_i(0,t)$ le nombre de cellules émises par la source i durant la période $[0,t]$. Le nombre des arrivées pour le flot agrégé est

$$N^s(0,t) = \sum_{i=1}^{n} N_i(0,t), \tag{7.3}$$

et donc

$$\begin{aligned} \mathrm{E}[N^s(0,t)] &= \sum_{i=1}^{n} \mathrm{E}[N_i(0,t)] \\ \mathrm{var}[N^s(0,t)] &= \sum_{i=1}^{n} \mathrm{var}[N_i(0,t)]. \end{aligned} \tag{7.4}$$

Posons

$$\lambda_i = \lim_{t\to\infty} \frac{1}{t}\mathrm{E}[N_i(0,t)]. \tag{7.5}$$

Lorsque le trafic en entrée est un processus de *renouvellement*, il existe une formule analytique d'agrégation des Cv^2 [GEL 76] :

$$Cv_a^2 \left[1 + 2\sum_{j=1}^{\infty} \rho_j\right] = \frac{1}{\lambda_a} \sum_{i=1}^{N} \lambda_i Cv_i^2. \tag{7.6}$$

Le terme de droite est la valeur asymptotique de l'index de dispersion [GEL 76, GUS 91], où ρ_j est le coefficient d'autocorrelation de délai j du trafic agrégé. L'estimation de ces coefficients étant difficilement envisageable d'un point de vue technique, ils sont délibérément omis. Ainsi les descripteurs, $(\lambda_a,\, p_a,\, Cv_a^2)$, du trafic agrégé résultant du multiplexage de N flots *indépendants* et *stationnaires* caractérisés par $\left(\lambda_i,\, p_i,\, Cv_i^2\right)_{1\leqslant i\leqslant N}$, sont approximés par **formule d'agrégation** [AUS 94b] suivante

$$\left(\sum_{i=1}^{N} \lambda_i, \quad \sum_{i=1}^{N} p_i, \quad \frac{1}{\lambda_a}\sum_{i=1}^{N} \lambda_i Cv_i^2\right) \tag{7.7}$$

Lorsque le trafic agrégé n'est pas un processus de renouvellement, le descripteur de *burstiness* dans Eq. (7.7) est erroné ; il rend compte malgré tout de

la variabilité du trafic. Il faut noter à ce titre que la valeur maximale de $I(t)$ pour $t \geq 0$ est considérée comme une alternative à C_a^2 [COM 96] et dans de nombreux cas[3] est atteint pour $t \to \infty$. A la lumière de ces observations, C_a^2 est considéré comme un candidat plausible pour la description synthétique des trafics [COM 96].

Le modèle de source OnOff

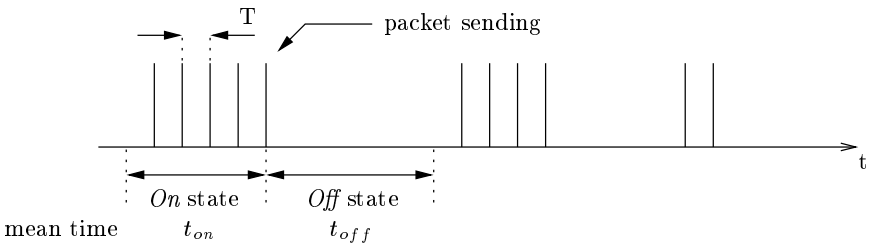


Figure 7.1. *Trafic généré par une source OnOff.*

Dans nos expérimentations, les sources sont représentées par un processus OnOff, un scénario simple et commun pour la modélisation de la voix. La source OnOff alterne les états *On* et *Off* de durée exponentielle, de moyenne t_{on} et t_{off} secondes respectivement (Fig. 7.1). Dans l'état *On*, les paquets sont émis toutes les $T = 1/\lambda_{on}$ secondes tandis que l'état *Off* correspond à une période de silence. Les descripteurs de la source OnOff s'obtiennent analytiquement [COM 96] :

$$\lambda = \frac{\lambda_{on} t_{on}}{(t_{on} + t_{off})}, \qquad p = \lambda_{on}, \qquad Cv^2 = \frac{2\lambda_{on} t_{on} - 1}{(1 + t_{on}/t_{off})^2} \tag{7.8}$$

7.3. Réseaux de neurones distribués

On cherche à estimer localement la QoS en chaque file d'attente en fonction de descripteurs du trafic en entrée. Il s'agit typiquement d'un problème de régression (non-linéaire) : soit $d = (\lambda, p, Cv^2)$ les caractéristiques du trafic en entrée et $q = (\bar{N}, Log(CLR))$ le critère local de QoS, on cherche une

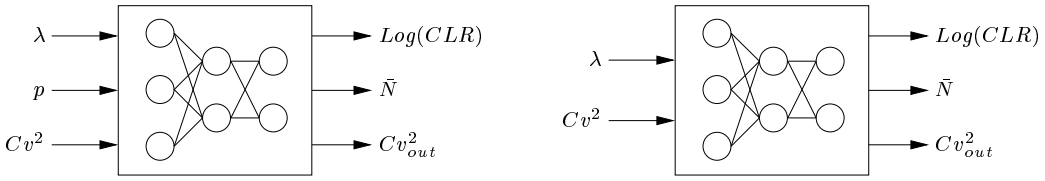


Figure 7.2. *Détails des entrées/sorties des MLP pour une file alimentée : (a) directement par une source externe, (b) par une commutateur distant.*

approximation de la relation $d \rightarrow q$. La relation étant *statique*, un MLP sera employé.

Il est nécessaire de propager ces descripteurs de file en file ; le débit moyen du trafic de sortie est $\lambda_{out} = \lambda_{in}(1 - CLR) \simeq \lambda_{in}$ car $CLR \ll 1$. Par ailleurs, le débit pic des files alimentées par un serveur distant est constant. Il n'est donc pas propagé. Seul, le coefficient de *burstiness*, Cv^2, est estimé en sortie des files. L'apprentissage est réalisé avec des données obtenues grâce à un simulateur à événements discrets. Seules les situations de charge critique proche de la congestion ont été sélectionnées. Les détails relatifs à l'apprentissage (e.g., sélection du MLP, l'algorithme d'optimisation des poids, le choix des paramètres d'apprentissage, la technique de validation) sont fournis dans [MAH 02].

7.4. Expérimentations

Quelques expérimentations ont été menées dans [AUS 99c] avec des files mono-serveur FIFO individuelles ainsi que des files positionnées en série et en parallèle [AUS 00c, MAH 02].

[3] par exemple lorsque les durées des *burst* et des silences ont une distribution exponentielle, ou lorsque la durée des *burst* est déterministe et la durée des silences est de distribution exponentielle.

7.4.1. *File unique*

OnOff/D/1/c

Considérons une file OnOff/D/1/c de capacité limitée ($c = 100$) avec un serveur déterministe et une politique de service FIFO. Le trafic en entrée est OnOff. Les entrées sont le débit moyen, le débit pic et Cv^2 calculés puis injectés en entrée du NN. Les paramètres mesurés sont $\bar{N}$, CLR et Cv^2_{out}. Le taux de service est 13000 paquets/ms. Les paramètres de source sont tirés aléatoirement entre 15000 et 17000 paquets/ms pour le débit pic, entre 6500 et 13000 cell/ms pour le débit moyen, entre 10^{-3} et 10^{-2} ms pour t_{on}. Avec ces valeurs, le CLR obtenu par simulation varie entre 10^{-4} et 10^{-2}. La base d'apprentissage est constituée de 456 exemples et 75 exemples en test. Le MLP sélectionné comporte 15 neurones cachés.

Les prédictions de $\bar{N}$ et Cv^2_{out} sont fiables. La NMSE sur la base de test pour $\bar{N}$ vaut 5.74×10^{-4}, 5.27×10^{-5} pour Cv^2_{out} et 1.36×10^{-3} pour le CLR. La qualité de la prédiction de dégrade quelque peu pour de faibles valeurs de CLR en raison du manque d'exemples correspondants en apprentissage.

OnOff/OnOff/1/c

Le serveur est cette fois de type OnOff, i.e. le serveur est inhibé pendant une durée de loi exponentielle et le service est déterministe dans l'état *On*. L'idée sous-jacente est de considérer un serveur partageant plusieurs buffeurs dont les transitions se produisent à des instants aléatoires selon la priorité des paquets en tête de file. A l'issue du service, le serveur passe dans l'état *Off* avec une probabilité p^s. Il reste dans cet état durant T^s, une durée constante de service. L'activation et les périodes de sommeil suivent une loi géométrique de paramètres p^s et q^s respectivement. Le trafic en entrée est OnOff. Le trafic d'entrée et l'état du serveur influent directement sur la valeur du CLR. Dans l'état *On*, la seule transition possible est vers l'état *Off* à la fin du service. Aussi, le débit d'entrée influe fortement sur le temps effectif d'activité du serveur.

Le taux de service moyen est de 13000 paquets/ms, $p^s = 0.7$ et $q^s = 0.4$, le débit pic varie entre 13000 et 20000 paquets/ms, le débit moyen en entrée varie entre 6500 et 13000 paquets/ms, et t_{on} varie entre 10^{-4} et 10^{-2} ms. Seuls les exemples de CLR dans l'intervalle $[10^{-4}, 10^{-2}]$ ont été sélectionnés pour l'apprentissage. Un MLP 3/15/3 a été employé. Ici encore, les prédictions de $\bar{N}$ et Cv^2_{out} sont très fiables. La prédiction du CLR est légèrement en deçà du cas précédent. La NMSE sur la base de test du $\bar{N}$ est de 3.55×10^{-3}, 1.21×10^{-4}, pour le Cv^2_{out} et 8.61×10^{-3} pour le CLR. La prédiction se dégrade pour de

faibles valeurs de CLR. Pour autant, le taux d'occupation, le taux de perte et la caractérisation du trafic en sortie sont jugés bonnes.

Multi-OnOff/D/1/c

Dans ce paragraphe, plusieurs sources OnOff indépendantes sont multiplexées. Les descripteurs agrégés sont considérés comme suffisants pour caractériser le trafic offert en entrée. Fixons les taux d'arrivée des sources à λ, et le débit pic à p, tels que

$$N\lambda \leq \mu \leq Np \tag{7.9}$$

Le nombre et les caractéristiques des sources OnOff, $N \in [10, 100]$, $\mu = 13000$ paquets/ms et $N\lambda \leq \mu \leq Np$, sont variés aléatoirement. Chaque instanciation exige une simulation pour estimer le CLR, et seules les situations pour lesquelles $10^{-4} < CLR < 10^{-2}$ ont été sélectionnées pour l'apprentissage. Remarquez que le Cv^2 agrégé est égal au Cv^2 des sources individuelles - il est donc indépendant du nombre de sources -, car toutes les sources OnOff sont identiques.

Un MLP 3/15/3 NN a été entraîné sur 533 exemples, et validé sur 55 exemples. 10000 époques d'apprentissage ont été nécessaires. Les valeurs de NMSE sur la base de test sont 1.28×10^{-2} pour $\bar{N}$, 2.71×10^{-4} pour Cv^2_{out} et 1.05×10^{-2} pour CLR. Ces résultats sont moins bons mais toujours acceptables. Par exemple, la partie entière de $Log(CLR)$ est exacte et la prédiction de Cv^2_{out} est acceptable. C'est encourageant car le Cv^2_{out} est transmis en entrée d'un MLP distant le long du chemin.

7.4.2. *Files en tandem*

L'objet de ce paragraphe est d'évaluer l'estimation de la QoS lorsque les descripteurs sont propagés de file en file le long d'une chemin. Un chemin de communication est constitué de plusieurs files en tandem, voir Fig. 7.3. Plusieurs scenarii sont envisagés : une source OnOff ainsi qu'une superposition de sources OnOff homogènes et identiques. Les apprentissages des MLP sont réalisés indépendamment les uns des autres grâce au simulateur.

Nous comparons aussi les performances obtenues lorsque le C_v^2 en entrée du MLP est remplacé par sa *vraie* valeur obtenue par simulation. Ce faisant, il est possible de de jauger la dégradation des prédictions le long du chemin de communication.

