7/04

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de lé Recherche Scientifique

Ecole Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

Département d'Electronique

Projet de Fin d'Etudes
En vue de l'obtention de diplôme
d'**Ingénieur d'Etat en Electronique**

Modélisation de la fraction d'insolation de l'Algérie par Réseau de Neurones

Etudié par : ***GUERCHOUCHE Rachid***

Soutenu publiquement le 23 Juin 2004 devant le jury composé de :

Présidente :	M^elle ***M. GUERTI***	Maître de Conférences (ENP)
Examinateurs :	M. ***A. MALEK***	Directeur de Recherche (CDER)
	M^elle ***A. MOUSSAOUI***	Chargé de Cours (ENP)
Promoteurs :	M. ***M. S. AIT CHEIKH***	Chargé de Cours (ENP)
	M. ***A. BELOUCHRANI***	Maître de Conférences (ENP)
	M. ***M. HADDADI***	Professeur (ENP)

Ecole Nationale Polytechnique
El-Harrach, Alger
Juin 2004

المدرسة الوطنية المتعددة التقنيات
SIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

Modélisation de la fraction d'insolation de l'Algérie par Réseau de Neurones

Présenté par

GUERCHOUCHE Rachid

Proposé par

M. M. HADDADI

Encadré par

M. M. HADDADI
M. A. BELOUCHRANI
M. M. S. AIT CHEIKH

en vue de l'obtention du diplôme

d'Ingénieur d'Etat en Electronique

Laboratoire des Dispositifs de Communication et de Conversion Photovoltaïque

Ecole Nationale Polytechnique

Juin 2004

A la mémoire de Dada Ramdane

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

Remerciements

Je remercie mes parents : ils m'ont appris à travailler et donné le goût de progresser.

Mes remerciements sincères vönt aussi à Monsieur Mourad Haddadi qui m'a proposé le sujet de ce mémoire et qui a été pour moi un promoteur compréhensif et m'a accompagné avec une grande humanité.

Ce travail doit beaucoup à Monsieur A. Belouchrani qui m'a beaucoup conseillé et orienté vers les méthodes à utiliser. Merci à lui.

Monsieur M. S. Ait Cheikh a bien voulu contrôler la partie programmation de ce travail, qu'il en soit remercié, ainsi que Monsieur Z. Terra qui m'a beaucoup encouragé et aidé sur le plan formel.

Je tiens à exprimer mes vifs remerciements à Madame G. Voguet qui a revu ce document sur le plan linguistique et sur sa mise en forme.

Enfin que toutes celles et ceux qui m'ont soutenu durant cette année moralement et professionnellement soient associés à mes remerciements.

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

Avant-propos

Plusieurs travaux de recherche se sont intéressés aux séries temporelles comme moyen de caractériser un système observé et de prévoir ses futurs comportements. Dans ce travail nous passons en revue les techniques utilisées pour le traitement et la modélisation des systèmes à partir des séries temporelles issues des mesures réelles. Nous faisons l'étude d'un cas de prédiction et d'un autre de modélisation.

Après une introduction rapide aux séries temporelles, nous présentons sans grands détails les modèles linéaires dit de Box-Jenkins, dont nous montrons l'inefficacité et la défaillance. Nous passerons ensuite aux techniques non linéaires de prédiction des comportements futurs. Notre étude est basée sur la méthode des réseaux de neurones FIR. Nous examinons l'approche connexioniste comme moyen d'obtenir les barres d'erreurs de ces prévisions. L'application pour la prédiction est faite sur les données de la Santa Fe Competition, plus précisément sur les données d'un laser 81.5-micron 14Nh3 cw (FIR), pompé optiquement par une ligne P(13) d'un laser N2O via une transition vibratoire aQ(8.7) HN3.

Le dernier chapitre est consacré à une application des réseaux de neurones afin de modéliser les données de la durée d'insolation de l'Algérie, données issues des mesures de la NASA. L'étude faite dans ce chapitre - qui est la première du genre - a été proposée par Monsieur HADDADI, enseignant - chercheur au laboratoire des Systèmes de Communication et de Conversion Photovoltaïque du département d'Electronique de l'Ecole Nationale Polytechnique d'Alger.

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

ملخص :

تندرج هذه المذكرة في إطار دراسة و تحليل السلاسل الزمنية كوسيلة لتخصيص نظام فيزيائي أو طبيعي معين. بعد التطرق في مقدمة وجيزة إلى عموميات حول هذه السلاسل, قمنا بتقديم نماذج خطيّة و ركزنا على النماذج المسماة "Box-Jenkins", مع تبيّن عدم صلاحيّتها لدراسة الأنظمة. و كبديل استعملنا نظام لاخطي المتمثل في شبكة العصبونات (réseau de neurones), و ذلك بتطبيقه في عمليتين:

- إحداها التنبؤ بالقيم المستقبلية لسلسلة زمنية أخذت من معطيات ليزر في حالة عشوائية.
- أمّا الأخرى فهي البحث عن نموذج لمعطيات مدّة الرعن الخاصة بالجزائر, و من أجل ذلك طبّقنا شبكة العصبونات على طريقة تحليل المركبات الأساسية (Analyse de la composante principale)

الكلمات المفتاحية : شبكة العصبونات, تحليل المركبات الأساسية, مد ة الرعن.

Résumé:

Ce projet de fin d'études est une contribution à l'étude et l'analyse des séries temporelles comme moyen de caractériser un système physique ou météorologique. Après une brève étude des systèmes linéaires dits « Box-Jenkins », lesquels sont inefficaces pour la représentation de toute l'information ; nous avons opté pour l'étude non linéaire. La méthode choisie est les réseaux de neurones. Deux application ont été faites : la première est une prédiction des données d'un laser dans un état chaotique. La deuxième - qui est l'objet essentiel de ce mémoire – est une modélisation des données de la fraction d'insolation de l'Algérie.

Mots clés : réseaux de neurones ; analyse de la composante principale, fraction d'insolation.

Abstract:

This project is a contribution to the study and analysis of time series as tool of characterizing a physical or meteorological system. After a short study of the linear systems, we focused ourselves on models known as "Box-Jenkins", which are ineffective for the representation of all information; we chose the non linear study. The selected method is the neural networks. Two applications were made: the first is a prediction of a laser data in a chaotic state. The second – which is the essential object of this study – is a modeling of the Algerian data of insolation clearness index.

Key words: neural networks, non linear principal component analysis, Fraction of insolation.

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

SOMMAIRE

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

Introduction générale

Le désir de prédire l'avenir et de comprendre le passé a poussé aux recherches de lois qui expliquent le comportement d'un phénomène observé. Les exemples sont nombreux, s'étendant des irrégularités dans le battement du coeur au rapport de volatilité d'échanges monétaires. Si les équations tendancielles déterministes régissant le phénomène sont connues, elles peuvent être résolues pour prévoir le résultat d'une expérience sur le phénomène et cela en connaissant ou en imposant des conditions initiales. Pour faire des prédictions quand ces équations ne sont pas connues, il faut trouver les règles du comportement du système ainsi que son état à l'instant actuel.

Dans ce travail, nous nous concentrons sur les phénomènes dont les équations ne sont pas données; les règles qui gouvernent l'évolution du système doivent être tirées des régularités dans le passé.

Pour exemple, le mouvement d'un pendule ou le rythme de la portée d'une période dedans, nous donne le pouvoir de prédire les comportements futurs en connaissant ses oscillations sans faire recours à l'aperçu des mécanismes tendanciels. Nous utiliserons tout au long de notre étude les deux termes "comprendre" (ou en anglais "understanding") et "apprendre" (en anglais "learning") pour faire référence à deux approches complémentaires utilisées pour analyser une série temporelle non familière. *Comprendre* est basé sur un aperçu mathématique explicite du comportement du système et *apprendre* est basé sur des algorithmes qui peuvent émuler la structure du système dans une série. Plus spécifiquement on utilise la théorie d'information pour obtenir quelques aperçus du système, et les réseaux de neurones pour construire des modèles qui émulerons le comportement du système. Il faut remarquer ici l'utilisation du terme "émuler" au lieu de "simuler", ce qui signifie un remplacement des comportements réels du système par de nouveaux comportements qu'on trouve par modélisation. Notons aussi que dans les deux approches "comprendre" et "apprendre", le but est d'expliquer les observations passées du phénomène[1].

L'analyse des séries temporelles à trois buts : prédiction, modélisation et caractérisation. L'objectif de la *prédiction* est de donner des prévisions le plus exactes possible sur l'évolution du système à court terme, celui de la *modélisation* est de trouver une description qui détecte les particularités d'un comportement à long terme. Ces deux notions "prédiction" et "modélisation", même si elles semblent identiques, ne le sont pas : trouver les règles

[1] Le problème de l'utilisation des connaissances récentes d'un système dans le but de le contrôler et de produire des comportements désirés dans le futur a fait l'objet de plusieurs recherches, une publication intéressante est celle de "Narenda et Li 1995"; cette partie ne sera pas abordée dans notre étude.

qui gouvernent un système avec des propriétés correctes sur un long terme peut ne pas être le moyen le plus fiable pour déterminer les paramètres pour une prédiction à court terme, et vice-versa. Un modèle utile pour des prédictions à court terme peut induire des propriétés incorrectes dans le cas long terme. Le troisième but, la *caractérisation*, tente de déterminer, avec plus ou moins de connaissances, les propriétés fondamentales du système, comme par exemple, le nombre des degrés de liberté ou la quantité des variables aléatoires influentes sur le phénomène. Le concept de caractérisation peut se chevaucher avec celui de la prédiction, mais peut aussi différer : la complexité d'un modèle utile pour la prédiction peut ne pas être liée à la complexité actuelle du système.

Avant les années 1920, la prédiction se faisait par simple extrapolation dans le domaine temporel jusqu'à la valeur désirée. Le début de la prédiction "moderne" peut être fixé à 1927 quand YULE mit au point la technique autorégressive "modèle AR" dans le but de prédire le nombre annuel des taches solaire "sunspots". Son modèle prévoyait la future valeur comme une somme pondérée des anciennes valeurs déjà observées. Dans le but d'obtenir des prévisions intéressantes d'un système linéaire ou supposé linéaire, une intervention externe sur les influents externes du système doit être apportée. Pour le demi-siècle qui suivit YULE, le règne des paradigmes est resté valable pour les modèles linéaires bruités.

Toutefois, il y a des cas simples où ces paradigmes étaient inadéquats. Par exemple, une simple itération -comme dans le cas de la fonction logistique- peut générer un spectre de puissance large bande, qui ne peut pas être obtenu par une approximation linéaire. Une complication des séries temporelles peut-être générée par de très simples équations qui ont besoin d'une structure théorique plus générale pour une modélisation et une prédiction.

Deux développements cruciaux se sont produits aux environs de 1980. Les deux étaient possibles grâce à la disponibilité générale des calculateurs superpuissants, qui ont permit l'enregistrement des données de grande taille et des séries temporelles sur des intervalles très étendus. Beaucoup d'algorithmes compliqués ont été appliqués et les données comme les résultats pouvaient être visualisées interactivement. Le premier développement, "Space State Reconstruction by Time Delay Embedding", établi sur les idées de la topologie différentielle et la dynamique des systèmes, avait pour but de produire une technique de reconnaissance quand une série temporelle est générée par un système dont les équations gouvernantes sont déterministes, et si c'est le cas, pour comprendre la structure géométrique tendancielle du comportement observé. Le second développement est l'émergence du domaine de l'apprentissage des machines "machine learning", représenté par les réseaux de neurones, qui peuvent s'adapter pour explorer un espace large des modèles

potentiels.

Dans ce mémoire, le premier chapitre est une introduction aux séries temporelles, définitions, caractéristiques et modèles utilisés jusqu'à présent, ainsi qu'une présentation des données utilisées pour l'étude. Le chapitre 2 est consacré à un aperçu rapide sur la modélisation linéaire - spécifiquement les modèles Box-Jenins - dont nous donnerons les raisons de leur défaillance. Dans le Chapitre 3, le modèle réseau FIR sera détaillé comme exemple des modèles non linéaires, et sera appliqué aux données du laser. Dans ce chapitre, nous parlerons aussi des plus récentes méthodes d'étude des séries temporelles. Le chapitre 4 -qui est l'objet essentiel de cette étude- est une étude d'une série de données multivariables à des fins de modélisation. Nous allons essayer de trouver un modèle mathématique, pour chaque mois, de la durée d'insolation de l'Algérie. Pour cela une méthode non linéaire multivariable est utilisée, c'est la méthode dite "Non Linear Principal Component Analysis by Neural Network".

Chapitre 1: Généralités sur les series temporelles

1 Introduction:

Arts ou sciences? La prévision et la modélisation hésitent souvent entre ces deux pôles. De la science, elles tirent un ensemble de méthodes rigoureuses basées sur des algorithmes soigneusement décrits et dûment étalonnés sur le passé. De l'art, art de prévisionniste ou de modéliste cela s'entend, elles tirent la capacité de détecter parmi une multitude de données, toutes à peine significatives sur le plan statistique, celles qui seront porteuses des germes de l'avenir, et des pouvoirs de modélisation.