Source OnOff unique

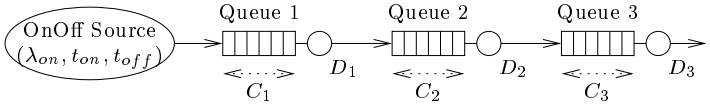


Figure 7.3. *3 files consécutives. Le trafic d'entrée est OnOff.*

Le trafic d'entrée est OnOff. Les files 1, 2 et 3 ont des serveurs déterministes de taux et de capacité : $(100, 13000)$; $(80, 12000)$ et $(20, 12800)$ respectivement. Les paramètres de la source sont tirés aléatoirement dans les intervalles : $\lambda \in [6500, 11000]$, $\lambda_{on} \in [15000, 17000]$, $t_{on} \in [10^{-3}, 10^{-2}]$. La file 1 est représentée par un MLP de type Fig. 7.2(a) ; Les files 2 et 3 le sont par des MLP du type Fig. 7.2(b).

Chaque MLP est constitué de 15 neurones cachés. Les MLP ont été entraînés sur 500 exemples, et validés sur 100 exemples. Les résultats sont illustrés Table 7.1. Un *distinguo* est fait entre descripteurs mesurés et descripteurs estimées en entrée des MLP pour jauger l'impact des erreurs sur la propagation du Cv^2. Les NMSE avoisinent obtenues les 10^{-2} en terme de nombre moyen de clients en attente. L'ordre de grandeur du CLR est correctement anticipé. On observe également de très bons résultats pour le Cv^2_{out}. On observe du reste un lissage du trafic de file en file au vu des performances qui s'améliorent entre la file 2 et 3.

		Cv^2_{out}	N	$Log(CLR)$
NN 1		4.85×10^{-5}	2.55×10^{-3}	5.97×10^{-3}
NN 2	entrées mesurées	3.11×10^{-5}	5.93×10^{-3}	1.31×10^{-2}
	entrées estimées	3.37×10^{-4}	6.12×10^{-3}	1.26×10^{-2}
NN 3	entrées mesurées	2.10×10^{-5}	5.19×10^{-3}	7.75×10^{-3}
	entrées estimées	4.83×10^{-4}	5.34×10^{-3}	7.15×10^{-2}

Tableau 7.1. *NMSE sur la base de test. 3 files en tandem alimentées par une source OnOff unique. Une distinction est faite entre descripteurs mesurés et descripteurs estimées en entrée des MLP pour jauger l'impact des erreurs sur la propagation du Cv^2.*

N sources OnOff homogènes

N sources OnOff homogènes (partageant les même paramètres) ont été multiplexées. Les paramètres de source sont tirés aléatoirement : débit moyen du

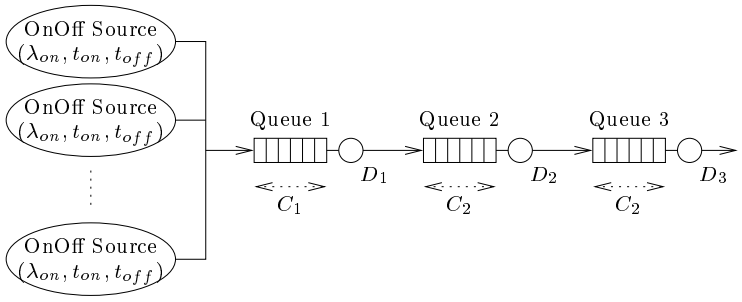


Figure 7.4. *3 files consécutives. Le trafic d'entrée est constitué de 5 sources OnOff homogène.*

trafic agrégé $\lambda \in [6000, 11000]$, débit moyen de la k-ème source, $\lambda^k = \lambda_{tot}/N$, $\lambda^k_{on} \in [5000, 8000]$, $t^k_{on} \in [10^{-3}, 10^{-2}]$, et $N = 5$. Les files 1, 2 et 3 ont des serveurs déterministes de taux et de capacité : (100, 16500) ; (60, 15000) et (40, 14000) respectivement. Avec ces valeurs, le CLR est majoritairement dans l'intervalle $[10^{-4}, 10^{-2}]$. Chaque MLP est constitué de 15 neurones cachés. Les MLP ont été entraînés sur 500 exemples, et validés sur 100 exemples. Les résultats sont illustrés Table 7.2. On observe de bons résultats hormis une légère dégradation pour les files 2 et 3 en terme de CLR. Cette dégradation disparaît dès lors que les entrées sont les valeurs mesurées directement sur le trafic offert et non propagés de file en file.

		Cv^2_{out}	N	$Log(CLR)$
NN 1		1.68×10^{-4}	1.50×10^{-2}	1.69×10^{-2}
NN 2	entrées mesurées	5.98×10^{-4}	2.44×10^{-2}	1.21×10^{-1}
	entrées estimées	1.32×10^{-3}	2.49×10^{-2}	1.18×10^{-1}
NN 3	entrées mesurées	6.71×10^{-4}	1.61×10^{-2}	5.21×10^{-2}
	entrées estimées	3.48×10^{-3}	1.74×10^{-2}	5.33×10^{-2}

Tableau 7.2. *NMSE sur la base de test. 3 files en tandem alimentées par une superposition de sources OnOff homogènes. Une distinction est faite entre descripteurs mesurés et descripteurs estimées en entrée des MLP pour jauger l'impact des erreurs sur la propagation du Cv^2.*

La performance globale en termes de $Log(CLR)$ et $\bar{N}$ est très satisfaisante. On observe toutefois une augmentation de l'erreur liée à l'estimation de Cv^2_{out} lorsque le Cv^2_{in} n'est pas pas mesuré, mais estimé par Eq. (7.7). Etant donné que les termes d'autocorrelation ont été omis dans la formule approximation Eq. (7.6), seul le moment d'ordre 2 des inter-arrivées caractérise la *burstiness*. C'est l'une des principales limitations du modèle.

Deux sources hétérogènes

Considérons deux sources indépendantes et hétérogènes OnOff. Les files 1, 2 et 3 ont des serveurs déterministes de taux et de capacité : (100, 13000) ; (60, 12000) et (40, 12500). Le CLR se situe entre 10^{-4} et 10^{-2}. Les paramètres du trafic sont choisis aléatoirement dans les intervalles : $\lambda \in [6500, 11000]$; λ^1_{on} et $\lambda^2_{on} \in [8000, 9000]$; t^1_{on} et $t^2_{on} \in [10^{-3}, 10^{-2}]$. Le débit moyen des source 1 et 2 sont : $\lambda^1 \in [3000, \min(\lambda, 8000)]$, et $\lambda^2 = \lambda - \lambda^1$.

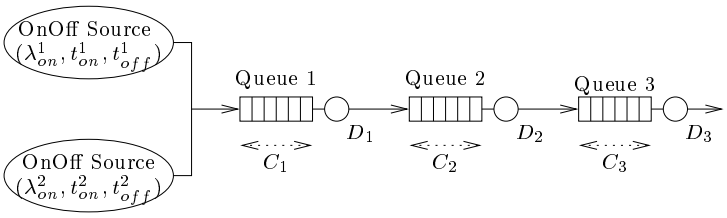


Figure 7.5. *3 files consécutives alimentées par 2 sources indépendantes et hétérogènes OnOff.*

	Cv^2_{out}	N	$Log(CLR)$
NN 1	1.04×10^{-3}	7.65×10^{-2}	2.44×10^{-1}
NN 2	2.86×10^{-3}	6.19×10^{-2}	1.25×10^{-1}
NN 3	2.38×10^{-3}	3.14×10^{-2}	5.78×10^{-2}

Tableau 7.3. *NMSE sur la base de test avec 2 sources indépendantes et hétérogènes OnOff.*

Les résultats sont affichés Table 7.3. Au regard des valeurs de NMSE, on observe une lente dégradation de l'erreur de prédiction de $Log(CLR)$, lorsqu'on se déplace de file en file. Les entrées sont estimées et non mesurées. Sur cet exemple simple de 3 files en tandem, il apparaît que les descripteurs (λ, p, Cv^2) définis au paragraphe 7.2 ne suffisent pas à caractériser parfaitement le trafic agrégé, même si l'estimation de Cv^2_{out} et $\bar{N}$ demeure fiable.

7.4.3. *Deux files en parallèle alimentant une troisième*

Considérons à présent deux files en parallèle alimentant une troisième, Fig. 7.6. Le trafic en entrée est OnOff. Il s'agit d'illustrer la pertinence de la formule de recombinaison des Cv^2. Les 2 sources OnOff sont indépendantes et hétérogènes. Leurs paramètres sont choisis aléatoirement comme au paragraphe 7.4.2.

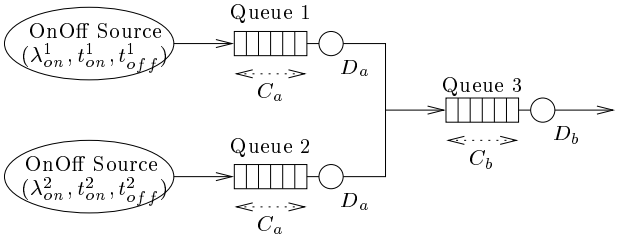


Figure 7.6. *Deux files en parallèle alimentant une troisième. Le trafic en entrée est OnOff.*

Les files 1, 2 et 3 ont des serveurs déterministes de taux et de capacité : (100, 13000) ; (100, 13000) et (120, 22000). Afin de compenser le manque d'information évoqué au paragraphe 7.4.2, on replace le débit pic par la durée moyenne d'un burst, t_{burst}. A la différence du débit pic, la durée moyenne d'un burst caractérise uniquement le trafic. La formule de recombinaison :

$$t^a_{burst} = \left(\sum_{i=1}^{N} \frac{1}{t^i_{burst}} \right)^{-1} \tag{7.10}$$

Pour un trafic OnOff, on a $t_{burst} = t_{on}$. Ainsi, les entrées du MLP sont $(\lambda, t_{burst}, Cv^2)$ et les sorties $(Cv^2_{out}, t_{burst}, Log(CLR))$. Le descripteur en entrée, Cv^2, de la file 3 est approximé par Eq. (7.7). Les résultats sont affichés Table 7.4.

	Cv^2_{out}	t_{burst}	$Log(CLR)$
NN 1 & 2	1.64×10^{-4}	5.57×10^{-4}	2.88×10^{-3}
NN 3	6.54×10^{-3}	6.94×10^{-3}	2.85×10^{-2}

Tableau 7.4. *NMSE sur la base de test. Les files 1 et 2 alimentent la file 3.*

7.5. Discussion et perspectives

Des estimations fiables de $\bar{N}$, CLR et Cv^2_{out} ont été obtenus avec des MLP standards d'une quinzaine de neurones. Cv^2_{out} est propagé le long du chemin de communication et réestimé par les MLP distribués sur les files d'attente. Cette approche offre potentiellement un champs d'application au routage et au contrôle d'admission (CAC) basés sur des contraintes de QoS en termes de délai de bout en bout et de taux de perte *moyens*, dans les réseaux multiservice [AUS 94b].

Prenons l'exemple du contrôleur d'admission des appels. Ce dernier accepte ou rejette les appels selon la QoS requise par l'utilisateur et celle que le réseau peut offrir. Si le contrôleur fonctionnait de manière optimale, la congestion serait évitée dans la mesure où les trafics responsables de l'excédent de charge seraient simplement rejetés. Lorsqu'un appel est accepté, un *contrat* tacite est établi entre l'administrateur réseau et le souscripteur spécifiant les caractéristiques du flot de données et la qualité de service requise par la connexion. Le rôle du réseau à intégration de services est de convoyer des trafics sporadiques décrits par des variables statistiques. Aussi, le contrôleur doit être suffisamment flexible pour appréhender des trafics aux fluctuations aléatoires. Afin de délivrer la QoS promise dans la phase d'établissement des connexions, le contrôleur doit superviser l'état du réseau en chaque nœud et estimer le surplus de charge occasionné par la transmission de trafics supplémentaires. Puisque le temps moyen de transit à travers une file d'attente se déduit du nombre moyen de client à l'état stationnaire d'après le théorème de Little, il suffit de prédire le nombre moyen de paquets en attente dans les files lorsqu'un nouveau trafic est convoyé. Pour chaque nouvelle requête émise au contrôleur, celui-ci sélectionne un ensemble de routes possibles, et pour chacune d'entre elles, estime le temps de transfert moyen des cellules de bout-en-bout. La qualité de service sur la route au délai minimal est comparée à la qualité requise par la connexion.

Quelques critiques méritent d'être formulées toutefois. Primo, la méthode opère *off-line* et n'a pas été testée avec un trafic réel (possiblement non-stationnaire). Secundo, les descripteurs ne recèle aucune information sur la structure de corrélation du processus des arrivées (implicitement négligée dans la formule de recombinaison). C'est la grande faiblesse de cette approche : comment recombiner facilement les coefficients de corrélation ? D'autres descripteurs ont été proposés dans la littérature [NOR 95] pour tenir compte de la corrélation mais ils ne sont spécifiques à la source (idéale) OnOff. Pour finir, la méthode vise à caractériser un *régime permanent* des buffeurs, par opposition à transitoire. Ce n'est hélas pas suffisant pour garantir la QoS en terme de délai maximum de bout en bout par exemple. Une compréhension du comportement transitoire des files est nécessaire. Notons enfin que l'estimation de la *gigue* (variabilité du délai de bout-en-bout) n'a pas été évoquée.

7.6. Conclusion

Dans ce chapitre, nous nous sommes intéressés à l'application des réseaux de neurones classique (MLP) à un problème de prévision de la QoS dans les buffeurs. Les trafics sont décrits par des grandeurs statistiques sommaires. Nous avons examiné la modélisation de chaque file d'attente dans un nœud de commutation.

Cette approche vise *in fine* à substituer à la "formule de Kleinrock", un réseau de neurones en charge de prédire la QoS en termes de délai de transmission, de taux de perte, et de délai en *régime permanent*. Une fois l'apprentissage effectué à l'aide d'un simulateur, le MLP fournit instantanément la prédiction au vu de quelques descripteurs du trafic incident. La méthode d'apprentissage combine plusieurs réseaux de neurones indépendants distribués sur les commutateurs et inter-connectés via les lignes de communication. L'interêt immédiat de ce travail est d'insérer un MLP dans un code d'optimisation dédié au routage multiflots.

Chapitre 8

MODELE HYBRIDE CHAINE DE MARKOV CACHEE & MLP

8.1. Introduction

Ce chapitre présente un modèle auto-régressif non-linéaire à changement de régime markovien pour la segmentation de séries temporelles stationnaires par morceaux, grâce à un couplage entre une chaîne de Markov cachée (HMM) avec des réseaux connexionistes de type MLP. Ce type d'approche hybride et modulaire alliant les capacités d'approximation et de généralisation des réseaux connexionnistes et la capacité de modélisation des HMMs, a été employé avec grand succès au début des années 90 dans le domaine de la reconnaissance de la parole, e.g. discrimination d'un ensemble de locuteurs (voir par exemple [BEN 92, BEN 93, BEN 94c, BEN 95b]) mais également dans le domaine de la reconnaissance de l'écriture manuscrite cursive [GAR 96]. L'idée est de tirer avantage des mérites respectifs des réseaux connexionnistes et des HMM, pour concevoir un système plus performant que chacune des méthodes prise séparément.

La stationnarité est souvent un postulat sur lequel repose la conception de modèle pour la prédiction de séries temporelles. Toutefois, cette hypothèse n'est pas toujours vérifiée en pratique (e.g., consommation électrique, cours financiers, etc.). L'idée est de modéliser une série stationnaire par morceaux par un mélange d'experts régresseurs, chaque expert étant en charge de la prédiction dans son régime dynamique. Les transitions entre les états sont gouvernés par une chaîne de Markov homogène à espace d'état fini. L'application présentée dans ce chapitre ne traite que de la *segmentation* (i.e. partitionnement) non-supervisée d'une série chronologique unidimensionnelle, en associant à chaque état de la HMM, un expert prédicteur supervisé, ici un MLP. Le problème de

la prévision, à proprement parler, des séries stationnaires par morceaux n'est pas abordé ici.

L'apprentissage est traité comme un problème de maximisation de la vraisemblance ; l'algorithme EM [DEM 77] estime les paramètres du modèle, en particulier les paramètres des réseaux de neurones, les probabilités de transitions et la variance du bruit, en considérant les états internes comme des variables manquantes. A la différence des modèles dits multi-experts [JOR 94] ou "input/output HMM" (IOHMM) [BEN 95a, BEN 00b], les transitions sont indépendantes des entrées et accomplissent une partition nette de la série, ce qui autorise la spécialisation des experts dans leur dynamique. Les méthode est illustrée sur des données artificielles chaotiques et financières.

8.2. Experts prédicteurs

Nous supposons le lecteur familier avec les principes des HMM, et nous adoptons les notations en vigueur dans l'article (de référence) de Rabiner [RAB 89]. La présentation s'inspire de l'article récent de Kohlmorgen et al. [KOH 99].

Considérons un HMM où chaque état $i = 1, \ldots, M$ est associé à un expert prédicteur. Ce dernier prédit la future valeur $y_t = x_{t+\tau}$ d'une série temporelle $\{x_t\}$ ou une variable exogène, étant donne un vecteur de valeurs passées $\mathbf{x}_{t-d}^t = (x_{t-d}, \ldots, x_t)$ de longueur arbitraire, d est l'horizon de prédiction et τ est le paramètre de délai. Nous supposons que la variable cible, y_t, à chaque instant, t, est donnée par une fonction déterministe, $f_i(\mathbf{x}_{t-d}^t)$, où i est le régime dynamique courant, avec un bruit blanc additif gaussien, ϵ_t, tel que

$$y_t = f_i(\mathbf{x}_{t-d}^t) + \epsilon_t \tag{8.1}$$

L'erreur ϵ_t suit une loi normale de moyenne nulle et d'écart type inconnu, σ, indépendant de $\mathbf{x}_{t-d}^t$. La densité de probabilité conditionnelle de la variable cible de l' expert, i, s'écrit donc

$$p(y_t|\mathbf{x}_{t-d}^t, s_t = i) = \frac{1}{\sqrt{2\pi}\sigma_i} \mathrm{e}^{-(y_t - f_i(\mathbf{x}_{t-d}^t))^2/2\sigma_i^2} \tag{8.2}$$

Notons à ce stade que les σ_i seront ajustés au cours de l'apprentissage. Une densité de probabilité sera notée par p et une probabilité par P.

Pour calculer la vraisemblance du système, $\mathcal{L}$, il faut noter que le passé de la séquence à l'instant t est entièrement résumé par l'entrée courante, $\mathbf{x}_{t-d}^t$, et l'indice du régime dynamique courant, s_t. On peut écrire

$$\begin{aligned} p(y_t|s_t, \mathbf{x}_1^t) &= p(y_t|s_t, \mathbf{x}_{t-d}^t) \\ P(s_{t+1}|s_t, \mathbf{x}_1^t) &= P(s_{t+1}|s_t, \mathbf{x}_{t-d}^t) \end{aligned}$$

Le modèle auquel on arrive est un input/output HMM (IOHMM) [BEN 00b]. Le *modèle de transition conditionnelle* peut être simplifié de deux façons. Soit on suppose que $P(s_{t+1}|s_t, \mathbf{x}_1^t) = P(s_{t+1}|\mathbf{x}_{t-d}^t)$, ce qui conduit à une architecture du type mixture d'experts [JOR 94]. Néanmoins, l'estimation des paramètres du modèle de transition demeure difficile sachant que peu de transitions sont observées au regard du nombre de données. C'est pourquoi les méthodes fondées sur le principe 'diviser-pour-règner' présentent une forte variance [JOR 94]. Soit, on suppose que les états s_{t+1} sont indépendants de x_t conditionnellement à s_t, i.e., $P(s_{t+1}|s_t, \mathbf{x}_{t-d}^t) = P(s_{t+1}|s_t)$, pour chaque t. Cette hypothèse conduit à un HMM.

La vraisemblance se calcule aisément. Sachant que $p(y_t|\mathbf{x}_{t-d}^t, s_t = i)$ est une fonction de $\epsilon_t = y_t - f_i(\mathbf{x}_{t-d}^t)$, $p(y_t|\mathbf{x}_{t-d}^t, s_t = i)$ sera noté par $p(\epsilon_t|i)$ par souci de compacité. Nous supposerons de plus que $x_{1-d}, \ldots, x_{-1}, x_0$ sont connus ainsi que les observations jusqu'à l'instant $T+\tau$. La matrice de transition, $A = \{a_{ij}\}$, détermine la probabilité de passer d'un état i à l'état j, et θ représente les paramètres du système.

Il existe plusieurs façons d'exprimer la vraisemblance du système, $\mathcal{L}$. Nous optons pour la formulation suivante [AUS 01c] :

$$\begin{aligned} \mathcal{L} &= \sum_{\mathbf{s}_1^T} p(\mathbf{x}_{1-d}^{T+\tau}, \mathbf{s}_1^T) \\ &= p(\mathbf{x}_{1-d}^{\tau}) \sum_{\mathbf{s}_1^T} P(s_1) \prod_{t=1}^{T-1} P(s_{t+1}|s_t) \prod_{t=1}^{T} P(y_t|s_t, \mathbf{x}_{t-d}^t) \\ &= p(\mathbf{x}_{1-d}^{\tau}) \sum_{s_1^T} P(s_1) \prod_{t=1}^{T-1} a_{s_t, s_{t+1}} \prod_{t=1}^{T} p(\epsilon_t|s_t; \theta) \end{aligned}$$

A la différence de Kohlmorgen et al. [KOH 99], nous nous proposons d'ajuster les probabilités de transition, a_{ij}, et les σ_i.

8.3. L'apprentissage des experts

On cherche à approximer $f_i()$ par des experts. L'algorithme EM permet de trouver une suite de paramètres qui augmente la vraisemblance à chaque itération et converge vers un maximum local de la vraisemblance. Rappelons le principe de cet algorithme.

La série $\{x_t\}$ est observée, les états $\{s_t\}$ ne le sont pas. La finalité de EM est de trouver, par une procédure itérative, le maximum local de la vraisemblance dans une situation ou la densité de probabilité des données observées conditionnée aux données non observées est connue explicitement. Posons θ^{old} un vecteur de paramètres. EM alterne deux étapes :

- E-step : Calculer $Q(\theta, \theta^{old}) = \mathrm{E}[\log p(\{x_t\}, \{s_t\}; \theta) | \{x_t\}; \theta^{old})]$.
- M-step : Trouver θ^{new} qui maximise $Q(\theta, \theta^{old})$.

La vraisemblance à chaque itération sera croissante à condition que $Q(\theta^{new}, \theta^{old}) > Q(\theta^{old}, \theta^{old})$, jusqu'à ce que θ^{old} soit un maximum local de la *pseudo-log-vraisemblance*, $Q(\theta, \theta^{old})$. En pratique, on a recours à un algorithme EM généralisé (GEM) qui se contente de produire un accroissement de la pseudo-log-vraisemblance à chaque étape M même si la convergence peut s'avérer plus lente.

On parlera de données *complètes* lorsque les observations, x_t, sont accompagnées des variables discrètes, s_t, spécifiant le régime dynamique correspondant.