Autant les algorithmes sont automatisables (mais parfois avec difficulté), autant cet art demande une expertise difficilement transposable, du moins pour le moment, dans les algorithmes. Et puis, plus qu'une collection d'algorithmes, la modélisation (ou la prévision) est, avant tout, une démarche faite de rigueur utilisant les meilleurs algorithmes du moment, quels qu'ils soient. La rigueur impose, comme dans toute discipline scientifique, l'étalonnage systématique parce que plus que d'une prévision occasionnellement réussie ou d'une modélisation presque parfaite, l'utilisateur a besoin d'une évaluation des risques qui utilise forcément l'outil statistique.

Par ailleurs la longueur des séries disponibles n'autorise pas toujours une grande sophistication des algorithmes. Les séries longues et homogènes sont le royaume des algorithmes complexes avec moult paramètres et fonctions non linéaires où ils sont à peu près imbattables. Ces cas sont rarissimes. Les données sont rares et chères, même dans ce monde si informatisé. Si par hasard elles sont disponibles en grand nombre, elles sont rarement homogènes. Le prévisionniste et le modéliste disposent généralement des séries courtes et perturbées. Les données ne permettent pas de séparer les nombreux modèles, également plausibles, qui s'ajustent d'une manière similaire sur les données. De surcroît, les points aberrants troublent ces algorithmes pur sang qui se font facilement désarçonner. Ils (prévisionniste et modéliste) sont alors contraints d'utiliser un modèle simplifié qui s'ajustera cahin-caha mais dont il connaît tous les défauts.

2 Concepts de base:

Les méthodes de prévision et de modélisation s'appuient sur l'analyse de séries chronologiques, lorsque celles-ci sont disponibles et de longueur suffi-

santes, de bonne qualité et pertinentes. Ces trois conditions sont supposées être réunies le long de ce travail. Il faut cependant noter qu'une ou plusieurs de ces trois conditions peuvent être absentes.

La notion de série temporelle (appelée aussi série chronologique) forme le socle sur lequel repose les méthodes actuelles. Une série temporelle est une suite d'observations chiffrées, ordonnées par le temps. Le fait que ces observations soit ordonnées par le temps différencie nettement les méthodes d'analyse des séries chronologiques des autres méthodes de la statistique. Les concepts de base de la statistique, s'ils sont d'un grand sécours, devront être adaptés à ce cadre particulier. Ces notions seront d'autant plus utiles qu'on s'attache plus aux observations chiffrées et moins à la chronologie. Comme la statistique ordinale n'a été, jusqu'à présent, que d'un faible secours dans les méthodes d'analyse des séries temporelles, la chronologie s'analyse dans le cadre d'un modèle, c'est-à-dire une représentation simplifiée et néanmoins fidèle de la série.

Une série temporelle sera notée, très classiquement : x_t.

Le but de la modélisation est de trouver une boîte noire, dont l'entrée est la variable et la sortie est notre signal, dont on veut modéliser le comportement. Celui de la prédiction peut être énoncé comme suit : étant donné une séquence $x(1), x(2), ..., x(N)$ jusqu'au temps N, trouver les continuations $x(N+1), x(N+2), ...$La série peut résulter d'un échantillonnage d'un système continu ou tout simplement des mesures sur le système, et peut être d'origine stochastique ou déterministe. Il faut noter que la modélisation et la prédiction utilisent les mêmes techniques, mais une prévision vient toujours après une modélisation.

3 Les composantes d'une série temporelle:

Une série chronologique est, en général, un mélange de quelques composantes qui jouent un rôle-clé. Ces composantes, au nombre de quatre, sont la tendance à long terme, les fluctuation autour de cette tendance (cycle), la saisonnalité et les points irréguliers. Outre leurs aspects statistiques, ces composantes jouent un grand rôle pratique, l'essentiel du travail du prévisionniste se bornant à en analyser et prévoir une seule, par exemple la tendance.

1. **La tendance** : elle exprime la linéarité ou non de la série. Une série temporelle est linéaire si $x(t+N)$ peut être exprimée comme combinaison linéaires des valeurs historiques $f(x(t), x(t-1), ...)$, dans le cas contraire elle est dite non linéaire.

2. **Le cycle** : il est plus connu sous le nom de *stationnarité,* une série temporelle est stationnaire si elle possède une moyenne et une variance con-

stantes; dans le cas contraire, elle est dite non stationnaire. La non stationnarité est difficile à modéliser, et ses futurs comportements sont difficiles à prédire. Dans ce travail, nous nous consacrerons aux cas stationnaires, et également à quelques cas non stationnaires appelés "séries temporelles chaotiques stationnaires par morceaux" ou encore "piecewise chaotic time-series". Une série temporelle est dite *piecewise chaotic* si elle consiste en plusieurs régimes, si chaque régime correspond à un processus chaotique, et si la série globale est la collection d'une multitude de régimes chaotiques.

3. **Périodicité**, ou encore *saisonnalité*. Une série est périodique si elle présente un comportement à variation régulier. Mathématiquement, on écrit : $x(t)$ est périodique, de période τ si $x(t+\tau) = x(t)$. Dans ce travail, nous n'allons pas nous intéresser à cette caractéristique.

4. **Les points irréguliers** : connus sous le nom de *bruit aléatoire*, ils peuvent être présents dans une partie ou dans la totalité du spectre de fréquence de la série. Comme ce bruit ne peut pas être prédit, on se contente généralement d'étudier la série avec des fréquences bruits très grandes et de prédire les comportements du bruit en basses fréquences.

4 Les travaux récents sur les séries temporelles:

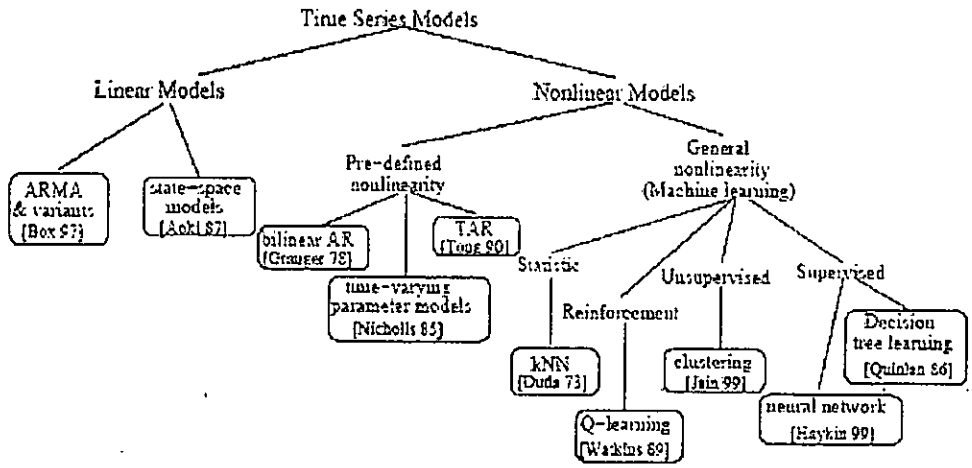


Figure 1: Classification des modèles pour la manipulation des série temporelles.

Dans les quatre décennies passées, plusieurs modèles ont été mis au point dans le but d'étudier les séries temporelles. Nous allons essayer ici de donner un bref aperçu des modèles les plus connus, et de leurs principaux inconvénients.

La figure 1 donne une classification des modèles existants [24], qui divise les modèles en deux grandes familles : linéaire et non linéaire; chaque famille, à son tour contient différents membres.

4.1 Linéarité:

Les modèles linéaires se répartissent en trois grandes classes :

1. **Box-Jenkins ARIMA** [2] et leurs variétés, par exemple le modèle Auto Rgressive (AR), le modèle Moving Average (MA) et celui Auto Regressive Moving Average (ARMA), qui décrivent les futures valeurs comme combinaison linéaire des valeurs historiques de la série et certains processus aléatoires.

2. **Lissage Exponentiel** [3], c'est un modèle qui fait un lissage des données $S(t)$ comme fonction des données brutes $x(t)$ avec :

$$S(t+1) = \alpha x(t) + (1-\alpha)S(t), \quad 0 \prec \alpha \preceq 1. \tag{1}$$

où α est le seul paramètre du modèle.

3. **Les modèles espace d'état** [1] qui représentent les entrées comme combinaison linéaire d'un ensemble de vecteurs d'état qui évoluent en temps selon quelques équations linéaires. Ces vecteurs ainsi que leur dimension sont généralement difficiles à choisir dans la pratique [6].

4.2 Non linéarité:

Les modèles non linéaires peuvent être classés en deux grandes catégories, ceux avec supposition d'une non linéarité prédéfinie et ceux avec une non linéarité générale. La première classe inclut les modèles *autorégressifs bilinéaires* [8], les modèles *à variation temporelle de paramètres "time-varying parameter models"* [21] et *"threshold auto-regressive models"* ou les mo-dèles *autorégressive à seuil* [23]. Il n'y a pas d'intérêt à modéliser une série temporelle lorsqu'on ignore le comportement non linéaire du système.

L'apprentissage des machines (machine learning) permet de manipuler les séries non linéaires car la machine apprend la non linéarité sans la supposer au début. Quelques modèles spécifiques peuvent modéliser une séquence temporelle en incluant un apprentissage statistique (exemple le modèle $k-$nearest-neighbors "kNN" ou $k-$plus proche voisin "$k-$ppv" [7]), un apprentissage par renforcement (comme le modèle $Q-$learning [28]), un apprentissage non supervisé (la méthode clustering [18]), et l'apprentissage supervisé (les arbres de décision [22] et les réseaux de neurones artificiels "ANNs" [9]). En général, ces méthodes apprennent en utilisant une série d'entraînements basée sur une

seule non linéarité objective. Comme résultat, elles utilisent plusieurs exemples pour les aider à éviter une optimisation locale, spécialement quand des méthodes basées sur un gradient sont utilisées [25,15]).

5 L'application:

Dans notre étude, l'application, pour la prédiction, se fera sur les données de la *Santa Fe competition* (données d'un laser). Ces données, ainsi qu'une multitude d'autres sont disponibles sur l'adresse Internet : *http://www.psych.stanford.edu/~andreas/Time-Series/SantaFe.html*

Les figures 1, 2 et 3 donnent trois exemples de séries temporelles issues de la *SantaFe competition.*

Pour la modélisation, nous essayerons de trouver un modèle mathématique pour chaque mois des données de la durée d'insolation de l'Algérie. C'est l'objet du dernier chapitre de ce travail (Chapitre 4). Cette étude est la première du genre, - contrairement à la prévision des données du laser qui a fait l'objet de plusieurs études, surtout dans le cadre de la *Santa Fe Competition.* Les données utilisées sont disponibles sur le site des données météorologique de la NASA.

Le chapitre 2 est consacré à un aperçu général des modèles linéaires, spécifiquement les modèles *Box-Jenkins.* Le chapitre 3, étudie la méthode des réseaux de neurones ainsi que son application aux données du laser. Nous nous sommes limités à l'étude des réseaux FIR.

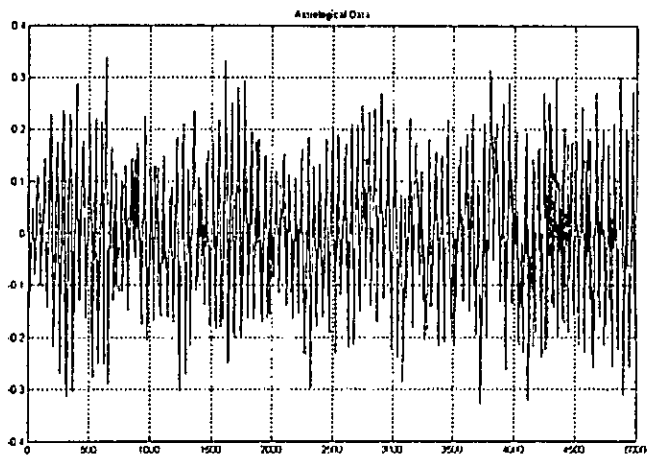


Figure 2: Des données astrologiques.

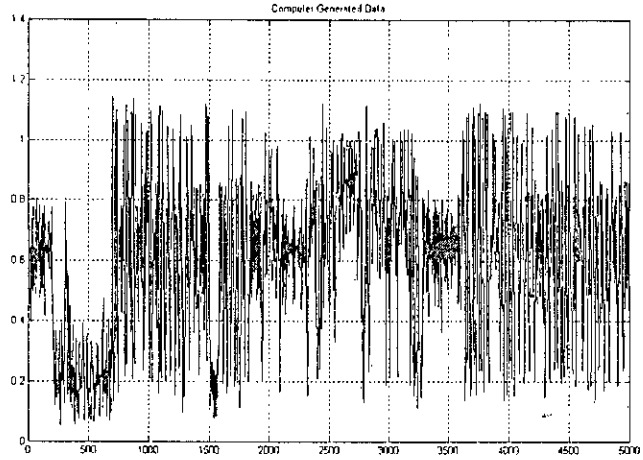


Figure 3: Des données générées par ordinateur.