8.4. Calcul de la pseudo-log-vraisemblance

L'information fournie par les données est résumée dans la séquence des innovations. Aussi, pour simplifier nos notations, on dira que $\epsilon_1^T = (\epsilon_1, \ldots, \epsilon_T)$ sont les données observées et $\mathbf{s}_1^T = (s_1, \ldots, s_T)$ les données cachées. Soit $s_{T+1} = 0$ et $a_{i0} = 1$ pour chaque i. Avec ces notations, la vraisemblance des données *complètes* s'écrit :

$$\mathcal{L}^{comp} = p(\epsilon_1^T, \mathbf{s}_1^T; \theta) = p(\mathbf{x}_{1-d}^\tau) P(s_1) \prod_{t=1}^{T} a_{s_t, s_{t+1}} p(\epsilon_t | s_t; \theta) \tag{8.3}$$

$p(\mathbf{x}_{1-d}^\tau)$ et $P(s_1)$ sont indépendants des paramètres du modèle, ils seront omis dans la suite. On peut écrire

$$Q(\theta, \theta^{old}) = \sum_{s_1=1}^{N} \cdots \sum_{s_T=1}^{N} \sum_{t=1}^{T} \ln\left[a_{s_t, s_{t+1}} p(\epsilon_t | s_t; \theta)\right] P(s_1^T | \epsilon_1^T; \theta) \tag{8.4}$$

Il est commode de ré-écrire la log-vraisemblance sous une forme équivalente :

$$\ln\left(\mathcal{L}^{comp}\right) = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \delta_{is_t} \delta_{js_{t+1}} \ln\left[a_{ij} p(\epsilon_t | i; \theta)\right] \tag{8.5}$$

L'usage du symbole de Kronecker, δ_{ij}, permet d'intégrer le logarithme dans la somme, simplifiant ainsi le problème de maximisation [BIS 95]. On a l'identité suivante :

$$\sum_{s_1=1}^{N} \cdots \sum_{s_T=1}^{N} \delta_{is_t} \delta_{js_{t+1}} P(s_1^T | \epsilon_1^T; \theta^{old}) = P(s_t = i, s_{t+1} = j | \epsilon_1^T; \theta^{old}) \tag{8.6}$$

Posons $\xi_t(i,j) = P(s_t = i, s_{t+1} = j | \epsilon_1^T; \theta^{old})$. $Q(\theta, \theta^{old})$ se simplifie en

$$Q(\theta, \theta^{old}) = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \ln\left[a_{ij} p(\epsilon_t | i; \theta)\right] \xi_t(i,j) \tag{8.7}$$

Il vient que maximiser $Q(\theta, \theta^{old})$ est équivalent à minimiser le coût :

$$\begin{aligned} E &= -\sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \ln\left[a_{ij} p(\epsilon_t | i; \theta)\right] \xi_t(i,j) \\ &= -\sum_{i=1}^{N} \sum_{t=1}^{T} \ln\left[p(\epsilon_t | i; \theta)\right] \sum_{j=1}^{N} \xi_t(i,j) - \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \xi_t(i,j) \ln a_{ij} \\ &= -\sum_{i=1}^{N} \sum_{t=1}^{T} \ln\left[p(\epsilon_t | i; \theta)\right] \gamma_t(i) - \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \xi_t(i,j) \ln a_{ij} \end{aligned} \tag{8.8}$$

où $\gamma_t(i) = P(s_t = i | \epsilon_1^T; \theta^{old})$ et $\sum_i \gamma_t(i) = 1$. Les $\gamma_t(i), \xi_t(i,j)$ s'obtiennent grâce à l'algorithme *forward-backward* de Baum et Welch [BAU 70].

8.5. Maximisation de la pseudo-log-vraisemblance

Il s'agit de maximiser E par rapport aux paramètres des MLP , des probabilités de transition et la dispersion du bruit. La pseudo-log-vraisemblance s'exprime comme la somme de deux termes. Les paramètres des experts influent sur le premier terme et les probabilités de transition sur le second uniquement. Il suffit donc de maximiser les termes séparément :

Probabilités de transition - Les valeurs optimales sont obtenues par application du théorème de Lagrange,

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i))}. \tag{8.9}$$

Variances σ^2 - De même,

$$\hat{\sigma}_i^2 = \frac{1}{\sum_{t=1}^{T} \gamma_t(i)} \sum_{t=1}^{T} \gamma_t(i)(y_t - f_i(x_t))^2. \tag{8.10}$$

Experts - Les paramètres $w(j)$ de l'expert j s'obtiennent par minimisation d'une fonction coût pondérée par la probabilité, $\gamma_t(j)$, d'être dans cet état :

$$\hat{w}(j) = arg \min_{w} \sum_{t=1}^{T} \gamma_t(j)(y_t - f_k(\mathbf{x}_{t-d}^t))^2. \tag{8.11}$$

En somme, l'algorithme EM obtenu alterne l'algorithme *forward-backward* de Baum et Welch [BAU 70] à l'étape E avec une série de problème de moindres carrés indépendants à l'étape M.

8.6. Segmentation

L'alternance des étapes E et M conduit à un maximum local de vraisemblance, toutefois, à la lumière de notre expérience, cette solution n'est pas toujours acceptable. Les experts obtenus demeurent influencés par les données qui ne lui sont pas assignées. C'est le cas lorsque les $\gamma_t(j)$ ne tendent pas exactement vers 0 ou 1. Pour remédier à ce problème et obtenir une segmentation "dure", Kohlmorgen et al. ont introduit un recuit déterministe au moyen de la fonction "soft-max" sur les $\gamma_t(i)$ (i.e., $e^{\gamma_t(i)/\theta}/(\sum_{j=1}^{N} e^{\gamma_t(j)/\theta})$) où

la "température" θ est graduellement abaissée durant l'apprentissage. Néanmoins, cette heuristique mène à une solution qui désigne, à l'instant t, l'expert $argmax_j P(s_t = j|x_1^T, y_1^T)$, une solution clairement *sous-optimale* comme le souligne [RAB 89]. Rappelons que la solution "optimale" est constituée des états que visite la séquence la plus vraisemblable, calculée par l'algorithme de Viterbi [FOR 73], (i.e., $argmax_{s_1,\ldots,s_T}\ P(s_1,\ldots,s_T|\mathbf{x}_1^T, \mathbf{y}_1^T)$).

Pour contourner ce problème, la méthode employée ici succinctement consiste à ralentir la convergence des σ_i afin d'augmenter graduellement le pouvoir de discrimination des experts car lorsque les σ_i sont grands, les $\gamma_t(j)$ sont équiprobables, et ce, indépendamment des erreurs des experts. L'idée est d'initialiser les σ_i à des grandes valeurs afin de promouvoir la diversification des experts et de faire decroître "lentement" les σ_i vers les $\hat{\sigma}_i$. A ce stade une segmentation s'est établie et l'algorithme de Viterbi est appliqué pour obtenir la séquence optimale et procéder à la spécialisation définitive des experts.

Viterbi substitut $\mathcal{L}'$, la vraisemblance sur les données "complètes", à $\mathcal{L}^{comp}$ dans la procédure EM. $\mathcal{L}'$ est donnée par

$$\mathcal{L}' = \max_{s_1^T} p(\epsilon_1^T, \mathbf{s}_1^T; \theta) \tag{8.12}$$

Soit $s_t^\star, t = 1,\ldots,T$ la séquence optimale des états de la HMM obtenue par Viterbi, la log-vraisemblance se décompose en

$$\begin{aligned} \ln \mathcal{L}' &= \ln p(\epsilon_1,\ldots,\epsilon_T|s_1^\star,\ldots,s_T^\star;\theta) + \ln p(s_1^\star,\ldots,s_T^\star;\theta) \\ &= \ln \pi_{s_1^\star} + \sum_{t=1}^{T} \ln p(\epsilon_t|s_t^\star;\theta) + \sum_{t=1}^{T} \ln a_{s_t^\star, s_{t+1}^\star} \end{aligned}$$

Maximiser $\mathcal{L}'$ revient à minimiser

$$E' = -\sum_{t=1}^{T} \ln\left[p(\epsilon_t|s_t^\star;\theta)\right] = -\sum_{i=1}^{N}\sum_{t=1}^{T} \ln\left[p(\epsilon_t|i;\theta)\right]\delta_{i,s_t^\star} \tag{8.13}$$

En comparant E et E', on voit que $\gamma_t(i)$ est replacé par $\delta_{i,s_t^\star}$. Dans le cas gaussien, l'expert j est entraîné avec la fonction coût :

$$\sum_{t=1}^{T} \delta_{k,s_t^\star} \cdot (y_t - f_j(\mathbf{x}_{t-d}^t)^2 \tag{8.14}$$

En résumé, la méthode opère en deux étapes : dans un premier temps, l'algorithme GEM est invoqué, les σ_i et a_{ij} décroissent "lentement" vers $\hat{\sigma}_i$ et $\hat{a}_{ij}$ respectivement. La segmentation est alors obtenue par application de l'algorithme de Viterbi et les experts sont exclusivement entraînés sur leurs données respectives jusqu'à convergence complète.

8.7. Estimation directe des paramètres

L'algorithme EM est la méthode d'estimation des paramètres la plus populaire des modèles hybrides. L'algorithme opère en deux phases : la phase d'apprentissage où la procédure GEM est répétée un certain nombre d'époques (i.e., une époque est une passe complète à travers la base d'apprentissage). Pour ne privilégier aucun expert a priori, nous fixons initialement $a_{ij} = 1/N$. Les γ sont calculés par algorithme *forward-backward* et stockés en mémoire avant chaque passe des experts à travers la base d'apprentissage. Il faut veiller à remettre à l'échelle les termes α (voir [RAB 89]) pour éviter les problèmes numériques. Après chaque étape M, les a_{ij} et σ_i sont ajustés. Une fois la phase d'apprentissage terminée, commence la phase de segmentation. E est alors remplacé par E' et les experts sont entraînés sur leurs données jusqu'à complète convergence. L'algorithme de Viterbi fournit alors la segmentation finale.

Cet algorithme est, par essence, très lent et ne se prête pas à une implémentation *on-line*. Une méthode pour l'estimation directe des paramètres a été développée récemment dans la thèse de J. Rynkiewicz [RYN 00]. Il est en effet possible de calculer récursivement la log-vraisemblance et sa dérivée sous une forme additive (même le calcul repose encore l'hypothèse erronée que les observations sont i.i.d.), ce qui autorise la mise en oeuvre d'un algorithme d'approximation stochastique pour l'estimation des paramètres. Les conditions de consistance et de normalité asymptotique des procédures d'estimation récursive ne sont pas établies pour ce modèle général. On aboutit cependant à une méthode d'estimation récursive *on-line* que les simulations ont clairement validée.

8.8. Simulations

La méthode *off-line* est illustrée sur des données artificielles et financières. Les experts sont des MLP à 6 entrées (la fenêtre temporelle), 10 neurones cachés et une sortie linéaire. Les apprentissages ont été répliqués 10 fois avec des poids de valeur aléatoire. Les σ_i sont égaux. Pour évaluer les performances de l'algorithme, une MSE de *référence* est calculée en n'utilisant qu'un seul MLP

sur l'ensemble des données. Lorsque c'est possible, nous calculons la MSE *minimale* en réalisant un pré-apprentissage des experts dans leur régime dynamique avant de leur combiner. On peut interpréter la MSE *minimale* comme la valeur cible de l'erreur de notre modèle.

8.8.1. *Fonction logistique*

Dans ce premier exemple, une suite présentant 2 régimes a été générée. Les régimes sont $U_{n+1} = f(U_n)$ et $U_{n+1} = fof(U_n)$ où $f(x) = 4x(1-x)$ est la fonction logistique. 2500 valeurs ont été simulées ainsi avec une période de transition tous les 50 itérations en moyenne (T=50) (i.e., $a_{12} = a_{21} = 1/50$). Par souci de clarté, la Figure 8.1 expose uniquement une portion de la suite avec les transitions originales et la segmentation obtenue.

Les σ_i sont pris égaux de valeur initiale $\sigma_i = 20$ et les a_{ij} à $1/2$ (équiprobabilité) de même que le vecteur Π des probabilités initiales. Les poids des MLP ont été initialisés selon une loi uniforme $\mathcal{U}_{-1,1}$. Les 2500 valeurs ont servi de base d'apprentissage. Pour les paramètres d'apprentissage, les choix de $\beta = 0.1$ et $\alpha = 0.01$ semblent satisfaisants. Ainsi, la matrice $\mathcal{A}$ converge plus vite que σ_i et w_{ij}, vers ses valeurs cibles à chaque itération M. Initialement, $\eta = 0.1$.

Une segmentation quasi parfaite est apparue après 400 époques même si certaines transitons brèves (e.g., 1495 et 1557) sont ignorées. Un faible décalage subsiste toujours entre les transitions (Fig. 8.1). La MSE de référence obtenue avec un MLP de 12 neurones cachés est $MSE = 0.0618$, la MSE minimale vaut $MSE = 0.0024$. L'algorithme converge vers la MSE minimale, $MSE = 0.0024$, avec 2 MLP experts de 10 neurones cachés. L'apprentissage a été relancé 3 fois, ce qui explique les pics de la courbe d'erreur. La segmentation finale est obtenue après une passe de l'algorithme de Viterbi, à la fin de l'apprentissage. En fin d'apprentissage les coefficients de la matrice de transition ont atteint pratiquement leurs valeurs théoriques :

$$a_{ij} = \begin{cases} \frac{T}{T+N-1} & \text{si } i = j \\ \frac{1}{T+N-1} & \text{si } i \neq j \end{cases}$$

ce qui nous donne pour $T = 50$ et $N = 2$

$$\mathcal{A} = \begin{pmatrix} 0.9803 & 0.0196 \\ 0.0196 & 0.9803 \end{pmatrix}$$

Or nous trouvons comme valeurs effectives :

$$\widehat{\mathcal{A}} = \begin{pmatrix} 0.9826 & 0.0173 \\ 0.0268 & 0.9731 \end{pmatrix}$$

8.8.2. *Hénon-Logistique*

La suite alterne entre le système bidimensionnel de Hénon $x(k+1) = 1.0 - 1.4 \cdot x^2(k) + 0.3 \cdot x(k-1)$ et la fonction logistique $x(k+1) = 4.0 \cdot x(k) \cdot (1 - x(k))$. 5000 valeurs ont été synthétisées avec des transitions aléatoires à chaque instant avec une probabilité de 1/100. La longueur moyenne d'une plage dynamique est de 100 unités de temps. Les deux suites ont été normalisées dans $(0, 1)$ et sont difficilement distinguables à l'oeil nu.

La MSE obtenue sur l'ensemble des données avec un seul MLP vaut $MSE = 0.018$. La MSE *minimale* obtenue avec 2 MLP pré-entraînés dans leur dynamique (donc en trichant) donne $MSE = 0.009$. L'approche hybride donne $MSE = 0.011$. Une transition est illustrée

8.8.3. *Mackey-Glass*

Dans cet exemple tiré de [KOH 99], on considère les équations de Mackey-Glass, $dx(t)/dt = -0.1x(t) + 0.2x(t - \Delta)/(1 + x(t - \Delta)^{10})$. 3 régimes distincts ont été obtenus avec 3 délais $\Delta = 17$, $\Delta = 23$, et $\Delta = 30$. Une série chaotique de 500 donnés a été générée par la méthode de Runge-Kutta à l'ordre 4 suivi d'un échantillonnage à une fréquence de $f = 6$. Chaque régime couvre 100 points exactement. La Figure 8.4 montre la segmentation obtenue avec les sorties de chaque MLP. On observe par exemple que l'expert 1, figure en haut à droite, fournit de bonnes prévisions uniquement sur sa plage dynamique : $[0 : 100]$ et $[300; 400]$. Il en va de même pour les expert 2 et 3. La MSE minimale vaut $MSE = 0.09$ et la MSE de référence vaut $MSE = 0.018$. Nous obtenons $MSE = 0.011$. Les experts se sont spécialisés dans leur régime dynamique.

8.8.4. *Données réelles*

Sur ce dernier exemple, nous avons pris la séquence des cotations d'une action d'une (grande) société à raison de 4 valeurs par jour, soit un total de 4085 valeurs. Le lecteur pourra consulter un travail récent de Y. Bengio et al. dans le même esprit [BEN 00b]. Afin de stationnariser la suite, nous avons considéré la séquence des rendements $y_t = (x_{t+1} - x_t)/x_t$ normalisés dans $(-1, +1)$. Le

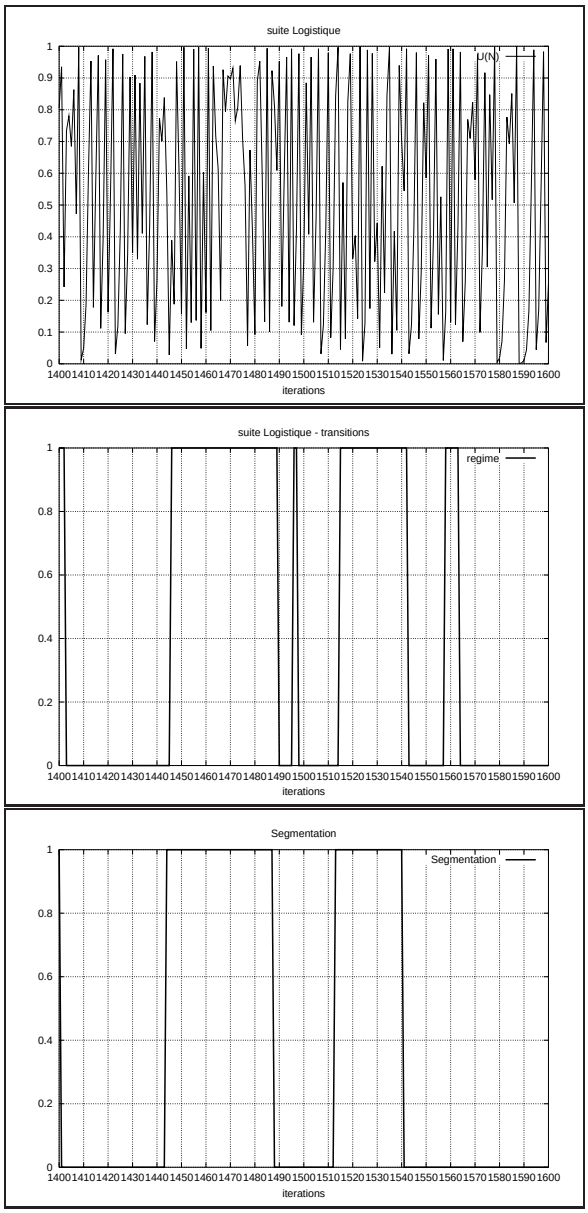


Figure 8.1. *Fonction logistique à deux dynamiques. Haut : la suite stationnaire par morceaux. Milieu : la vraie segmentation sur une portion des données. Bas : Segmentation obtenue après apprentissage. Certains changements brefs de dynamique n'ont manifestement pas été identifiés.*

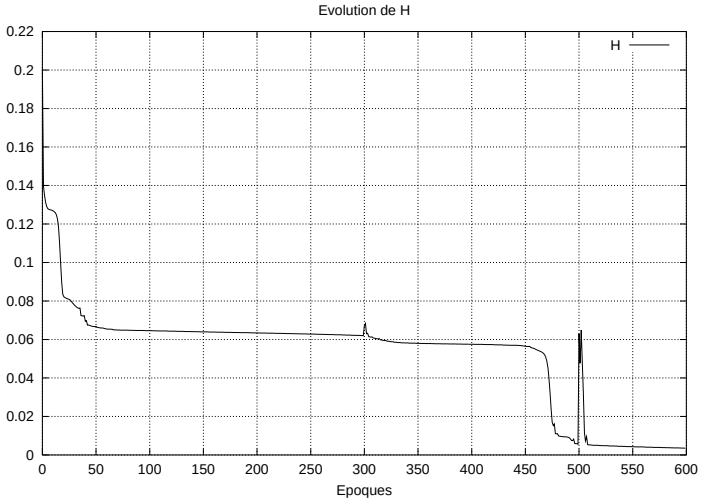


Figure 8.2. *Evolution de l'erreur lors de l'apprentissage.*

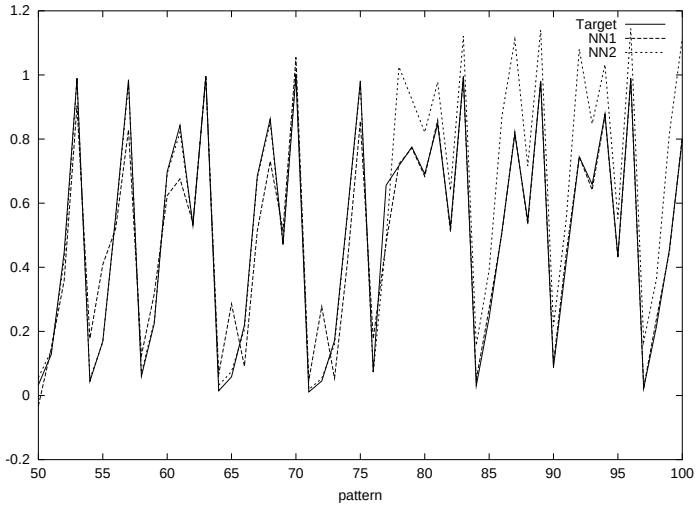


Figure 8.3. *Hénon-Logistique. Les sorties des 2 MLP sont visualisées en pointillé sur une portion des données centrée sur une transition à l'instant 77.*

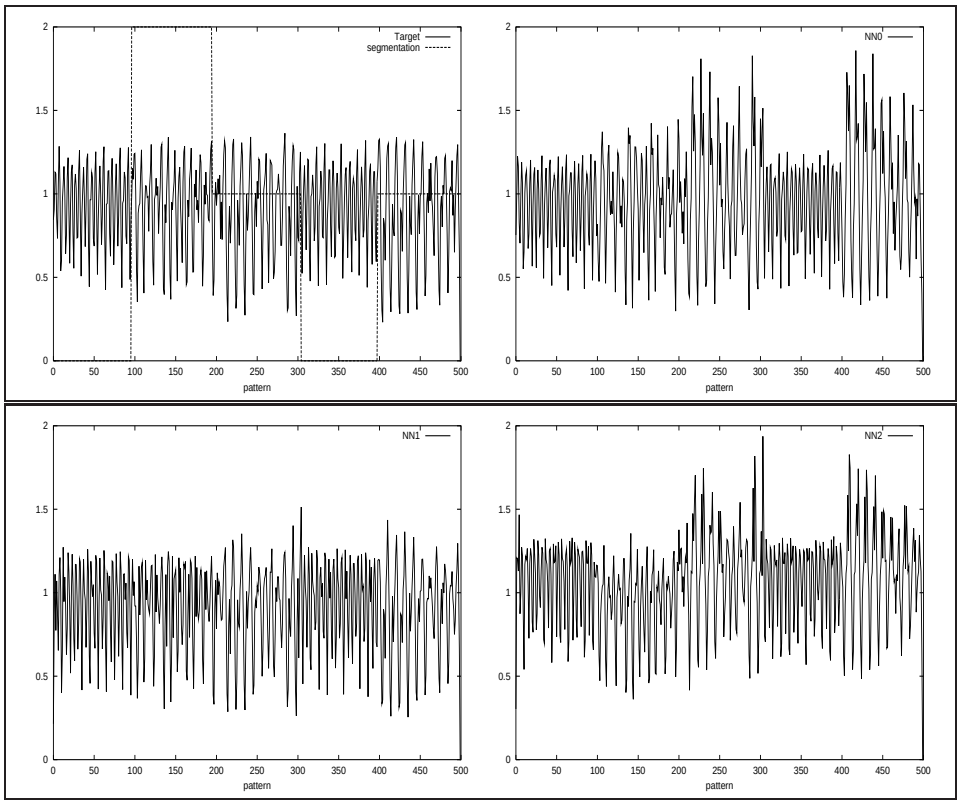


Figure 8.4. *Mackey-Glass. Haut gauche : vraie série et la segmentation obtenue avec l'algorithme. Autres figures : les sorties des 3 experts neuronaux.*

nombre d'experts à été varié de 1 à 4. Un seul expert donne $MSE = 0.6$. La meilleure performance a été obtenue avec 3 experts, $MSE = 0.022$. Au delà, les experts ne sont jamais sélectionnés donc inutiles. On observe Figure 8.5, qu'à défaut d'exhiber des dynamiques distinctes, l'algorithme a réparti les données aux experts suivant leur amplitude de façon à sur-apprendre parfaitement la série entière (4085 valeurs) malgré un nombre modéré de paramètres (151 au total).

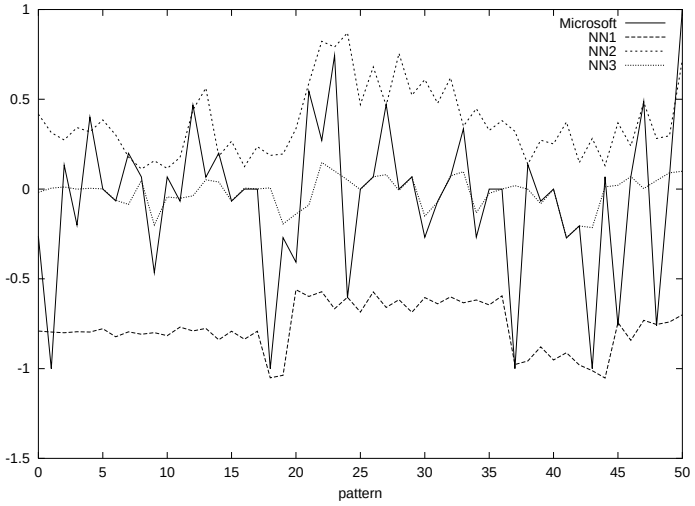


Figure 8.5. *Rendement d'une action en trait plein, 3 experts neuronaux en pointillé. L'algorithme segmente selon l'amplitude de la série originale. Le sur-apprentissage est patent.*

8.9. Conclusion et Perspectives

Une méthode pour la segmentation non supervisée des séries temporelles a été présentée. La segmentation de la suite opère grâce à la combinaison d'une chaîne de Markov cachée et de prédicteurs (neuronaux) : les experts. La méthode a été illustrée sur des données artificielles et réelles.