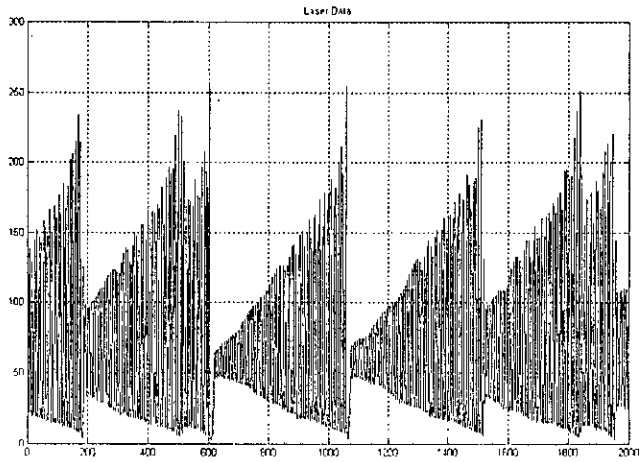


Figure 4: Les données du laser utilisées comme application à la prédiction.

6 Conclusion:

Dans ce chapitre, nous avons essayer de passer en revu les caractéristiques et les composantes essentielles des séries temporelles. Nous allons, dans le

chapitre suivant, aborder les modèles linéaires -spécifiquement ceux de Box-Jenkins- et nous démontreront à travers un exemple, leur inefficacité.

Chapitre 2: Modélisation linéaire

1 Introduction:

La modélisation linéaire des séries temporelles a deux grands avantages: le premier est que les modèles linéaires sont faciles à comprendre, en raison de leur simplicité et de leur didactique. Le deuxième avantage est la simplicité d'implémentation. Le prix à payer pour ces deux avantages est l'inefficacité des modèles linéaires pour la majorité des systèmes compliqués, surtout les plus récents. Dans ce chapitre nous allons voir les principales propriétés des modèles linéaires et nous essayerons d'expliquer pourquoi ils sont inefficaces. La littérature dans le domaine de l'analyse linéaire des séries temporelles est très vaste. La théorie de la prédiction linéaire a été tracée et fondée par Kolmogorov (1941) et Wiener (1949).

Dans cette analyse linéaire, deux aspects complémentaires vont être discutés: "comprendre" le comportement d'une série donnée, et "trouver" le modèle approprié à une cette série. Pour cela, on distingue les deux cas, celui où on introduit uniquement l'effet des entrées externes "Moving Average", et celui où l'effet de mémorisation est pris en compte "AutoRegressive".

2 Les modèles linéaires:

2.1 Le modèle MA:

Supposons une série de données $\{e_t\}$ qu'on appellera "entrée". On désire la modifier pour obtenir une autre série qu'on appellera "observée" et qu'on notera $\{x_t\}$. En supposant la linéarité du système et sa causalité, (causalité veut dire que la valeur de x n'est influencée que par la présente valeur et les N valeurs précédentes de l'entré e). La relation entre la sortie est l'entrée est donnée par :

$$x_t = \sum_{n=0}^{N} b_n . e_{t-n} = b_0 e_t + b_1 e_{t-1} + ... + b_N e_{t-N} \qquad (1)$$

Cette relation décrit un filtre de convolution : les nouvelles valeurs de x sont générées par un filtre dont les coefficients sont $b_0, b_1, b_2, ..., b_N$, tirés de la série e. Les statisticiens et les économistes appellent cela: "Nth-order moving average model", MA(N), ou encore "le modèle à moyenne ajustée d'ordre N". L'origine de cette appelation un peu confuse, peut être vue si on décrit un simple filtre de lissage qui "moyenne" les quelques valeurs passées

de la série e. En ingenierie, on appelle ce genre de filtre "*Finite Impulse Response*" FIR filter, ou filtre à Réponse Impulsionnelle Finie RIF. Cette appellation est due au fait que la sortie est forcée à aller vers zéro, N pas de temps après que l'entrée s'est annulée.

Rappel sur la Transformée en z:
La transformée en z permet de mieux comprendre la sortie des filtres, elle généralise la DFT "Transformée de Fourier Discrète", au plan complexe :

$$X(z) = \sum_{t=-\infty}^{t=+\infty} x_t z^t.$$

A l'intérieur du cercle unité, $z = \exp(-i2\pi f)$, la transformée en z traduit la DFT, et à son extérieur, elle mesure le rapport de la divergence ou de la convergence de la série.

Le calcul de la transformée en z de l'équation (1) donne :

$$\frac{X(z)}{E(z)} = b_0 + b_1 z^{-1} + b_2 z^{-2} + ... + b_N z^{-N} \tag{2}$$

$\frac{X(z)}{E(z)}$ est appelée la fonction de transfert du filtre FIR qu'on notera $H(z)$. Le problème revient à fixer un ordre N du filtre et à trouver les coefficients de la fonction de transfert, les b_i.

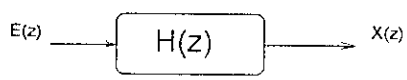


Figure 1: Schématisation d'un filtre FIR.

Les propriétés de la série de sortie x dépendent bien sûr de la série d'entrée e. La besogne est d'écrire le système indépendamment d'une séquence d'entrée spécifique. Pour un système linéaire, la réponse du filtre est indépendante de l'entrée.

2.1.1 Les trois caractérisations d'un modèle MA:

Dans cette section, nous allons exposer trois manières de caractériser le modèle MA. La première est dans le domaine temporel, c'est la réponse impulsionnelle du filtre; la seconde est dans le domaine fréquentiel, c'est le spectre, et la dernière est en termes des coefficients d'auto corrélation.

La réponse impulsionnelle: on suppose que l'entrée est non nulle seulement dans un pas de temps t_0 et s'éclipse pour les autres valeurs de t. La réponse (dans ce domaine) pour cette "impulsion" est tout simplement donnée par les b_i de l'équation (1). Dans chaque pas de temps l'impulsion se décale vers le nouveau coefficient, jusqu'à la disparition, après N pas de la sortie. La série $b_N, b_{N-1}, ..., b_0$ est donc la réponse impulsionnelle du système. La réponse pour une entrée arbitraire peut être calculée en super-imposant les réponses à des retards appropriés, pondérés par les entrées respectives (convolution). Ainsi la fonction de transfert décrit entièrement le système linéaire, c'est-à-dire un système possédant une superposition principale. La sortie est déterminée par la réponse impulsionnelle et l'entrée.

Le spectre: dans beaucoup de cas, il vaut mieux décrire le filtre dans le domaine fréquentiel. C'est plus simple et plus pratique, car la convolution dans le domaine temporel devient un produit dans le domaine fréquentiel. Si l'entrée d'un modèle MA est une impulsion, dont le spectre de puissance est uniforme, la transformée de Fourier discrète (DFT) de la sortie peut être donnée par :

$$\sum_{n=0}^{N} b_N \exp(-i2\pi nf) \tag{3}$$

Le spectre de puissance est donné par le carré de la magnitude de cette dernière expression :

$$\left|b_0 + b_1 e^{-i2\pi f} + b_2 e^{-i2\pi 2f} + ... + b_N e^{-i2\pi Nf}\right|^2 \tag{4}$$

Les coefficients d'autocorrélation: une autre manière, mais qui nous donne la même information, est d'utiliser les coefficients d'autocorrélation. On définit la moyenne par $\mu = \langle x_t \rangle$ et la variance par $\sigma^2 = \langle (x_t - \mu)^2 \rangle$, les coefficients d'autocorrélation sont alors donnés par :

$$\rho_\tau = \frac{1}{\sigma^2} \langle (x_t - \mu)(x_{t-\tau} - \mu) \rangle \tag{5}$$

La notation $\langle \cdot \rangle$ indique l'espérance de la valeur, (en statistique, l'espérance est désignée par $E\{\cdot\}$). Les coefficients d'autocorrélation décrivent le degré de covariance, en moyenne, de deux valeurs de la série qui sont séparées par un pas de temps τ, avec chacune des autres valeurs de la série. Si l'entrée du système est un processus stochastique avec des valeurs incorrolées dans le temps, $\langle e_i, e_j \rangle = 0$ pour $i \neq j$, tous les termes croisés disparaissent de l'espérance de l'équation (4), et les coefficients d'autocorrélation résultants sont donnés par :

$$\rho_\tau = \begin{cases} \frac{1}{\sum_{n=0}^{N} b_n^2} \sum_{n=\tau}^{N} b_n b_{n-|\tau|} & |\tau| \leq N, \\ 0 & |\tau| \succ N. \end{cases} \quad (6)$$

2.2 Le modèle AR:

Dans le modèle MA (ou FIR), les filtres opèrent dans une structure de boucle ouverte sans retour de la sortie. Ils peuvent transmettre uniquement une entrée qui leur est appliquée. Si on ne veut pas générer la série de l'extérieur, on a besoin de produire un retour d'état (ou contre réaction ou encore une mémoire), dans un but de générer les dynamiques internes de la série :

$$x_t = \sum_{m=1}^{M} a_m . x_{t-m} + e_t \quad (7)$$

Cela a pour nom Mth-autorégressive model AR(M), ou un modèle autorégressif d'ordre M, appelé aussi "*Infinite Impulse Response*" IIR filter ou filtre à Réponse Impulsionnelle Infinie RIF (car la sortie peut continuer après que l'entrée passe à zéro). Dépendant de l'application, e_t peut représenter soit l'entrée contrôlée du système, soit un bruit. Si e est en bruit blanc, les autocorrélations de la série de sortie x peuvent être exprimées en termes de coefficients du modèle.

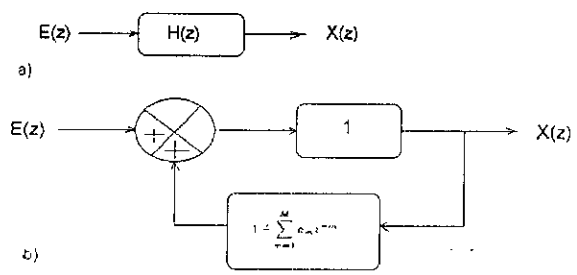


Figure 2: Schématisation du filtre IIR (a) : schéma global, (b) : en distinguant la boucle de retour.

De la même façon que précédemment, si on calcule la transformée en z de l'équation 7, on obtiendra la fonction de transfert $H(z) = \frac{X(z)}{E(z)}$ du filtre IIR :

$$H(z) = \frac{X(z)}{E(z)} = \frac{1}{1 - \sum_{m=1}^{M} a_m z^{-m}} \tag{8}$$

Ici aussi, le problème consiste à imposer l'ordre du filtre M et à trouver les coefficients a_m.

Dans ce modèle, plutôt que d'obtenir une seule équation pour chaque coefficient d'autocorrélation, on obtient un ensemble d'équations linéaires, chose due au couplage du feed-back des étapes passées. En multipliant l'équation (7) par $x_{t-\tau}$, en prenant les valeurs de l'espérance, et en normalisant, les coefficients d'autocorrélation d'un modèle AR sont trouvés par résolution de ces ensembles d'équations, appelés *les équations de Yule-Walker,*

$$\rho_\tau = \sum_{m=1}^{M} a_m \rho_{\tau - m}, \qquad \tau \succ 0. \tag{9}$$

Comme dans le cas du modèle MA, les coefficients d'autocorrélation doivent disparaître après M étapes. En prenant la transformée de Fourier des deux côtés de l'équation (7) et en réarrangeant les termes, on obtient :

$$lasortie = \frac{l'entrée}{1 - \sum_{m=1}^{M} a_m \exp(-i2\pi m f)} \tag{10}$$

Le spetre de puissance de la sortie est donc :

$$\frac{1}{|1 - a_1 \exp(-i2\pi 1 f) - a_2 \exp(-i2\pi 2 f) - ..... - a_M \exp(-i2\pi M f)|^2} \tag{11}$$

Pour générer une réalisation spécifique de la série, on doit toujours spécifier - par les M premières valeurs de la série x - les conditions initiales. Au-delà, le terme d'entrée e_t est crucial pour la survie d'un modèle AR. Si on n'a pas d'entrée, on sera forcément déçu par la série obtenue : dépendant de la quantité des feed-backs, après itération pour quelque temps, la sortie produite peut seulement tendre vers zéro, diverger ou osciller périodiquement[1].

[1] Dans le cas d'un modèle AR d'ordre 1 AR(1), cela peut être vu facilement : si la valeur absolue du coefficient est inférieure à l'unité, la valeur de x décroit exponentillement vers zéro; si elle est supérieure à l'unité, cette valeur explose exponentiellement. Pour les modèles d'ordre supérieur, le comportement à long terme est déterminé par le lieu des zéros du polynôme de coefficients a_i.

2.3 Le modèle ARMA:

Un modèle plus compliqué est celui où on introduit les deux parties AR(M) et MA(N). On le désigne par le modèle ARMA(M,N), qui est caractérisé par l'équation :

$$x_t = \sum_{m=1}^{M} a_m x_{t-m} + \sum_{n=0}^{N} b_n e_{t-n} \tag{12}$$

La sortie est plus simple à comprendre en utilsant la transformée en z, qui nous donne la fonction de transfert $H(z) = \frac{X(z)}{E(z)}$ du filtre, la sortie du filtre est donc:

$$X(z) = \frac{\sum_{n=0}^{N} b_n z^{-n}}{1 - \sum_{m=1}^{M} a_m z^{-m}} E(z) = \frac{B(z)}{1 - A(z)} E(z) \tag{13}$$

La transformée en z de l'entrée est multipliée par la fonction de transfert qui ne lui est pas reliée; la fonction de transfert s'éclipsera vers zéro pour le terme MA ($B(z) = 0$) et divergera pour les pôles ($A(z) = 1$) chose due au terme AR (sauf annulation par les zéros dans le numérateur).

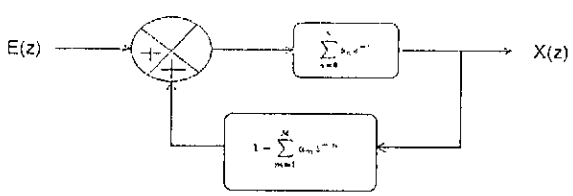


Figure 3: Schématisation de la réponse du filtre d'un modèle ARMA.

Puisque $A(z)$ est un polynôme complexe d'ordre M, et $B(z)$ est d'ordre N, il y aura M pôles et N zéros. Toutefois, la transformée en z produite par l'équation (12) d'une série temporelle peut se décomposer en une fonction rationnelle et un reste (probablement continu), qui est dû à l'entrée. Le nombre de zéros et de pôles détermine le nombre de degrés de liberté du système (c'est le nombre d'états précédents que le dynamisme du système retient). Il faut noter qu'on peut avoir plusieurs modèles ARMA, car ce sont M et N qui déterminent le modèle. Dans les cas extrêmes, un modèle AR

d'ordre fini peut être exprimé par un modèle MA d'ordre infini, et vice-versa. La figure 3 donne une illustration simple d'un filtre du modèle ARMA.

Pendant plus d'un demi-siècle, le modèle ARMA a dominé tout les domaines de l'analyse des séries temporelles et celui du traitement numérique du signal "discrete time signal processing". Par exemple, dans le domaine de la reconnaissance vocale et synthèse de la parole, le codage par prédiction linéaire "linear predictive coding" - plutôt que de transmettre le signal original - compressait la parole en transmettant les coefficients à variation lente d'un modèle linéaire (et probablement le reste de l'erreur entre le signal désiré et celui prédit linéairement). Si le modèle est bon, il transforme le signal en un petit nombre de coefficients plus un résidu considéré comme bruit.

3 Ajustement d'un modèle linéaire pour une série temporelle donnée:

3.1 Ajustement des coefficients:

L'ensemble des équations linéaires de Yule-Walker (Equation 9) nous permet d'exprimer les coefficients d'autocorrélation d'une série temporelle en termes des coefficients AR qui l'ont généré. Mais il existe une seconde lecture des mêmes équations: elles nous permettent d'estimer les coefficients d'un modèle AR(M) d'un signal considéré à partir des structures corrélées observées[2]. Une autre approche considère l'estimation des coefficients comme un problème de régression : en exprimant la prochaine valeur comme fonction des M valeurs précédentes, c'est-à-dire régresser linéairement x_t sur $\{x_{t-1}, x_{t-2,\ldots,}x_{t-M}\}$. Cela peut être fait en minimisant l'erreur quadratique : les paramètres sont déterminés tels que la différence au carré entre la sortie du modèle et la valeur observée, sommée sur tous les pas de temps dans la région d'ajustement, est la plus petite possible. Il n'existe pas d'expression conceptuelle simple comparable pour trouver les coefficients MA et ARMA à partir des données observées. Néanmoins, des techniques standard existent, qui sont efficaces en tant que procédures récursives.

Quoiqu'il n'y ait pas de raison à escompter qu'un signal arbitraire produit par un système puissent être écrit sous la forme de l'équation (12), il est raisonnable de tenter d'approximer une fonction de transfert d'un système

[2]En statistique, il est commun d'accentuer la différence entre un modèle estimé en utilisant des symboles différents, comme $\hat{a}$ pour désigner le coefficient estimé d'un modèle AR. Dans ce travail, nous allons utiliser indifféremment cette notation.

linéaire (la transformée en z) par un rapport de polynômes, c'est-à-dire un modèle ARMA.

3.2 Selection (ou ordre) du modèle:

Plus haut, nous avons discuté le problème de l'estimation les coefficients à partir des données pour un modèle ARMA d'ordre (M,N), mais nous n'avons pas abordé le choix de l'ordre du modèle. Il n'existe pas de meilleur choix unique, soit pour les valeurs ou pour le nombre des coefficients pour modéliser un ensemble de données. *"En majorant l'ordre du modèle, les erreurs de li-ssage diminuent, mais le test des erreurs des prédictions au-delà de l'ensemble des apprentissages donnera généralement des valeurs qui commenceront à augmenter en quelques points car le modèle adaptera un bruit externe au système.* Il existe plusieurs prescriptions pour trouver le "bon" ordre (comme le Akaike Information Criterion (AIC)). Mais le choix repose sur une supposition de la linéarité du modèle et de la distribution de laquelle on prélève le bruit. Quand il n'est pas clair que ces suppositions tiennent debout, une simple approche (mais probablement peu rentable en termes de données) est de conserver en aval quelques données d'apprentissage "training data" et de les utiliser pour évaluer les performances du modèle. Sélectionner l'ordre d'un modèle linéaire est un exemple spécifique du problème général de sélection du modèle. Ce problème réapparaîtra régulièrement et plus persistant encore dans le contexte de la modélisation non linéaire, car les modèles non linéaires sont plus flexibles et, de là, plus capables de modéliser les bruits hors rapport. Nous reviendrons sur ce problème de sur-ajustement dans la partie consacrée aux réseaux de neurones.

4. La défaillance des modèles linéaires:

Nous avons vu que les coefficients du modèle ARMA, le spectre de puissance et les coefficients d'autocorrélation contiennent la même information pour un système linéaire bruité par un bruit blanc incorrélé. Ainsi, si (et seulement si) le spectre de puissance est une caractérisation utile pour les faciès appropriés d'une série temporelle, le modèle ARMA sera le bon choix pour la description du système. Cette séduisante simplicité peut entièrement faillir pour de simples non linéarités; il suffit que le spectre de puissance se complique un peu. Deux séries temporelles ayant des spectres à large bande presque similaires mais qui peuvent être générées par deux systèmes dont les propriétés sont très différentes, comme par exemple, l'un drossé stochastiquement par un bruit externe, et le deuxième, selon une non linéarité déterministe (noise-free

ou bruit libre) avec un petit nombre de degrés de liberté. L'un des problèmes clé de cette question est de savoir comment faire la distinction entre ces deux cas. Les opérateurs linéaires déjà définis ne peuvent pas le faire.

Considérons deux exemples de non linéarité pour des systèmes en temps discret (comme le modèle AR, mais non linéaire):

▷Le premier exemple peut être esquissé comme étant le problème d'Ulman et von Neumann (1947): la future valeur de la série est puisée de la présente valeur renfermée dans une simple parabole :

$$x_t = \lambda x_{t-1}(1 - x_{t-1}) \tag{14}$$

Cette écriture est bien connue dans le contexte des "dynamiques des populations" comme un exemple type "des modèles mathématiques simples avec une dynamique très complexe". Il a été utilisé - à cause de l'universalité des diagrammes réguliers unimodes - pour décrire plusieurs systèmes contrôlés dans les laboratoires. Exemples : les écoulements hydrodynamiques et les réactions chimiques. Dans ce contexte, cette parabole est appelée *diagramme logistique* ou *diagramme quadratique*. La valeur de x_t dépend de la valeur précédente x_{t-1}; λ est un paramètre qui contrôle qualitativement le comportement de la série, rangé d'un point fixe (pour des petites valeurs de λ), jusqu'à des chaos déterministes. Par exemple, pour $\lambda = 4$, chaque itération détruit (ou crée selon la perspective) un bit d'information. Considérons que, en dessinant x_t en fonction de x, chaque valeur de x_t a deux prédécesseurs vraisemblablement égaux (ou réellement égaux), la pente moyenne (sa valeur absolue) est égale à deux : si on connaît le lieu avant l'itération dans une limite ε, on connaîtra ce lieu après l'itération dans une limite de 2ε. Cette exponentielle - qui s'accroît dans l'incertitude - est le poinçon des chaos déterministes ("divergence des trajectoires proches").

▷Le deuxième exemple est aussi simple que le premier. Considérons la série temporelle générée par l'équation :

$$x_t = 2x_{t-1} (\mathrm{mod}\, 1) \tag{15}$$

La compréhension du sens et du comportement de cette équation est simple, il suffit de considérer la position x_t écrite en forme de fraction binaire expansée (c'est-à-dire $x_t = 0.d_1d_2.... = (d_1 \times 2^{-1}) + (d_2 \times 2^{-2}) + ...$) : chaque itération décale les bits

d'une case vers la gauche ($d_i \leftarrow d_{i+1}$). Cela veut dire que le bit le plus significatif d_1 est débarrassé et qu'un bit de l'expansion binaire de la condition initiale est révélé. Cette structure peut être facilement implémentée dans un système physique consistant en un classique "billiard ball" en des surfaces de réflexion, où les x_t sont les positions successives dans lesquelles la balle a traversé une ligne donnée.

Les deux systèmes sont complètement déterministes (leurs évolutions sont complètement déterminées par la condition initiale x_0). Ils peuvent facilement générer des séries temporelles avec des spectres de puissance à large bande. Dans le contexte du modèle ARMA la composante à large bande dans le spectre de puissance de la sortie doit provenir d'une entrée constituée par un bruit externe au système, mais ici elle apparaît dans deux systèmes à une seule dimension aussi simple qu'une parabole et deux lignes droites. Les non linéarités sont essentielles pour reproduire les comportements intéressants dans un système déterministe. L'intérêt ici est que même une simple non linéarité suffise. Dans le but de capter des non linéarités, on doit lâcher la supposition tenace d'un modèle linéaire et considérer une large classe de solides (ou de faible) modèles.

Les modèles solides possèdent de fortes suppositions. Ils sont généralement exprimés en quelques équations avec peu de paramètres, et peuvent souvent expliquer une pléthore de phénomènes. De l'autre côté il y a les modèles faibles, qui s'appliquent seulement pour quelques domaines où les suppositions ne sont pas immuables. Pour compenser le manque en matière d'explicitation des connaissances, les modèles faibles contiennent habituellement beaucoup plus de paramètres (ce qui peut compliquer une interprétation claire). Il peut être utile de conceptualiser des modèles dans un espace à deux dimensions constitué par les axes *(pauvre-données)* $\leftrightarrow$ *(riche-donnée)* et *(théorie-pauvre)*$\leftrightarrow$*(théorie-riche)*. En raison de l'expansion dynamique de l'aptitude à l'acquisition automatique des données et de traitement, il est de plus en plus possible de risquer dans le domaine de "*la pauvre-théorie et des données-riches*". La prémisse de l'approche décrite dans cette étude est que nous n'avons pas "*des principes premiers*" concernant la série temporelle observée qu'on peut commencer à former.

5 Application aux données du laser:

Nous allons maintenant présenter une application des modèles linéaires aux données du laser, et nous verrons que la qualité de la prédiction est médiocre.

Nous allons utiliser les fonctions prédéfinies de MATLAB afin de simuler le problème.

La fonction *lpc* détermine les coefficients d'un prédicteur linéaire en avance, en minimisant l'erreur de prédiction dans le sens du moindre carré. Elle trouve son application dans le design des filtres et le codage de la parole. La fonction $[a, g] = lpc(x, p)$ trouve les coefficients d'un prédicteur linéaire d'ordre p (Filtre FIR), qui prévoit la valeur actuelle de la série temporelle réelle en se basant sur les valeurs passées:

$$\hat{x}(n) = -a2x(n-1) - a(3)x(n-2) - \cdots - a(p+1)x(n-p). \qquad (16)$$

p est l'ordre du filtre polynomial de prédiction, $a = [1\ a(2)\ \ldots\ a(p+1)]$. Si p n'est pas spécifié, *lpc* utilise par défaut $p = length(x) - 1$.

Différents choix de p ont permis de donner différents résultats, mais l'erreur est assez considérable dans tous les cas.

La figure 4 donne les 100 continuations des données du laser avec des filtres d'ordre 1, 25 et 999; le trait continu représente la série originale, le trait discontinu, celle prédite. La figure 5 donne les erreurs d'estimation pour les trois cas. Nous constatons bien que l'erreur reste considérable, quel que soit le choix de p.