Bien que performant en segmentation, cet algorithme tel quel n'est pas adapté à la prédiction : $\gamma_i(t)$ dans sa définition dépend du futur (il est non-causal et donc hors-ligne). De plus l'algorithme *forward-backward* nécessite de l'ordre de TM^2 opérations à chaque instant t et une mémoire de l'ordre de TM. Pour réaliser des prédictions, des algorithmes séquentiels "en ligne" ont

été proposés pour l'estimation des paramètres de la HMM [KRI 93], des modèles auto-régressifs [HOL 94] et récemment généralisés aux MLP [RYN 01]. En effet, partant d'une formulation additive et récursive de la log-vraisemblance, - différente de celle proposée dans cet article -, il est possible d'appliquer un algorithme d'optimisation différentielle classique afin de réduire considérablement les temps de calcul et accélérer la convergence [RYN 01].

Chapitre 9

PERSPECTIVES

J'ai rassemblé et synthétisé dans ce "document de synthèse" mes travaux de recherche entamés au cours de ma dernière année de thèse en 1995, et poursuivis au LIMOS de 1996 jusqu'à nos jours.

L'analyse exhaustive de la *décroissance du flot arrière du gradient d"erreur* (GEBF) a été menée pour une classe assez vaste de d'architectures neuronales bouclées à délais pour l'apprentissage de trajectoires et de points fixes. Cette analyse met en lumière la difficulté d'apprentissage des dépendances à longue portée par des algorithmes fondés sur le gradient de la fonction coût relatif aux paramètres du modèle. Ce comportement qualifié de *forgetting behavior* dans la littérature est une pierre d'achoppement que nombre de travaux ont tenté de contourner par des techniques diverses.

Des conditions suffisantes pour garantir la convergence de l'GEBF ont été établies. Elles s'expriment explicitement en fonction de la matrice de poids et s'appliquent à de nombreux les réseaux bouclés introduits dans la littérature ces dernières années. De ces conditions découle une borne supérieure sur le nombre de rétro-propagations dans le temps pour limiter l'erreur de (BPTT).

Les chapitres suivants ont illustré les aptitudes des réseaux bouclés à délais à réaliser des prévisions à court terme de séries temporelles issue des sciences environnementales et des télécom, au sens large. Une méthode hybride MLP/ondelettes fondée sur une analyse multirésolution a été décrite pour traiter des processus aux dépendances à long terme, typiquement le télé-trafic. Les derniers développements d'un projet au long cours dédié à la gestion des ressources dans un réseau télécom multiservice, ont également été présentés. Ce document s'achève sur un problème connexe à la prévision, à savoir la segmentation de séries temporelles stationnaires par morceaux. L'algorithme EM a été employé avec succès pour l'estimation des paramètres du modèle, les probabi-

lités de transition et la variance du bruit.

Perspectives - J'ai l'intention d'intensifier en 2003 ma collaboration avec Patrice Abry (ENS Lyon) et Pierre Chainais (MCF au LIMOS) afin de caractériser, synthétiser et prédire le comportement du télé-trafic à court terme dans le cadre de l'AS (CNRS) Métrologie Internet.

Je souhaite également mettre en oeuvre des méthodes de régularisation structurelle (e.g. élimination de connexions/neurones superflus) pour l'identification (presque sûre) du modèle DRNN, ainsi que des méthodes analytiques pour construire des intervalles de confiance, le tout sous la forme d'une *toolbox Matlab*. Il me faudra aussi achever le construction du site Web 'NEUROP' à l'ESO pour fournir aux astronomes des prévisions en temps réel du seeing sur des horizons de 10 à 60 minutes assortis d'intervalles de confiance.

A titre de remarque, je travaille également avec Jean-Marc Petit (LIMOS) sur un problème d'extraction de *dépendances fonctionnelles* dans les bases de données génomiques [AUS 02d] dans le cadre de l'AS (CNRS) Gafodonnées. Les dépendances fonctionnelles sont des contraintes d'intégrité importantes pour la conception et l'analyse des bases de données relationnelle. Nous réfléchissons à la façon d'appliquer ces techniques à l'inférence de dépendances fonctionnelles floues afin de mettre à jour certains mécanismes (activation/inhibition) de régulation entre les gènes.

Bibliographie

[ABR 98] P. Abry and D. Veitch. The wavelet analysis of long-range dependent traffic. *IEEE IT*, 44(1):377–283, 1998.

[ABR 01] P. Abry. *Lois d'Echelle. Multirésolution et Ondelettes. Applications au Télétrafic Informatique et à la Turbulence Développée.* Thèse d'Habilitation, Université Claude Bernard, lyon I, 2001.

[ABR 02] P. Abry, R. Baraniuk, P. Flandrin, R. Riedi and D. Veitch. The multiscale nature of network traffic: Discovery, analysis and modelling. *IEEE Signal Processing*, April 2002.

[ALM 87] L.B. Almeida. Backpropagation in perceptrons with feedback. *Neural Computers*, pages 199–208, 1987.

[ATI 00] A.F. Atiya and G.P. Parlos. New results on recurrent network training: Unifying the algorithms and accelerating convergence. *IEEE Transactions on Neural Networks*, 11(3):697–709, 2000.

[AUS 94a] A. Aussem. Call admission control in ATM networks with the random neural network. In *Proc. of the International Conference on Neural Networks*, volume 4, pages 2482–2487, 1994.

[AUS 94b] A. Aussem, F. Murtagh and M. Sarazin. Dynamical recurrent neural networks and pattern recognition methods for time series prediction: Application to seeing and temperature forecasting in the context of ESO's VLT astronomical weather station. *Vistas in Astronomy*, 38:357–374, 1994.

[AUS 95a] A. Aussem. *Théorie et Applications des Réseaux de Neurones Récurrents et Dynamiques à la Prédiction, à la Modélisation et au Contrôle Adaptatif des Processus Dynamiques.* Thèse de Doctorat. Université René Descartes, Juillet 1995.

[AUS 95b] A. Aussem, F. Murtagh and M. Sarazin. Dynamical recurrent neural networks towards environmental time series prediction. *International Journal on Neural Systems*, 6(2):145–170, 1995.

[AUS 96] A. Aussem, F. Murtagh and M. Sarazin. Fuzzy astronomical seeing nowcasts with a dynamical and recurrent connectionist network. *Neurocomputing*, 13(4):359–373, 1996.

[AUS 97] A. Aussem, F. Murtagh and M. Sarazin. Combining neural network forecasts on wavelet-transformed time series. *Connection Science*, 9(1):113–121, 1997.

[AUS 98a] A. Aussem. Nonlinear modeling of chaotic processes with dynamical recurrent neural networks. In *Proc. of the Neural Networks and Their Applications NEURAP'98, Marseille, France*, pages 425–433, 1998.

[AUS 98b] A. Aussem, J. Campbell and F. Murtagh. Wavelet-based feature extraction and decomposition strategies for financial forecasting. *Journal of Computational Intelligence in Finance*, 6(2):5–12, 1998.

[AUS 98c] A. Aussem and F. Murtagh. A neuro-wavelet strategy for web traffic forecasting. *Research in Official Statistics*, 1(1):65–87, 1998.

[AUS 99a] A. Aussem. Dynamical recurrent neural networks towards prediction and modeling of dynamical systems. *Neurocomputing*, 28(3):207–232, 1999.

[AUS 99b] A. Aussem and D. Hill. Wedding connectionist and algorithmic modelling towards forecasting C*aulerpa taxifolia* development in the north-western mediterranean sea. *Ecological Modelling*, 120:225–236, 1999.

[AUS 99c] A. Aussem, S. Rouxel and R. Marie. Neural-based queueing system modelling for service quality estimation in B-ISDN networks. In *Proc. of the International Conference on Artificial Neural Networks Edinburgh, Scotland*, pages 970–975, 1999.

[AUS 00a] A. Aussem. Sufficient conditions for error back flow convergence in dynamical recurrent neural networks. In *Proc. of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, Como, Italy*, pages 577–582, July 2000.

[AUS 00b] A. Aussem and D. Hill. Neural networks metamodelling for the prediction of C*aulerpa taxifolia* development in the mediterranean sea. *Neurocomputing*, 30:71–78, 2000.

[AUS 00c] A. Aussem, A. Mahul and R. Marie. Queueing network modelling with distributed neural networks for service quality estimation in B-ISDN networks. In *Proc. of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, Como, Italy*, volume 5, pages 392–397, July 2000.

[AUS 00d] A. Aussem, G. Tran and M. Sarazin. On-line prediction of astronomical seeing fluctuations with neural networks. In *Astronomical Site Evaluation in the Visible and Radio Range, IAU Site 2000, ASP Conference Series*, volume 266, 2000.

[AUS 01a] A. Aussem, and M. Fuentes. Segmentation de suite chronologique par une chaîne de Markov cachée d'experts neuronaux. In *Proc. of the Conférence d'Apprentissage Grenoble, France*, pages 165–177, Juin 2001.

[AUS 01b] A. Aussem and C. Boutevin. Segmentation of switching dynamics with a hidden Markov model of neural prediction experts. In *Proc. of the European Symposium on Artificial Neural Networks, Bruges, Belgium*, pages 251–256, April 2001.

[AUS 01c] A. Aussem and F. Murtagh. Web traffic demand forecasting using a wavelet-based multiscale decomposition. *International Journal of Intelligent Systems*, 16(2):215–236, 2001.

[AUS 02a] A. Aussem. Global asymptotic stable relaxation for a discrete-time delayed neural networks. *Rapport de recherche du LIMOS*, 2002.

[AUS 02b] A. Aussem. Gradient decay analysis and stability conditions for dynamical recurrent neural networks. *Neural Computation*, 14(8):1907–1927, 2002.

[AUS 02c] A. Aussem. Optimal combination of model predictions based on a covariance weighting scheme. *Rapport de recherche du LIMOS*, 2002.

[AUS 02d] A. Aussem and J.-M. Petit. ϵ-functional dependency inference: application to DNA microarray expression data. In *18èmes Journées Bases de Données Avancées, BDA'02, Evry, France*, Octobre 2002.

[BAK 00] R. Bakker, J.C. Schooten, C.L. Giles, F. Takens and C.M Van den Bleek. Learning chaotic attractors by neural networks. *Neural Computation*, 12:2355–2383, 2000.

[BAL 94] P. Baldi and S. Atiya. How delays affect neurtal dynamics and learning. *IEEE Transactions on Neural Networks*, 5(4):612–621, 1994.

[BAL 95] P Baldi. Gradient descent learning algorithm overview: A general dynamical systems perspective. *IEEE Transactions on Neural Networks*, 6:182–195, 1995.

[BAU 70] L.E. Baum, Petrie T., Soules G. and Weiss N. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, 41:164–171, 1970.

[BEN 92] Y. Bennani and P. Gallinari. Task decomposition through a modular connectionist architecture: A talker identification system. In *Proc. of the International Conference on Artificial Neural Networks,Britghton, U.K.*, 1992.

[BEN 93] Y. Bennani. Probabilistic cooperation of connectionist expert modules: Validation on a speaker identification task. In *International Conference on Acoustics, Speech, and Signal Processing ICASSP'93, Minneapolis, Minnesota, U.S.A.*, pages 541–544, 1993.

[BEN 94a] Y. Bengio and P. Frasconi. Credit assignment through time: Alternatives to backpropagation. In J.D. Cowan, G. Tesauro and J. Alspector, eds, *Advances in Neural Information Processing Systems*, volume 6, pages 75–82. Cambridge, MA:MIT Press, 1994.

[BEN 94b] Y. Bengio, P. Simard and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5:157–166, 1994.

[BEN 94c] Y. Bennani and P. Gallinari. Multi-modular and hybrid connectionist approach for pattern recognition. *International Journal of Neural Systems*, 5(3):207–216, 1994.

[BEN 95a] Y. Bengio and P. Frasconi. An input-output HMM architecture. In G. Tesauro, D.S. Touretzky and T.K. Leen, eds, *Advances in Neural Information Processing Systems*, volume 7. Cambridge, MA:MIT Press, 1995.

[BEN 95b] Y. Bennani. Modular and hybrid connectionist system for automatic speaker identification. *Neural Computation*, 7(4):791–797, 1995.

[BEN 00a] Y. Bengio. Probabilistic neural network models for sequential data. In *Proc. of the IEEE-INNS-ENNS International Conference on Artificial Neural Networks, Como, Italy*, 2000.