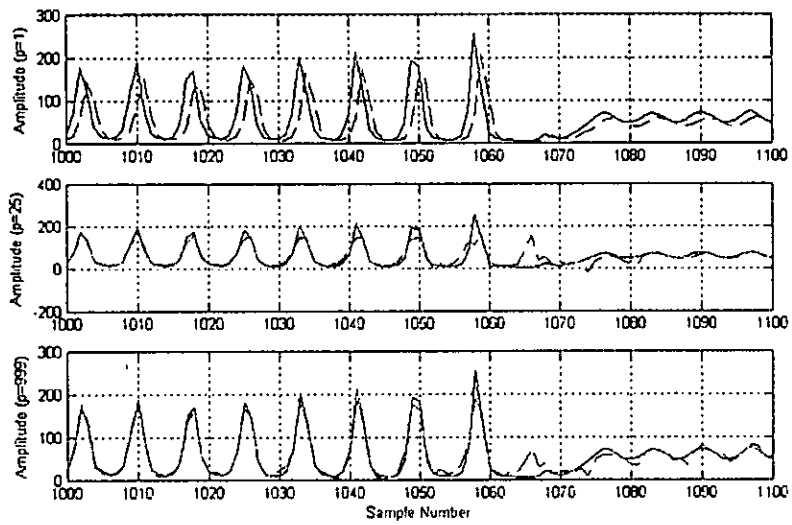


Figure 4: Prédiction en utilisant la fonction LPC, pour les 3 ordes: 1, 25 et 999.

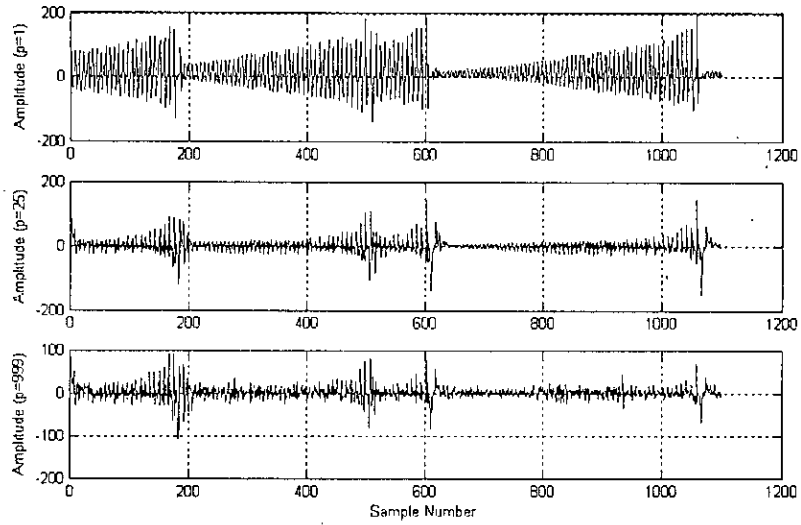


Figure 5: Les erreurs d'estimation.

6 Conclusion:

L'étude linéaire des séries temporelles ne permet pas de renseigner et de représenter toute l'information. Nous avons donné quelques exemples où la modélisation linéaire est inefficace, et nous avons présenté un exemple pratique de la défaillance d'une technique linéaire pour la prédiction.

Comme alternative aux modèles linéaires, nous allons continuer notre travail par des méthodes linéaires. Dans le chapitre suivant, le réseau de neurone FIR est utilisé pour la prédiction des données du laser. Dans le dernier chapitre, l'analyse non linéaire de la composante principale par un réseau perceptron neurones multicouches est utilisés pour modéliser la fraction d'insolation de l'Algérie.

Chapitre 3: Réseaux de Neurones FIR pour la prédiction autorégressive des séries temporelles

1 Introduction:

Nous avons vu que le but de la prédiction des séries temporelles est principalement de trouver - à partir d'une séquence $y(1)$, $y(2)$, ..., $y(N)$ dans N étapes de temps - les continuations $y(N+1)$, $y(N+2)$, ... La série peut provenir d'un système régi par des dynamismes d'origine déterministe ou aléatoires. L'approche standard de la prédiction est de construire un modèle qui donne les continuations de la séquence observée. Dans le premier chapitre, nous avons vu la méthode qui a été longuement utilisée et qui date de Yule, la méthode autorégressive linéaire (AR) qui ajuste les données de telle sorte que:

$$y(k) = \sum_{n=1}^{N} a(n)y(n-k) = \hat{y}(k) + e(k). \qquad (1)$$

Ce modèle AR construit $y(k)$ comme une somme pondérée des valeurs précédentes de la séquence. La valeur prédite de $y(k)$ est donnée par $\hat{y}(k)$; le terme d'erreur $e(k) = y(k) - \hat{y}(k)$; est généralement supposé être un bruit blanc pour une analyse dans un cadre stochastique.

Des techniques plus modernes utilisent une prédiction non linéaire. Dans ce qui suit, les réseaux de neurones sont utilisés comme alternative aux modèles linéaires dont on a vu la défaillance. La forme de base $y(k) = \hat{y}(k)+e(k)$ est retenue, toutefois, l'estimation $\hat{y}(k)$ est prise comme étant la sortie $\aleph$ d'un réseau de neurones conduit par les valeurs passées de la séquence. Cela s'écrit comme suit :

$$y(k) = \hat{y}(k) + e(k) = \aleph[y(k-1), y(k-2), ..., y(k-N)] + e(k). \qquad (2)$$

Ce modèle est appliqué pour des séquences scalaires (séries temporelles uni variantes), ou vectorielles (séries temporelles multi variables).

L'utilisation de cette autorégressivité non linéaire s'explique ainsi :

> ► Pour une large classe de systèmes dynamiques, il existe un diffeomorphisme (diagramme différentiel pas à pas) entre une fenêtre finie de la série $[y(k-1), y(k-2), ..., y(k-N)]$ et l'*état*

gouvernant du système dynamique. Cela implique l'existence, en théorie, d'une autorégression non linéaire de la forme $y(k) = g[y(k-1), y(k-2), ..., y(k-N)]$, qui modélise exactement la série (en assumant l'absence de bruit). Ainsi, le réseau de neurones forme une approximation de la fonction idéale $g(.)$.

► Il a été démontré qu'un réseau de neurones Feed-Forward ℵ avec un nombre arbitraire de neurones est capable d'approximer toute fonction *uniformément* continue.

L'utilisation des réseaux de neurones dans la prédiction des séries temporelles ne date pas d'hier. Plusieurs travaux ont été faits. L'entrée du connexionnisme dans la *SFI competition* est témoin du succès et de l'importance des réseaux dans ce domaine. Dans l'étude nous intéressant, nous nous baserons sur une autorégression non linéaire en utilisant un réseau à réponse impulsionnelle finie (FIR). Nous présenterons tout d'abord la structure du réseau FIR et son algorithme adapté appelé Temporal BackPropagation. Nous discuterons ensuite l'utilisation du réseau dans une configuration de prédiction. Les résultats de la SFI competition sont ensuite présentés comme application de ce réseau.

2 Le modèle réseau FIR:

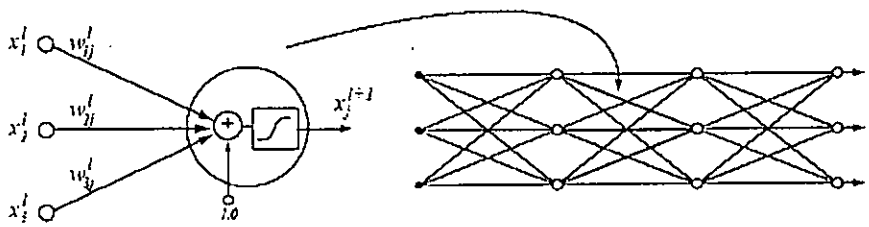


Figure1: Modèle d'un neurone statique et d'un réseau Feed-Forward: chaque neurone passe la somme pondérée de ses entrées à travers une fonction sigmoïde. La sortie d'un neurone d'une couche donnée est une entrée pour les neurones de la couche suivante. Dans l'illustration du réseau, chaque ligne représente une connexion synaptique. Pas de feedback dans cette strusture.

Le modèle du réseau de neurones multicouches traditionnel est illustré par la figure 1. Il est composé d'un arrangement de couches de neurones artificiels où chaque neurone d'une couche donnée alimente tous les neurones de la couche suivante. Un neurone singulier extrait de le $l^{ième}$ couche d'un réseau à L couches est représenté sur la figure. L'entrée x_i^l du neurone est

multipliée par un coefficient variable $\omega_{i,j}^l$ appelé *poids*, il représente la connexion synaptique entre le neurone i dans la couche précédente et le neurone j dans la couche l. La sortie d'un neurone, x_j^{l+1}, est tout simplement prise comme étant une fonction sigmoïde [1] de la somme pondérée de ses entrées :

$$x_j^{l+1} = f\left(\sum_i \omega_{i,j}^l x_i^l\right). \tag{3}$$

Une entrée au neurone – *le biais* – est réalisé en fixant x_0^l à 1. La structure du réseau est complètement définie en prenant x_i^0 comme étant les entrées externes, et x_i^L comme les sorties finales du réseau. L'apprentissage du réseau peut être réalisé en utilisant l'algorithme familier connu sous le nom de *rétro-propagation* (en anglais *backpropagation*).

Le modèle du réseau Feed-Forward décrit précédemment forme une architecture compliquée des entrées de la première couche aux sorties de la dernière couche. Néanmoins, c'est une architecture *statique*, où il n'y a pas de dynamismes internes. Une modification du neurone de base est réalisée en remplaçant chaque poids synaptique statique par un filtre linéaire FIR [2]. Par FIR, on veut dire que pour chaque entrée excitatrice de durée finie, la sortie du filtre sera aussi de durée finie. Le filtre FIR le plus simple peut être modélisé par des lignes à unités de retard[3] comme le montre la figure 2. Pour ce filtre, la sortie $y(k)$ correspond à une somme pondérée des valeurs passées **retardées** de l'entrée :

$$y(k) = \sum_{n=0}^{N} w(n)x(k-n). \tag{4}$$

Notons que cela correspond à la composante MA d'un simple modèle ARMA. En effet, le filtre FIR fut l'un des premiers éléments de base adapté et étudié. D'un point de vue biologique, le filtre synaptique représente un modèle de Markov de la transmission du signal correspondant au processus du transport axonal, modulation synaptique, et dissipation de charge dans une membrane cellulaire.

[1] La sigmoïde est une fonction continue en forme de S choisie pour modéliser en gros les propriétés limites d'un neurone réel. Mathématiquement, elle est généralement la fonction tangent hyperbolique, *tanh*.

[2] Le cas d'un filtre à réponse impulsionnelle infinie (IIR) ne sera pas abordé dans notre étude.

[3] En anglais "tapped delay line", en français, nous garderons l'appellation : lignes à unités de retard

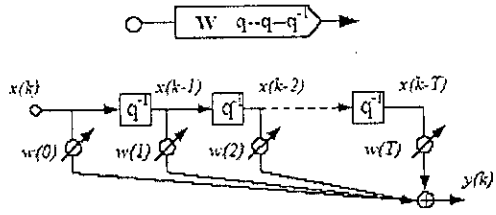


Figure 2: Modèle du filtre FIR: Une ligne à unités de rertard montre le modèle fonctionnel d'un "synapse" à réponse impulsionnelle finie (q^{-1} représente un opérateur d'unité de retard, c'est-à-dire $x(k-1) = q^{-1}x(k)$).

Continuons avec les notations; les coefficients pour le filtre synaptique connectant le neurone i au neurone j dans la couche l sont spécifiés par le vecteur $\mathbf{w}_{i,j}^l = [w_{i,j}^l(0), w_{i,j}^l(1), ..., w_{i,j}^l(N^l)]$. De manière similaire, $\mathbf{x}_i^l(k) = [x_i^l(k), x_i^l(k-1), ..., x_i^l(k-N^l)]$ dénote le vecteur des états retardés le long du filtre synaptique. Cela nous permet d'exprimer l'opération d'un filtre par un produit vectoriel $\mathbf{w}_{i,j}^l \cdot \mathbf{x}_i^l(k)$[4]. La sortie $x_j^{l+1}(k)$ d'un neurone dans la couche l au temps k est maintenant prise comme la fonction sigmoïde de la somme de toutes les sorties du filtre qui alimentent le neurone (Figure 3).

$$x_j^{l+1}(k) = f\left(\sum_i \mathbf{w}_{i,j}^l \cdot \mathbf{x}_i^l(k)\right). \tag{5}$$

Noter la similitude frappante dans l'apparence entre ces équations et celles du modèle statique (Equation 3) avec leurs figures associées. Du côté notation, les scalaires sont remplacés par des vecteurs et la multiplication par un produit vectoriel. L'opération de convolution des synapses est implicite dans la définition. Comme on va le voir, ces simples analogies portent à travers quand on compare la *rétropropagation standard* des réseaux statiques à la *rétropropagation temporelle* des réseaux FIR.