[BEN 00b] Y. Bengio, V.-P. Lauzon and R. Ducharme. Experiments on the application of iohmms to model financial returns series. *IEEE Transactions on Neural Networks*, 12(1):113–123, 2000.

[BEN 03] Y. Bengio and N. Chapados. Extensions to metric-based model selection. *Jounral of Machine Learning Research, to appear*, 2003.

[BIS 95] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[BOD 90] U. Bodenhausen. Learning internal representation of pattern sequences with adaptive time delays. In *Proc. of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, 1990.

[BOL 98] R. Bolla, F. Davoli, P. Maryni and T. Parisini. An adaptative neural network admission controller for dynamic bandwith allocation,. *IEEE Transactioncs on Systems, Man and Cybernetics - Part B: Cybernetics*, 28(4):592–601, 1998.

[BON 00] R. Boné, M. Cruciani, G. Verley and J.P. Asselin de Beauville. A bounded exploration approach to constructive algorithms for recurrent neural networks. In *Proc. of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, pages 27–32, 2000.

[BRY 75] A. Bryson and Y.-C. Ho. *Applied Optimal Control*. Washington, D.C.: Hemisphere, 1975.

[CAM 99] P. Campolucci, A. Uncini, F. Piazza and B.D. Rao. On-line learning algorithms for locally recurrent neural networks. *IEEE Transactions on Neural Networks*, 10:253–271, 1999.

[CAN 99] S. Canu and A. Elisseff. Why sigmoid-shaped function are good for learning. In *Proc. of the Snowbird Workshop*, Avril 1999.

[CAO 98] J.D. Cao and D.M. Zhou. Stability analysis of delayed cellular neural networks. *Neural Networks*, 11:1601–1605, 1998.

[CAS 98] E. Casilari, A. Alfaro and A. Reyes et al. Neural modelling of Ethernet traffic over ATM networks. In *Proceedings of EANN'98*, pages 304–307, June 1998.

[CLé 98] F. Clérot, P. Gouzien, R. Feraud, A. Gravey and D. Collobert. Using neural networks for predicting transient leaky bucket characterisations of data traffic. Applications to dynamic resource allocation in ATM networks. In *Proceedings of IFIP 98 ATM*, pages 304–307, July 1998.

[COH 97] B. Cohen, D. Saad and E. Marom. Efficient training of recurrent neural networks with delays. *Neural Networks*, 10:51–59, 1997.

[COM 96] COST 242 Management committee, ed. *Methods for the Performance Evaluation and Design of Broadband Multiservice Networks*. COST 242 Final report, 1996.

[CON 94] J.T. Connor, R.D. Martin and L.E. Atlas. Recurrent neural networks and robust time series prediction. *IEEE Transactions on Neural Networks*, 5:240–254, 1994.

[CRO 96] M.E. Crovella and A. Bestravos. Self-similarity in world wide web traffic. In *Proc. of the Sigmetrics'96*, pages 160–169, 1996.

[DAU 92] I. Daubechies. Ten lectures on wavelets. *Phildelphia: Society for Industrial and Applied Mathematics (SIAM)*, 1992.

[DAY 93] S.P. Day and M.R. Davenport. Continuous-time temporal backpropagation with adaptive time delays. *IEEE Transactions on Neural Networks*, 4:348–354, 1993.

[DEM 77] A. Dempster, N. Laird and O. D. RubinAgazzi. Maximum likelihood from incomplete data via the em algorithm. *J. Royal Stat. Soc. Series B*, 39:1–38, 1977.

[DRE 02] G. Dreyfus, J.M. Martinez, M. Samuelides, M.B. Gordon, F. Badran, S. Thiria et L. Herault. *Réseaux de Neurones : Méthodologie et Applications*. Editions Eyrolles, 2002.

[DUR 99] R.J. Duro and J. Santos Reyes. Discrete-time backpropagation for training synaptic delay-based artificial neural networks. *IEEE Transactions on Neural Networks*, 10:779–789, 1999.

[ELM 90] J.L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

[FEN 01] C. Feng and R. Plamondon. On the stability analysis of delayed neural networks. *Neural Networks*, 14:1181–1188, 2001.

[FOR 73] G.D. Forney. The viterbi algorithm. In *Proceedings of the IEEE*, volume 61, 1973.

[FRA 92] P. Frasconi, M. Gori and G. Soda. Local feedback multilayered networks. *Neural Computation*, 4:120–130, 1992.

[FUE 00] M. Fuentes. *Prédiction de Température de Surface à l'Aide de Réseaux de Neurones.* Mémoire de stage, ISIMA, Université Blaise Pascal, Clermont II, 2000.

[GAR 96] S. Garcia-Salicetti. *Une approche neuronale prédictive pour la reconnaissance en-ligne de l'écriture récursive.* Thèse de Doctorat, Université Paris 6, 1996.

[GEL 76] E. Gelenbe et G. Pujolle. The behaviour of a single-queue in a general queueing network. *Acta Informatica*, 7 :123–136, 1976.

[GEL 91] E. Gelenbe. Queuing networks with negative and positive customers. *J. Appl. Prob.*, 28 :656–663, 1991.

[GEL 99] E. Gelenbe et J-M.Fourneau. Random neural networks with multiple classes of signals. *Neural Computation*, 11(4) :953–963, 1999.

[GEL 02] E. Gelenbe et J-M.Fourneau. G-networks with resets. *Neural Computation*, 49 :179–191, 2002.

[GER 00] F.A. Gers, J. Schmidhuber and F. Cummins. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12:2451–2471, 2000.

[GIL 92] C.L. Giles, C.B. Miller, D. Chen, G.Z. Sun and Y.C. Lee. Learning and extracting finite state automata with second order neural networks. *Neural Computation*, 4(3):393–405, 1992.

[GOL 96] R.M. Golden. *Mathematical Methods for Neural Network Analysis and Design.* The M.I.T. Press, Cambridge, MA, 1996.

[GOU 97] C Goutte. *Statistical Learning and Regularisation for Regression.* Thèse de Doctorat, Université Paris 6, Paris, France, 1997.

[GRO 88] S. Grossberg. Nonlinear neural networks : Principles, mechanisms and architecture. *Neural Networks*, 1(1) :17–61, 1988.

[GUS 91] R. Gusella. Characterizing the variability of arrival processes with indexes of dispersion. *IEEE Journal on Selected Areas in Communications*, 9(2):203–211, 1991.

[HAB 96] I. W. Habib. Applications of neurocomputing in traffic management of ATM networks. *Proceedings of the IEEE*, 84(10):1430–1441, 1996.

[HAY 94] S. Haykin. *Neural Networks : A Comprehensive Foundation.* Prentice Hall, Upper Saddle River, NJ, 1994.

[HEB 49] D.O. Hebb. *The Organization of Behavior : A Neuropsychological Theory.* Wiley, New York, 1949.

[HER 91] J. Hertz, A. Krogh and R. Palmer. *An Introduction to the Theory of Neural Computation.* Redwood City, CA: Addison-Wesley, 1991.

[HIH 96] S El Hihi and Y. Bengio. Hierarchical recurrent neural networks for long-term dependencies. In G. Tesauro, D.S. Touretzky and T.K. Leen, eds, *Advances in Neural Information Processing Systems*, volume 8. Cambridge, MA:MIT Press, 1996.

[HIL 01] D. Hill, P. Coquillard, A. Aussem, J. de Vaugelas, T. Thibault and A. Meinesz. Modeling the ultimate seeweed expansion. *Simulation*, 76(2):118–126, 2001.

[HIR 95] A. Hiramatsu. Training techniques for neural network applications in ATM. *IEEE Communications Magazine*, 1995.

[HIR 01] K. Hirasawa and al. Improvement of generalization ability for identifying dynamical systems by using universal learning networks. *Neural Networks*, 14(10):1389–1404, 2001.

[HOC 97a] S. Hochreiter and J. Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

[HOC 97b] S. Hochreiter and J. Schmidhuber. Long short term memory. *Neural Computation*, 9:1735–1780, 1997.

[HOL 90] M. Holschneider et P. Tchamitchian. *Les Ondelettes en 1989*. Berlin : Springer-Verlag, 1990.

[HOL 94] U. Holst, G. Lindgren, J. Holst and M. Thuvesholmen. Recursive estimation in switching autoregressions with a Markov. *Journal of Time Series Analysis*, 77:257–287, 1994.

[HOP 82] J.J. Hopfield. Neural networks and physical systems with emergent collective abilities. *Proceedings of the National Academy of Sciences, USA*, 81:3088–3092, 1982.

[HUS 98] D. Husmeier and J.G. Taylor. Neural networks for predicting conditional probability densities: Improved training scheme combining em and rvfl. *Neural Networks*, 11(1):89–116, 1998.

[ISH 01] S. Ishii and M.A. Sato. Reconstruction of chaotic dynamics by on-line em algorithm. *Neural Networks*, 14:1239–1256, 2001.

[JIN 94] L. Jin, P.N. Nikiforuk and M.M. Gupta. Absolute stability conditions for discrete-time recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(6):954–964, 1994.

[JOR 92] M. Jordan. Constrained supervised learning. *Jounral of Mathematical Psychology*, 36:396–425, 1992.

[JOR 94] M.I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6:181–214, 1994.

[KHA 96] H.K. Khalil. *Nonlinear Systems*. Prentice-Hall, Upper Saddle River, NJ, 1996.

[KIL 94] R.A Kilmer. *Artificial Neural Network Metamodels of Stochastic Computer Simulations*. Ph.D. Thesis, University of Pittsburgh, CA, 1994.

[KOH 99] J. Kohlmorgen, Lemm S., Müller K.-R, Liehr S. and Pawelzik K. Fast change point detection in switching dynamics using a hidden Markov model of prediction experts. In *Proc. of the International Conference on Artificial Neural Networks*, pages 204–208, 1999.

[KRE 01] S.C. Kremer. Spatiotemporal connectionist networks: A taxonomy and review. *Neural Computation*, 13:249–306, 2001.

[KRI 93] V. Krishnamurthy and Moore J.B. On-line estimation of hidden Markov model parameters based on the kullback-leibler information measure. *IEEE Trasnactions on Signal Processing*, 41:2557–2573, 1993.

[LAN 88] K. Lang and G. Hinton. *Time-Delay Neural Network architecture for speech recognition.* rapport technique CMU-CS-88-152, Carnegie-Mellon University, Pittsburgh, 1988.

[LEE 00] S.-J. Lee and C.-L. Hou. A neural-fuzzy system for congestion control in ATM networks. *IEEE Transactions on Sysrtems, Man, and Cybernetics - Part B: Cybernetics*, 20(1), 2000.

[LEI 91] R. Leighton and B. Conrad. The autoregressive backpropagation algorithm. In *Proc. of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, 1991.

[LIA 94] J. Liang, P. Nikiforuk and M. Gupka. Absolute stability conditions for discrete-time recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(6):954–964, 1994.

[LIN 89] R. Linsker. How to generate ordered maps by maximizing the mutual information between input and outpu signals. *Neural Computation*, 1:401–411, 1989.

[LIN 96] T. Lin, B.G. Horne, P. Tino and C.L. Giles. Learning long-term dependencies in narx recurrent neural networks. *IEEE Transactions on Neural Networks*, 7:1329, 1996, 1996.

[LOR 63] E.N. Lorenz. Deterministic nonperiodic flow,. *J. Atmos. Sci.*, 20:130–141, 1963.

[MA 98] S. Ma and C. Ji. Fast training of recurrent networks based on the EM algorithm. *IEEE Transactions on Neural Networks*, 9(1):11–26, 1998.

[MAC 77] M. Mackey and L. Glass. Oscillations and chaos in phsysiological control systems. *Science*, 20:197–287, 1977.

[MAH 02] A. Mahul and A. Aussem. Distributed neural networks for QoS estimation in communication networks. *International Journal of Computational Intelligence and Applications, to appear*, 2002.

[MAK 99] M.W. Mak, K.W. Ku and Y.L. Lu. On the improvement of the real time recurrent learning algorithm for recurrent neural networks. *Neurocomputing*, 24:13–36, 1999.

[MAL 89] S. Mallat. A theory for multiresolution signal decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:674–693, 1989.

[MAN 95] M. Mangeas. *Propriétés Statistiques des Modèles Paramétriques Non-Linéaires de Prévision de Séries Temporelles.* Thèse de Doctorat, Université Paris 1, Paris, France, 1995.

[MAN 01] D.P. Mandic and J.A. Chambers. *Recurrent Neural Networks for Prediction.* John Wiley & Sons, Chichester, England, 2001.

[MEY 93] Y. Meyer. Wavelets: Algorithms and applications. *Phildelphia: Society for Industrial and Applied Mathematics (SIAM)*, 1993.

[MOH 95] W. M. Moh, M.-J. Chen N.-M. Chu and C.-D. Liao. Traffic prediction and dynamic bandwith allocation over ATM: a neural network approach. *Computer Communications*, 18(28):203–211, 1995.

[MUR 95] F. Murtagh, A. Aussem and M. Sarazin. Nowcasting astronomical seeing: towards an operational approach. *Publications of the Astronomical Society of the Pacific*, 107:702–707, 1995.

[MUR 96a] F. Murtagh and A. Aussem. New problems and approaches related to large databases in astronomy. In G.J. Babu and E.D. Feigelson, eds, *Statistical Challenges in Modern Astronomy II*, pages 123–133. Springer-Verlag, June 1996.

[MUR 96b] F. Murtagh, A. Aussem and O. Kardaun. The wavelet transform in multivariate data analysis. In A. Prat, ed, *Proc. of the COMPSTAT*, pages 397–402. Physica-Verlag, Heidelberg, 1996.

[MUR 97] F. Murtagh, A. Aussem and J.-L. Starck. Multiscale data analysis – information fusion and constant-time clustering. *Vistas in Astronomy*, 41:359–364, 1997.

[MUR 98a] F. Murtagh and A. Aussem. Using the wavelet transform for multivariate data analysis and time series forecasting. In K. Yajima C. Hayashi, H.H. Bock, ed, *Data Science, Classification and Related Methods*, pages 617–624. Springer-Verlag, 1998.

[MUR 98b] F. Murtagh, A. Aussem, J.-L. Starck, J. G. Campbell and G. Zheng. Wavelet-based decomposition methods for feature extraction and forecasting. In D. Vernon, ed, *Proc. of the OESI-IMVIP'98, Optical Engineering Society of Ireland and Irish Machine Vision and Image Processing Joint Conference*, pages 55–67. NUI Maynooth, 1998.

[MUR 98c] F. Murtagh, J. Campbell, G. Zheng, A. Aussem, M. Ouberdous, E. Demirov, W. Eifler and M. Crépon. Data imputation and nowcasting using clustering and connectionist modeling. In R. Payne and P. Green, eds, *Proc. of the International Conference on Computational Statistics*, pages 401–406. Springer-Verlag, Berlin, 1998.

[MUR 00] F. Murtagh, G. Zheng, J.G. Campbell and A. Aussem. Neural networks modelling for environmental prediction. *Neurocomputing*, 30:65–70, 2000.

[NAR 91] K.S. Narendra and K. Parthasarathy. Gradient methods for the optimization of linear control systems. *IEEE Transactions on Neural Networks*, pages 252–262, 1991.

[NOR 95] H. Brandtand E. Nordstöm and O. Gällmo et al. Na hybrid neural network approach to ATM admission control. In *Proceedings of International Switching Symposium (ISS'95)*, pages 283–287, April 1995.

[OJA 82] E. Oja. A simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15:267–273, 1982.

[OTT 93] E. Ott. *Chaos in Dynamical Systems.* Cambridge University Press, 1993.

[PAR 01] A.G. Parlos, S.K. Menon and A. Atiya. An algorithmic approach to adaptive state filtering using recurrent neural networks. *IEEE Transactions on Neural Networks*, 12(6):1411–1432, 2001.

[PEA 95] P.A. Pearlmutter. Gradient calculations for dynamica recurrent neural networks: A survey. *IEEE Transactions on Neural Networks*, 6(5):1212–1227, 1995.

[PER 95] M.P. Perrone. Putting it all together: Methods for combining neural networks. In G. Tesauro, D.S. Touretzky and T.K. Leen, eds, *Advances in Neural Information Processing Systems*, volume 7. Cambridge, MA:MIT Press, 1995.

[PIC 94] S.W. Piche. Steepest descent algorithms for neural network controllers and filters. *IEEE Transactions on Neural Networks*, 5:198–212, 1994.

[PIN 87] F.J. Pineda. Generalization of back-propagation to recurrent neural networks. *Physical Review Letters*, 59:2229–2232, 1987.

[PIN 89] F.J. Pineda. Recurrent back-propagation and the dynamical approach to adaptive neural computation. *Neural Computation*, 1:161–172, 1989.

[POG 90] T. Poggio and F. Girosi. Networks for approximation and learning. *Proc. of the IEEE*, 78:1481–1497, 1990.

[RAB 89] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceeding of the IEEE*, 77(2):257–285, 1989.

[REN 90] S. Renals and R. Rohwer. A study of network dynamics. *Journal of Statistical Physics*, 58:825–848, 1990.

[ROD 99] P. Rodriguez. A recurrent network that leans to count. *Connection Science*, 11(1), 1999.

[ROD 01] P. Rodriguez. Simple recurrent networks lean context-free and context-dependent languages by counting. *Neural Computation*, 13:2093–2118, 2001.

[RUM 86] D.E. Rumelhart, G.E. Hinton and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, eds, *Parallel Distributed Processing Explorations in the Microstructure of Cognition*, pages 318–362. Cambridge, MA:MIT Press, 1986.

[RYN 99] J. Rynkiewicz. Hybrid HMM/MLP models for time series prediction. In *Proc. of the European Symposium on Artificial Neural Networks, Bruges, Belgium*, pages 383–389, 1999.

[RYN 00] J. Rynkiewicz. *Modèles hybrides intégrant des réseaux de neurones artificiels à de smodèles chaîne de Markov cachées : application à la prédiction de séries temporelles.* Thèse de Doctorat, Université Paris 1, 2000.

[RYN 01] J. Rynkiewicz. Estimation of hybrid HMM/MLP models. In *Proc. of the European Symposium on Artificial Neural Networks, Bruges, Belgium*, pages 383–389, 2001.

[SAN 89] T.D. Sanger. Optimal unsupervised learning in a single layer feedforward neural network. *Neural Networks*, 12:459–473, 1989.

[SAS 94] P.S. Sastry, G. Santharam and K.P. Unnikrishnan. Memory neuron networks for identification and control of dynamical systems. *IEEE Transactions on Neural Networks*, 5:306–319, 1994.

[SCH 92] J. Schmidhuber. A fixed storage o(n^3) time complexity learning algorithm for fully recurrent continually running networks. *Neural Computation*, 4(2):243–248, 1992.

[SCH 96] J. Schmidhuber. Guessing can outperform many long time lag problems. *Neural Computation*, 1996.

[SCH 02] J. Schmidhuber, F. Gers and D. Eck. Learning nonregular languages: A comparison of simple recurrent networks and LSTM. *Neural Computation*, 14(9):2039–2041, 2002.

[SHE 92] M.J. Shensa. Discrete wavelet transforms: wedding the à trous and mallat algorithms. *IEEE Transactions on Signal Processing*, 40, 1992.

[SIN 89] S. Singhal and L. Wu. Training multilayer perceptron with the extended Kalman algorithm. In G. Tesauro, D.S. Touretzky and T.K. Leen, eds, *Advances in Neural Information Processing Systems*, pages 133–140. Cambridge, MA:MIT Press, 1989.

[SOH 01] W.-S. Soh and C.-K. Tham. Modular neural networks for multi-service connection admission control. *Computer Networks*, 36:181–202, 2001.

[SOL 00] S. Soltani, D. Boichu, P. Simard and S. Canu. The long-term memory prediction by multiscale decomposition. *Signal Processing*, 80:2195–2205, 2000.

[SON 96] E.D. Sontag. *Recurrent neural networks: Some system-theoretic aspects.* Dept. of Mathematics, Rutgers University, NB, 1996.

[SRI 94] B. Srinivasan, U.R. Prasad and N.J. Rao. Back propagation through adjoints for the identification of nonlinear dynamic systems using recurrent neural networks. *IEEE Transactions on Neural Networks*, 5:213–228, 1994.

[STA 94] J.-L. Starck and A. Bijaoui. Filtering and deconvolution by the wavelet transform. *Signal Processing*, 35, 1994.

[STA 95] J.-L. Starck, F. Murtagh and A. Bijaoui. Multiresolution support applied to image filtering and deconvolution. *Graphical Models and Image Processing*, 57, 1995.

[STA 96] J.-L. Starck, F. Murtagh and A. Bijaoui. *Image Processing and Data Analysis.* Cambridge University Press, 1996.

[STA 00] G. Stauffer et F. Moerel. *Prédiction de la Fluctuation du Seeing à l'Aide de Réseaux de Neurones.* Mémoire de projet, ISIMA, Université Blaise Pascal, Clermont II, 2000.

[STR 96] G. Strang and T. Nguyen. *Wavelets and Filter Banks.* Wellesley-Cambridge Press, 1996.

[SUN 92] G.-Z. Sun, H.-H Chen and Y.-C Lee. Green's function method for fast on-line learning algoprithm of recurrent neural networks. In G. Tesauro, D.S. Touretzky and T.K. Leen, eds, *Advances in Neural Information Processing Systems*, volume 4, pages 333–340. Cambridge, MA:MIT Press, 1992.

[TOO 91] N. Toomarian and J. Barhen. Adjoint-functions and temporal learning algorithms in neural networks. *Advances in Neural Information Processing Systems*, 1991.

[TOO 92] N. Toomarian and J. Barhen. Learning a trajectory using adjoint functions and teacher forcing. *Neural Networks*, 5(3):473–484, 1992.

[TRA 01] G. Tran. *Prédiction de la Fluctuation du Seeing à l'Aide de Réseaux de Neurones.* Mémoire de stage, ISIMA, Université Blaise Pascal, Clermont II, 2001.

[TSO 94] A.T. Tsoi and A.D. Back. Locally recurrent globally feedforward networks: a critical review of architectures. *IEEE Transactions on Neural Networks*, 5:229–239, 1994.

[VRI 92] B. de Vries and J. Principe. The gamma model - a new neural model for temporal processing. *Neural Computation*, 5:565–576, 1992.

[WAI 89] A. Waibel, T. Hanazawa, G. Hinton, J. Lang and K. Shikano. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustic Speech Signal Processing*, 37:328–339, 1989.

[WAN 93] E.A Wan. *Finite Impulse Response Neural Networks with Applications in Time Series Prediction.* Ph.D. Thesis, Stanford University, CA, 1993.

[WEI 90] A.S. Weigend, D.E. Rumelhart and B.A. Huberman. Predicting the future : A connectionist approach. *International Journal of Neural Systems*, 1:195–220, 1990.

[WER 74] P Werbos. *Beyond Regression: New Tools for Prediction and Analysis in Behavioral Sciences*. Ph.D. Thesis, Harvard University, Cambridge, MA, 1974.

[WER 90] P Werbos. Backpropagation through time: What it does and how to do it. *Proc. IEEE*, 78, Oct. 1990.

[WIL 89] R.J. Williams and D. Zipser. Experimental analysis of the real-time recurrent learning algorithm. *Connection Science*, 1:87–111, 1989.

[WIL 90] R.J. Williams and J. Peng. An efficient gradient-based learning algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501, 1990.

[WIL 95] W. Willinger, M. Taqqu, W.E. Leland and D. Wilson. Self-similarity in high-speed packet traffic. *Statistical Science*, 10(2):67–85, 1995.

[YU 97] H.J. Yu and S.Y. Bang. An improved time series prediction by applying layer-by-layer learning method to fir neural networks. *Neural Networks*, 10:1717–1729, 1997.

[ZEN 94] Z. Zeng, R.M. Goodman and P. Smyth. Discrete recurrent neural networks for grammatical inference. *IEEE Transactions on Neural Networks*, 5(2):320–330, 1994.

[ZHA 00] J. Zhang and B. Phaneendra. Global stability analysis in delayed hopfield neural network models. *Neural Networks*, 13:745–753, 2000.

[ZHE 97] G. Zheng, S. Rouxel, A. Aussem, J. Campbell, F. Murtagh, M. Ouberdous, E. Demirov, W. Eifler and M. Crépon. Forecasting of ocean state using satellite-sensor data. In P. Mc Kevitt F. Murtagh, J. Campbell, ed, *Proc. of the Irish Marine Vision and Image Processing Conferences and Eighth Irish Conference on Artificial Intelligence*, volume 1, pages 234–240. University of Ulster, 1997.