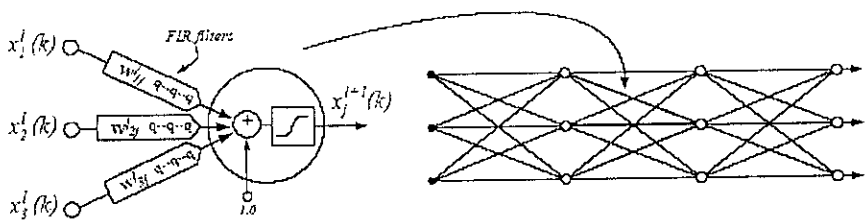


Figure 3: Le neurone et le réseau FIR: Dans le modèle temporel du neurone, les signaux d'entrée passent par des filtres synaptiques. La somme des entrées filtrées passe à travers une fonction sigmoïde pour former la sortie du neurone. Dans le réseau Feed-Forward, toutes les connexions sont modélisées par des filtres FIR.

[4]Dans cette notation, la relation temps est implicite.

2.1 Une représentation alternative des topologies FIR:

Une structure réseau incorporant des temps retards plongés est dite *Time-Delay Neural Network* (TDNNs) ou *réseau de neurones à temps retardés.* TDNNs est devenue populaire pour son utilisation réussie dans la classification des phénomènes. Une TDNNs est typiquement décrite comme étant un réseau à couches dans lequel les sorties d'une couche sont amorties en plusieurs étapes temps et donc alimentant entièrement la couche suivante (voir figure 4). En effet, les réseaux TDNN et FIR peuvent être assumés comme fonctionnellement équivalents, les différences entre les topologies sont dues à leurs représentations illustrées et au formalisme additif dans la notation pour les réseaux FIR. En plus, le réseau FIR est plus simple à approcher d'un réseau multicouche standard comme une simple extension temporelle ou vectorielle. Comme on va le voir, la représentation FIR est aussi menée à plus d'adaptation et d'arrangements désirables pour un apprentissage non linéaire.

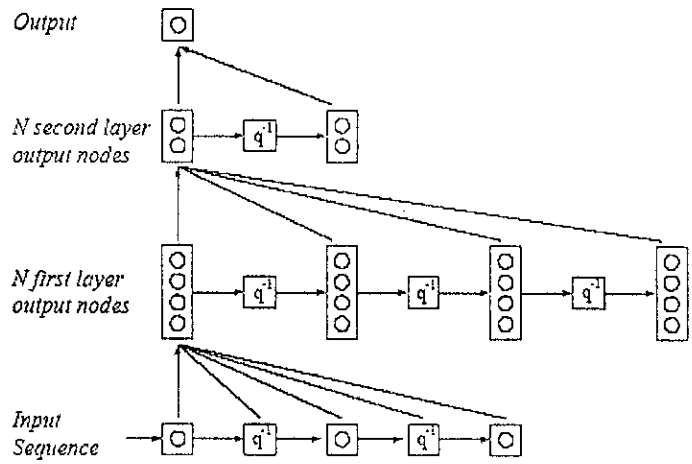


Figure 4: Time Delay Neural Network: Tous les noeuds de sortie dans une couche donnée sont mémorisés sur plusieurs étapes de temps. Ensuite, les sorties et les états mémorisés sont entièrement connectés pour alimenter la prochaine couche. Cette structure est fonctionnellement équivalente à un réseau FIR.

Une représentation alternative du réseau FIR (et TDNN) peut être trouvée en utilisant une technique référée comme *déploiement dans le temps.* La stratégie générale est d'enlever tous les retards temps en agrandissant le réseau sur une structure statique équivalente plus large. Comme exemple, considérons le réseau très simple de la figure 5a. Le réseau est constitué de trois couches avec un neurone de sortie unique et deux neurones dans chaque

couche cachée. Toutes les connexions sont faites par des synapses du second ordre (deux unités de retard). Donc, quand il y a uniquement 10 synapses dans le réseau, il y aura un total de 30 variables – les coefficients des filtres. Commençant par la dernière couche, chaque retard est interprété comme « un neurone virtuel » à qui l'entrée est retardée d'un nombre approprié de fois. Un retard est ensuite « enlevé » en repliant la couche précédente du réseau et en conséquence, il faut effacer l'entrée du réseau (Figure 5b). Le procédé est poursuivi vers l'arrière à travers chaque couche jusqu'à ce que tous les retards soient enlevés. Le réseau final dévoilé est illustré sur la Figure 5c.

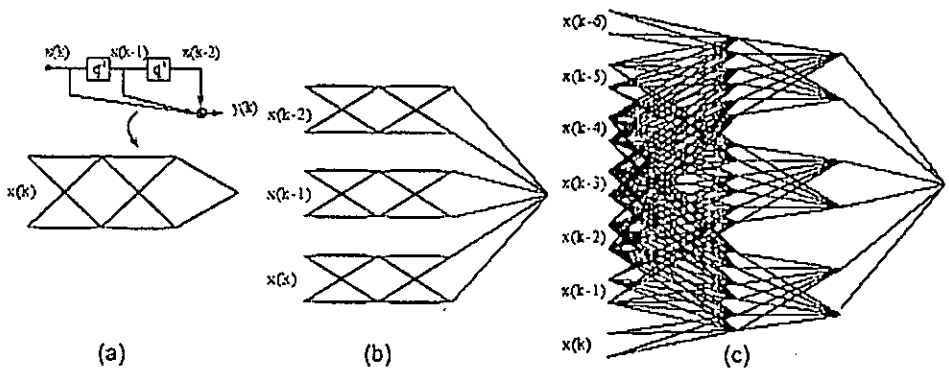


Figure 5: Un réseau FIR avec des unités de retard du $2^{ème}$ ordre pour toutes les connexions est déployé dans un réseau statique contraint. La structure d'origine contient 30 coefficients filtre variables tandis que le réseau résultant a 150 synapses statiques.

Cette méthode produit une structure statique contrainte équivalente où la dépendance temps a été faite extérieurement au réseau. Noter que tandis qu'il y a eu initialement 30 coefficients de filtre, la structure équi-valente dévoilée a 150 synapses statiques. Cela peut être vu comme le résultat d'une redondance dans les coefficients. En effet, la taille du réseau statique équivalent augmente *géométriquement* avec le nombre de couches et le nombre de retards (voir Tableau 1). Dans ce cadre là, on peut voir un réseau FIR comme une représentation compacte d'un réseau statique plus large avec des symétries imposées. Les symétries forcent le réseau à subdiviser l'échantillon d'entrée sur de recouvrantes régions locales. Chaque région est identiquement traitée avec le résultat qui a été combiné, successivement, à travers les couches subséquentes du réseau. Cela est en opposition avec un réseau entièrement connecté qui tente d'analyser toute la scène en une seule fois. Des contraintes symétriques locales similaires ont été un motif pour l'utilisation dans la classification des échantillons des réseaux dits « à poids partagé ».

Dimension du réseau		Paramètres variables	l'Equivalent statique
Noeuds*	Ordre**		
$2 \times 2 \times 2 \times 1$	$2:2:2$	30	150
$5 \times 5 \times 5 \times 5$	$10:10:10$	605	36,355
$3 \times 3 \times 3$	$9:9$	180	990
$3 \times 3 \times 3 \times 3$	$9:9:9$	270	9,900
3^n	9^{n-1}	$(n-1)90$	$10^n - 10$

Tableau 1: Le réseau FIR, l'Equivalent Statique.
* : Nombre d'entrées×Neurones cachés×Sorties.
** : Ordre du synapses FIR dans chaque couche.

3 Adaptation: Rétropropagation Temporelle:

Etant donné une séquence d'entrée $x(k)$, le réseau produit la séquence de sortie $y(k) = \aleph[\mathrm{W}, x(k)]$, où W représente l'ensemble de tous les coefficients des filtres dans le réseau. Pour le moment, supposons qu'à chaque instant dans le temps une sortie désirée $d(k)$ est fournie au réseau (nous allons reformuler cela en termes de prédiction des séries temporelles dans la section prochaine). L'erreur instantanée est définie par $e^2(k) = \|d(k) - \aleph[\mathrm{W}, x(k)]\|^2$ comme étant la distance euclidienne entre les sorties du réseau et les sorties désirées. L'objectif d'un apprentissage est de minimiser, en plus de W, la fonction coût "the cost function" :

$$C = \sum_{k-1}^{K} e^2(k), \tag{6}$$

où la somme est prise sur l'ensemble des points K de la séquence d'apprentissage. Les termes de régularisation (e. g, contraintes sur W) ne sont pas décrits dans cette étude. La méthode la plus directe avancée pour minimiser C est *le gradient stochastique déscent.* Les filtres synaptiques sont mis à jour à chaque incrémentation temps selon la règle:

$$\mathrm{w}_{ij}^{l}(k+1) = \mathrm{w}_{ij}^{l}(k) + \eta \frac{\partial e^2(k)}{\partial \mathrm{w}_{ij}^{l}(k)}. \tag{7}$$

où η contrôle le taux d'apprentissage.

La façon la plus évidente pour obtenir les termes du gradient nécessite premièrement un déroulement de la structure dans son équivalent statique, puis d'appliquer une rétropropagation standard. Malheureusement, cela mène vers un algorithme ayant des caractéristiques indésirables. La rétropropagation appliquée à un réseau statique permet de trouver les termes du gradient

associés à chaque poids dans le réseau. Comme le réseau contraint contient des poids "duplicata", les termes individuels du gradient doivent être soigneusement et tardivement recombinés pour trouver le gradient total pour chaque coefficient unique du filtre. Un traitement local distribué, considéré comme une comptabilité globale, devient nécessaire afin de suivre tous les termes; il n'y a pas de formule de récurrence simple possible. Ces inconvénients sont identiques pour la structure TDDN [5].

Un algorithme plus attirant peut être dérivé si on approche le problème dans une perspective légèrement différente. Le gradient de la fonction coût en ce qui concerne un filtre synaptique est développé comme suit :

$$\frac{\partial C}{\partial \mathbf{w}_{ij}^l} = \sum_k \frac{\partial C}{\partial s_j^{l+1}(k)} \cdot \frac{\partial s_j^{l+1}(k)}{\partial \mathbf{w}_{ij}^l}. \tag{8}$$

où $s_j^{l+1}(k) = \sum_i \mathbf{w}_{i,j}^l \cdot \mathbf{x}_i^i(k)$ est l'entrée du neurone j antérieurement à la sigmoïde. On peut interpréter $\partial C/\partial s_j^{l+1}$ comme une mutation dans l'erreur quadratique totale sur tous le temps, due à une modification dans l'entrée du neurone en un simple instant de temps. Noter qu'on n'exprime pas le gradient total d'une manière traditionnelle comme une somme des gradients instantanés. En utilisant cette nouvelle expansion, l'algorithme stochastique suivant est obtenu :

$$\mathbf{w}_{ij}^l(k+1) = \mathbf{w}_{ij}^l(k) - \eta \frac{\partial C}{\partial s_j^{l+1}(k)} \cdot \frac{\partial s_j^{l+1}(k)}{\partial \mathbf{w}_{ij}^l}. \tag{9}$$

Une dérivation complète de cet algorithme est donnée dans l'annexe A. L'algorithme final obtenu, appelé *temporal backpropagation* ou *rétropropagation temporelle* peut être résumé comme suit :

$$\begin{aligned} \mathbf{w}_{ij}^l(k+1) &= \mathbf{w}_{ij}^l(k) - \eta \delta_j^{l+1}(k) \cdot \mathbf{x}_i^l(k) \\ \delta_j^l(k) &= \begin{cases} -2e_j(k) f'(s_j^L(k)) & l = L \\ f'(s_j^l(k)) \cdot \sum_{m=1}^{N_{l+1}} \delta_m^{l+1}(k) \cdot \mathbf{w}_{jm}^l & 1 \le l \le L-1 \end{cases} \end{aligned} \tag{10}$$

où e_j^l est l'erreur en nœud d'entrée, $f'()$ est la dérivée de la fonction sigmoïde, et $\delta_m^l(k) = [\delta_m^l(k)\ \delta_m^l(k+1)\ \delta_m^l(k+2)\ .....\delta_m^l(k+N^{l-1})]$ est le vecteur des

[5] Les réseaux TDNN sont typiquement utilisés pour la classification dans l'adaptation mode batch. L'apprentissage consiste en une mémorisation totale de tous les états jusqu'à ce que la totalité des échantillons intéressants soient capturés, ensuite on utilise une back-propagation à travers de multiples versions du réseau de "décalage-temps". Cela peut être une équivalence à l'utilisation d'un réseau déployé comme ci-dessus.

termes propagés du gradient. On observe immédiatement que ces équations sont vues comme une généralisation vectorielle du familier algorithme de rétropropagation. En effet, en remplaçant les vecteurs x, w, et δ par des scalaires, les équations précédentes sont réduites exactement à l'algorithme de la rétropropagation standard pour des réseaux statiques. Les différences dans la version temporelle sont une affaire de relations mathématiques implicites et d'opérations de filtrage. Pour calculer $\delta_j^l(k)$ en un neurone donné, on *filtre* la forme de δ à partir de l'arrière de la prochaine couche à travers les synapses FIR que le neurone envisagé alimente (Figure 6). Ainsi, les δ ne sont pas formés en prenant simplement une somme pondérée, mais par un filtrage en arrière. Pour chaque nouvelle entrée et un vecteur réponse désiré, le filtrage en avant est incrémenté par une étape dans le temps, la même chose pour le filtrage en arrière. Les poids sont alors adaptés en ligne à chaque incrémentation temps[6].

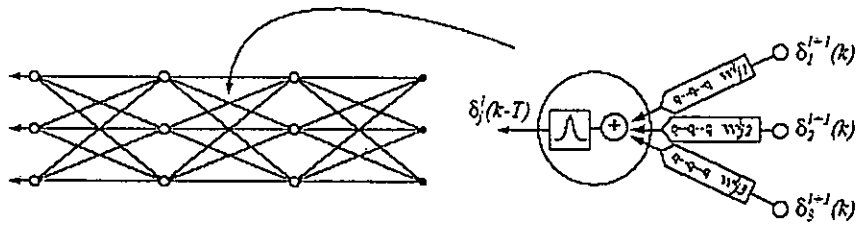


Figure 6: Dans la rétropropagation temporelle, les termes en delta sont filtrés à travers les connexions synaptiques pour former les deltas des couches précédentes. Le processus est appliqué à travers tout le réseau, couche par couche, en travaillant en arrière.

La rétropropagation temporelle préserve la symétrie entre la propagation en avant des états et la propagation en arrière des termes d'erreur. Le traitement distribué parallèle est maintenu. En outre, maintenant, le nombre d'opérations par itération augmente de façon *linéaire*, uniquement avec le nombre de couches et de synapses. Cette économie vient comme une conséquence de la formulation récursive efficace. Chaque unique coefficient entre dans le calcul seulement une fois par contraste avec l'utilisation redondante des termes quand on applique la rétropropagation standard pour un réseau développé[7].

[6]Les erreurs du gradient peuvent aussi être accumulées pour une adaptation en mode batch.

[7]En général,cette comparaison est aussi appliquée aux méthodes traditionnelles d'apprentissage des TDNNs. Toutefois, TDNNs sont typiquement utilisés pour les prob-

4 Equation-Erreur, adaptation et prédiction:

Nous sommes maintenant en position de discuter l'utilisation du réseau FIR dans le contexte de la prédiction des séries temporelles. Rappelons qu'on désire modéliser la série $y(k)$ (notre étude dans ce chapitre se limite à des séries scalaires, c'est-à-dire uni variantes). La figure 7a illustre la configuration de base de l'apprentissage pour une prédiction. Pour chaque pas de temps, l'entrée du réseau FIR est la valeur connue $y(k-1)$, et la sortie $\hat{y}(k) = \aleph_q[y(k-1)]$ est le pas d'estimation singulière de la vraie valeur de la série $y(k)$. Notre modèle construit est donc :

$$y(k) = \aleph_q[y(k-1)] + e(k). \tag{11}$$

Comme cela à été déjà expliqué, le réseau FIR $\aleph_q$ peut être représenté comme étant un réseau contraint agissant sur une fenêtre finie de l'entré (c'est-à-dire $\aleph_q[y(k-1)] \equiv \aleph_c[y(k-1), y(k-2), ..., y(k-N)]$). Cela nous permet de réécrire l'équation 11 comme suit :

$$y(k) = \aleph_c[y(k-1), y(k-2), ..., y(k-N)] + e(k), \tag{12}$$

ce qui accentue l'autorégression non linéaire.

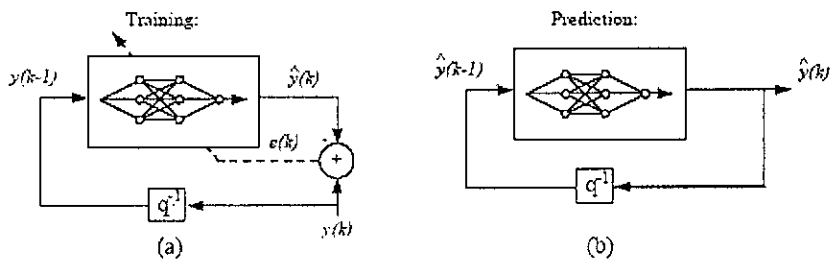


Figure 7: La configuration réseau pour la prédiction: Le pas singulier de prédiction $\hat{y}(k)$ est prise comme l'entrée du réseau conduit par les échantillons précédents de la séquence $y(k)$. Pendant l'apprentissage l'erreur quadratique $e^2(k) = (y(k) - \hat{y}(k))^2$, est minimisée en utilisant la backpropagation. Alimentant l'estimé $\hat{y}(k)$ en aval, il forme un processus de boucle fermée utilisé pour les prédictions itératives à long terme.

Durant l'apprentissage, l'erreur quadratique $e^2(k) = (y(k) - \hat{y}(k))^2$ est minimisée en utilisant l'algorithme de rétropropagation temporelle pour adapter

lèmes de classification des échantillons, dans lesquels seulement une erreur unique est donnée pour la totalité des échantillons. Dans quelques situations spéciales, un arrangement déplié peut être utilisé, pour des exigences de calcul, comme rétropropagation temporelle.

le réseau ($y(k)$ agit comme étant la réponse désirée). Noter qu'on est en train de faire une adaptation boucle ouverte; l'entrée et de la réponse désirée sont fournies depuis les séries d'apprentissage qui sont connues. La sortie actuelle du réseau n'est pas réinjectée comme entrée durant l'apprentissage. Un arrangement pareil est appelé adaptation de *l'équation-erreur*. La communauté des réseaux de neurones a ensuite adopté le terme : "*teacher-forcing*" ou *maitre-forçant*.

Un simple argument pour l'adaptation dans ce mode est le suivant : dans un environnement stochastique stationnaire, minimiser la somme des erreurs quadratiques $e^2(k)$ correspond à minimiser l'espérance de l'erreur quadratique :

$$E[e^2(k)] = E[y(k) - \hat{y}(k)]^2 = E[y(k) - \aleph_c[\mathbf{y}_1^N(k)]]^2. \tag{13}$$

où $\mathbf{y}_1^N(k) = [y(k), y(k-1), ..., y(k-N)]$ spécifie le régresseur[8]. L'espérance est pris en tenant compte de la distribution commune sur $y(k)$ à travers $y(k-N)$. Maintenant considérons l'identité :

$$E[y(k) - \aleph_c[\mathbf{y}_1^N(k)]]^2 = E[y(k) - E[y(k) \mid \mathbf{y}_1^N(k)]]^2 + E[\aleph_c - E[y(k) \mid \mathbf{y}_1^N(k)]]^2. \tag{14}$$

Le premier terme sur la droite de cette équation est indépendant de l'estimateur, d'où l'architecture réseau optimale $\aleph_c^*$ est immédiatement vue comme étant la suivante :

$$\aleph_c^* = E[y(k) \mid \mathbf{y}_1^N(k)], \tag{15}$$

c'est-à-dire la moyenne conditionnelle de $y(k)$ connaissant les valeurs de $y(k-1)$ jusqu'à $y(k-N)$, qu'on a voulu moyenner. Encore une fois, on doit souligner que ceci est la seule motivation pour l'utilisation de l'apprentissage d'un réseau prédicteur dans cette manière de voir. Nous ne pouvons pas conclure que l'adaptation réalisera nécessairement l'optimum pour une structure et une séquence d'apprentissage donnée. Pour plus d'informations au sujet des estimateurs biaisés dans le contexte de l'apprentissage d'un réseau voir [Geman S., Bienenstock E., et Doursat R. 1992].

Une fois le réseau entraîné, la prédiction long terme itérée est réalisée en prenant l'estimateur $\hat{y}(k)$ et en l'alimentant en retour comme entrée au réseau :

$$\hat{y}(k) = \aleph_q[\hat{y}(k-1)]. \tag{16}$$

Ce système en boucle fermée est illustré sur la figure 7b. L'équation 16 peut être itérée en avant, dans le temps, pour réaliser des prédictions

[8]Dans l'équation 13, $y(k)$ correspond à des variables aléatoires dans un procédé stationnaire stochastique, et non à des points spécifiques dans une série donnée.

désirées aussi loin que possible dans le futur. Noter que pour un système linéaire, les racines des coefficients de régression doivent être contrôlées pour assurer que le système en boucle fermée reste stable. Cependant, pour le réseau de neurones, la réponse de la boucle fermée aura toujours une sortie bornée stable, chose due à la sigmoïde qui limite la portée des dynamismes des sorties.

Comme l'apprentissage est uniquement basé sur un pas singulier de prédiction, l'exactitude des prédictions itérées à long terme ne peut pas être garantie d'avance. Une question très importante se pose; elle concerne la solidité de l'apprentissage de la boucle ouverte quand le système final est à exécuter en boucle fermée. En effet, l'adaptation de l'équation-erreur pour des autorégressions linéaires souffre des convergences vers une solution boucle fermée biaisée (c'est-à-dire, $\theta = \theta^* + biais$, où θ^* correspond à l'ensemble optimal des paramètres de la boucle fermée). Il peut sembler plus prudent d'utiliser une configuration qui adapte directement le système boucle fermée. Une telle configuration est référée comme étant l'adaptation *sortie-erreur* ou « *output-error adaptation* ». Dans le cas linéaire, la méthode aboutit à un estimateur qui n'est pas biaisé. Paradoxalement, le prédicteur linéaire peut converger vers un minimum local. Toutefois, les algorithmes d'adaptation eux-mêmes deviennent plus compliqués et moins sûrs, en raison du feedback. En conséquence, nous n'allons pas considérer dans notre étude l'approche sortie-erreur avec réseaux de neurones.

4.1 Résultats des données de la SFI :

La figure 8 illustre les données du laser (1100 points). On suppose qu'on dispose uniquement de 1000 points, le but étant de prédire les futurs 100 points suivants.

L'utilisation du réseau FIR permet de donner les continuations sur un ensemble de 100 points. Dans le but de comparer, la figure 9 illustre, en plus des points réels dont nous disposons (plus de 10 000 échantillons), les continuations trouvées. Nous remarquons que les 25 derniers points prédits ont encore besoin d'un ajustement (c'est ce qu'on appelle le comportement à long terme). Il est important de souligner le fait que la prédiction a été faite en utilisant uniquement les 1000 premières valeurs. Les valeurs réelles de la série après 1000 points ne sont pas fournies au réseau malgré leur disponibilité lors de l'apprentissage. Comme on peut le voir, la prédiction est remarquablement précise avec seulement une légère dégradation de phase. Une prédiction basée sur un modèle linéaire, une autorégression d'ordre 25, est aussi illustrée pour accentuer la différence avec un modèle traditionnel linéaire.

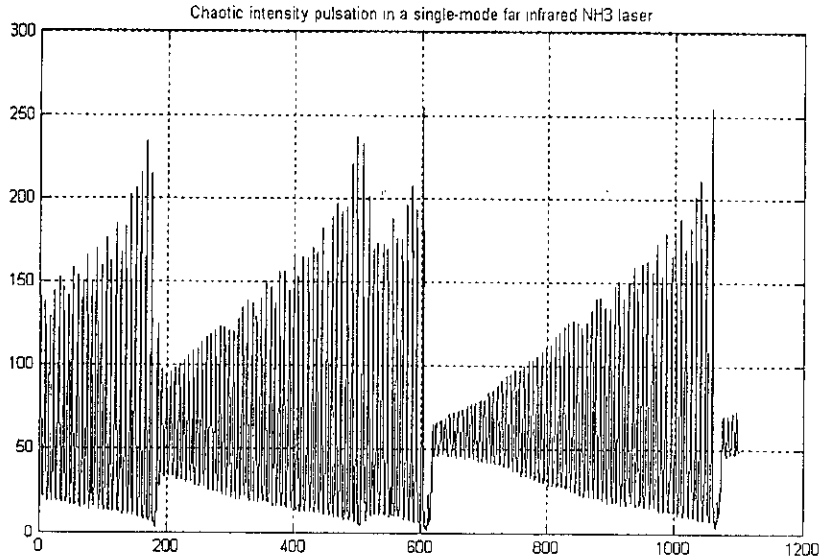


Figure 8: 1100 points des données du laser.

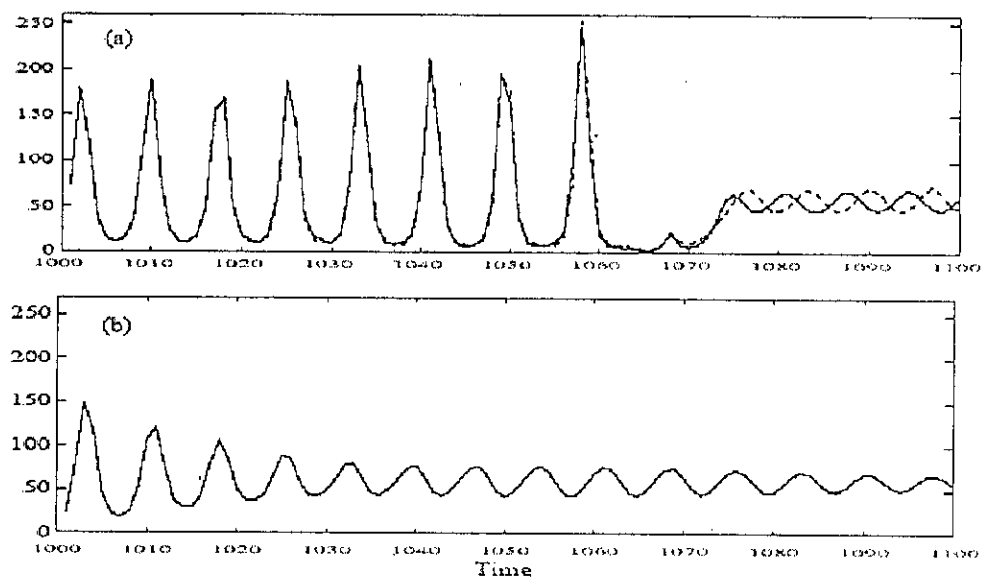


Figure 9: Prédiction de la série temporelle: (a) prédiction itérée par réseau de neurones (les 75 premiers points). La prédiction est basée uniquement sur un apprentissage de 1000 points. Les lignes discontinues correspondent aux continuations actuelles de la série. (b) 100 points représentant les itérations prédites par une autorégression d'ordre 25. Les coefficients de la régression ont été trouvés par la méthode du moindre carré.

Pour ce genre de données, d'autres modèles ont été appliqués. On cite l'arbre k-d, piecewise interpolation linéaire, le plongement passe-bas, SVD, le plus proche voisin, et les filtres de Wiener. Il a été démontré que pour les mé-thodes citées, la prédiction avec un réseau FIR, présente de meilleurs résultats. La comparaison de ces méthodes a été faite dans le cadre de la *SFI competition*. Alors que l'application de ce modèle aux données du laser n'est qu'un simple exemple, ces premiers résultats sont très encourageants. Dans ce qui suit nous allons apporter quelques détails concernant la mesure de la performance, la sélection des paramètres du réseau, l'apprentissage, le test, et l'analyse des résultats.

La performance d'une mesure : Une mesure de la qualité de la prédiction est donnée par «the normalized sum squared error » ou la somme normalisée des erreurs quadratiques :

$$nSSE = \frac{1}{\sigma^2 N} \sum_{k=1}^{N} (y(k) - \hat{y}(k))^2. \tag{17}$$

où $y(k)$ est la vraie valeur de la séquence, $\hat{y}(k)$ est la prédiction, et σ^2 est la variance de la séquence réelle sur la durée N de la prédiction. Une valeur de $nSSE = 1$ correspond donc à une prédiction de la moyenne. Le tableau 2 résume différentes valeurs pour deux cas : pour une prédiction à pas singulier et pour une prédiction itérée.

Durée		Prédiction à pas singulier	Prédiction itérée
100	1000	0.00044	-
900	1000	0.00070	0.0026
1001	1050	0.00061	0.0061
1001	1100	0.0230	0.0551
1001	1100	-	0.0273

Tableau 2: Les mesures de la somme des erreurs quadratiques normalisées.

Sélection de la dimension du réseau : le réseau FIR utilisé est un réseau à trois couches avec $1 \times 12 \times 12 \times 1$ nœuds et $25 : 5 : 5$ unités de retard. La sélection de la dimension 3 est basée le plus souvent sur essais et sur l'erreur le long des heuristiques. Parce que la première couche du réseau agit comme une banque de filtres linéaires, la sélection de l'ordre des filtres est motivée par des techniques linéaires. Comme le montre la figure 10, l'erreur résiduelle à pas singulier en utilisant un prédicteur linéaire AR fait apparaître une amélioration négligeable pour un ordre supérieur à 15; quant aux autocorrélations, elles indiquent une corrélation substantielle pour un retard de 60 approximativement. Les réseaux candidats évalués incluent des

filtres d'ordre 10, 50, et 100 dans la première couche, avec un nombre variant d'unités dans les couches cachées. Une tentative d'analyse plus performante sur les filtres pris individuellement pour un réseau convergeant n'est pas plus éclairante que les résultats précédents. En général, la sélection des dimensions des réseaux de neurones reste un problème difficile qui demande d'autres recherches.

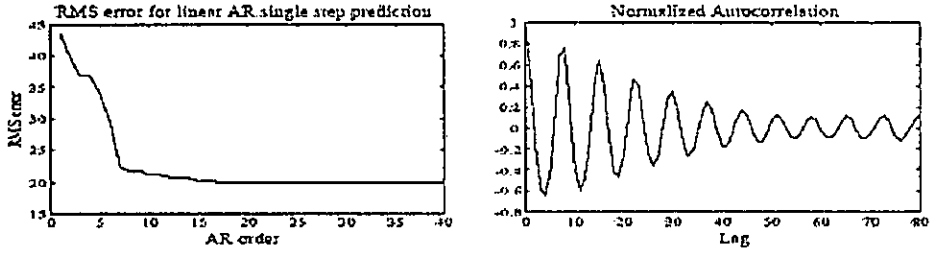


Figure 10: (a) Erreurs de prédiction à pas singulier en utilisant un filtre linéaire AR(N). (b) Autocorrélation des données du laser.

Apprentissage et validation croisée : pour un but d'apprentissage, les donnée on été réduites à une moyenne nulle et une variance égale à l'unité. Les valeurs initiales des poids sont choisies aléatoirement puis forcées à prendre une même variance à chaque neurone de sortie. Le taux d'apprentissage est pris symboliquement égal à 1% (cela a été sélectionné heuristiquement puis varié au cours de l'apprentissage). Pour notre étude, l'apprentissage a été fait sur 900 points de la série et les 100 points restants sont utilisés pour la validation croisée. Les deux cas - la prédiction de l'erreur par pas singulier et la prédiction itérée en boucle fermée - sont contrôlés pour 100 points. On n'observe pas de sur-ajustement pour l'erreur à pas singulier, mais un réseau plus large a une meilleur réponse. Toutefois, un réseau plus large avec une petite erreur à pas singulier a souvent une mauvaise prédiction itérée. Comme la tâche réelle est d'impliquer la prédiction à long terme, la mesure de la performance itérée a été finalement celle utilisée pour évaluer les réseaux candidats. Il est clair, d'après la nature des données, que la prédiction de l'intensité d'affaissement de haut en bas peut être la plus importante et l'aspect le plus difficile à apprendre. Comme les 100 points – utilisés pour la validation croisée- ne contiennent pas de pareilles caractéristiques, la prédiction itérée commence auprès du point 550 où l'exécution détermine comment le réseau prévoit l'affaissement connue au point 600.

La même structure du réseau a été entraînée avec différents poids au co-mmencement pour accéder à la sensibilité des conditions initiales. La

majorité des cas mènent vers le même résultat, et si ce n'est pas le même, la différence est infiniment petite.

Filtrage paramétrique et comportement à long terme : Le réseau utilisé pour apprendre la série a un total de 1105 paramètres variables et à été seulement entraîné avec 1000 points. (Un statisticien serait très surpris). On spécule qu'il est nécessaire de modéliser exactement l'occurrence d'un seul affaissement du signal à partir de deux exemples seulement de l'ensemble de l'apprentissage. Dans un sens, nous sommes forcés de faire avec l'apprentissage qui apparaît statistiquement être un autre. Il faut aussi réaliser que pour ce système non linéaire les degrés de liberté ne correspondent pas au nombre des paramètres libres. Nous sommes réellement en train d'essayer d'adapter une fonction contrainte par la topologie du réseau. Pour plus de détails sur les degrés de liberté effectifs, voir [Moody J. 1992].

Une conséquence du nombre important de paramètres peut être observée dans la prédiction itérée étendue sur un long terme (Figure 11). Le signal devient endommagé et manifeste un comportement bruité, chose due au nombre excessif des degrés de liberté. Tandis qu'un réseau plus petit (ou l'ajout d'une régularisation des poids) empêche ce phénomène, et peut prédire exactement le lieu de l'affaissement de l'intensité. Noter que la prédiction a des sorties bornées stables, en raison de la limitation des sigmoïdes à l'intérieur du réseau.

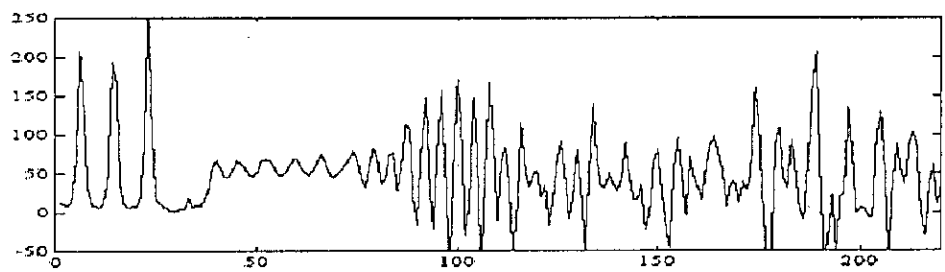


Figure 11: Prédiction itérée prolongée.

A cause d'une éventuelle corruption du signal, seulement les 75 premiers points des prédictions itérées sont pris en charge. Cela correspond à quelques pas de temps après la détection de l'affaissement de l'intensité et a été choisi sur la base d'une inspection visuelle du lieu où la prédiction itérée se détériore. Puisque après un affaissement d'intensité, la vraie série tend à montrer une oscillation simple de croissance lente, les 25 points restants ont été choisis adjoints d'une séquence semblable prise à partir de l'ensemble d'apprentissage.

Prédiction des erreurs : Pour les données du laser, une estimation de l'incertitude des prédictions a aussi été donnée. Il est supposé que la séquence

réelle observée, $y(k)$, est dérivée d'un processus de Gauss indépendant avec une moyenne correspondante à la prédiction $\hat{y}(k)$, et une variance $\hat{\sigma}(k)^2$. Donc, la déviation standard $\hat{\sigma}(k)$ détermine les barres d'erreur pour chaque valeur prédite. Une mesure de la probabilité que la séquence observée a été générée par un modèle Gaussien est donnée par « the negative average log-Likelihood » ou la *log-probabilité moyenne négative*

$$nalL = \frac{1}{N}\sum_{k=1}^{N} \log\left(\frac{1}{\sqrt{2\pi\hat{\sigma}(k)^2}}\int_{y(k)-0.5}^{y(k)+0.5} \exp\frac{(\tau-\hat{y}(k))^2}{2\hat{\sigma}(k)^2}d\tau\right) \qquad (18)$$

Dans le but d'estimer $\hat{\sigma}(k)^2$, nous faisons la moyenne des erreurs de prédiction quadratiques itérées connues en commençant du temps 400 jusqu'à 550 (c'est-à-dire 150 prédiction itérées séparées sont utilisées). Il est clair qu'il est souhaitable de baser ces estimations sur des segments de données en dehors de l'ensemble d'apprentissage actuel ; toutefois, en raison de la limitation des données fournies, cela n'est pas possible. Les barres d'erreurs trouvées en utilisant l'approche citée en plus des prédictions sont illustrées dans la figure 12 et correspondent à un $nalL = 1.51$ (La courbe discontinue correspond à la série actuelle). D'autres méthodes bayesiennes pour l'estimation des incertitudes ont été proposées par [MacKay 1992] et [Skilling 1990].

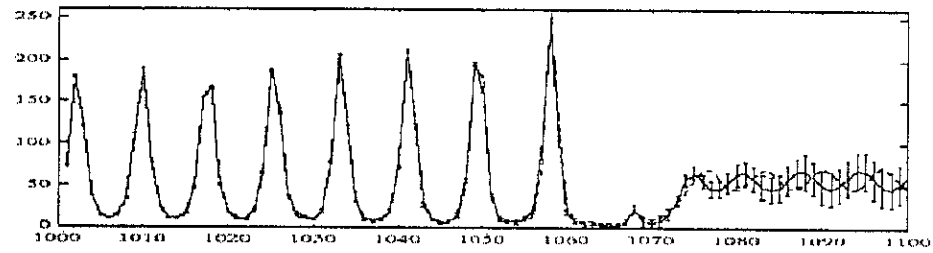


Figure 12: Estimation des déviations standard des barres d'erreurs; l'estimation est centrée en chaque point prédit.

Prévisions additionnelles : Les continuations complètes de la série sont fournies (plus de 10000 points). La figure 13 montre les différentes prédictions itérées en commençant par différentes localisations à l'intérieur de la série. Le réseau d'origine (qui a subi un apprentissage sur 1000 points uniquement) est utilisé. Tandis qu'il y a souvent dégradation apparente d'exécution, le réseau est toujours étonnamment précis à prévoir l'occurrence des affaissements d'intensité.

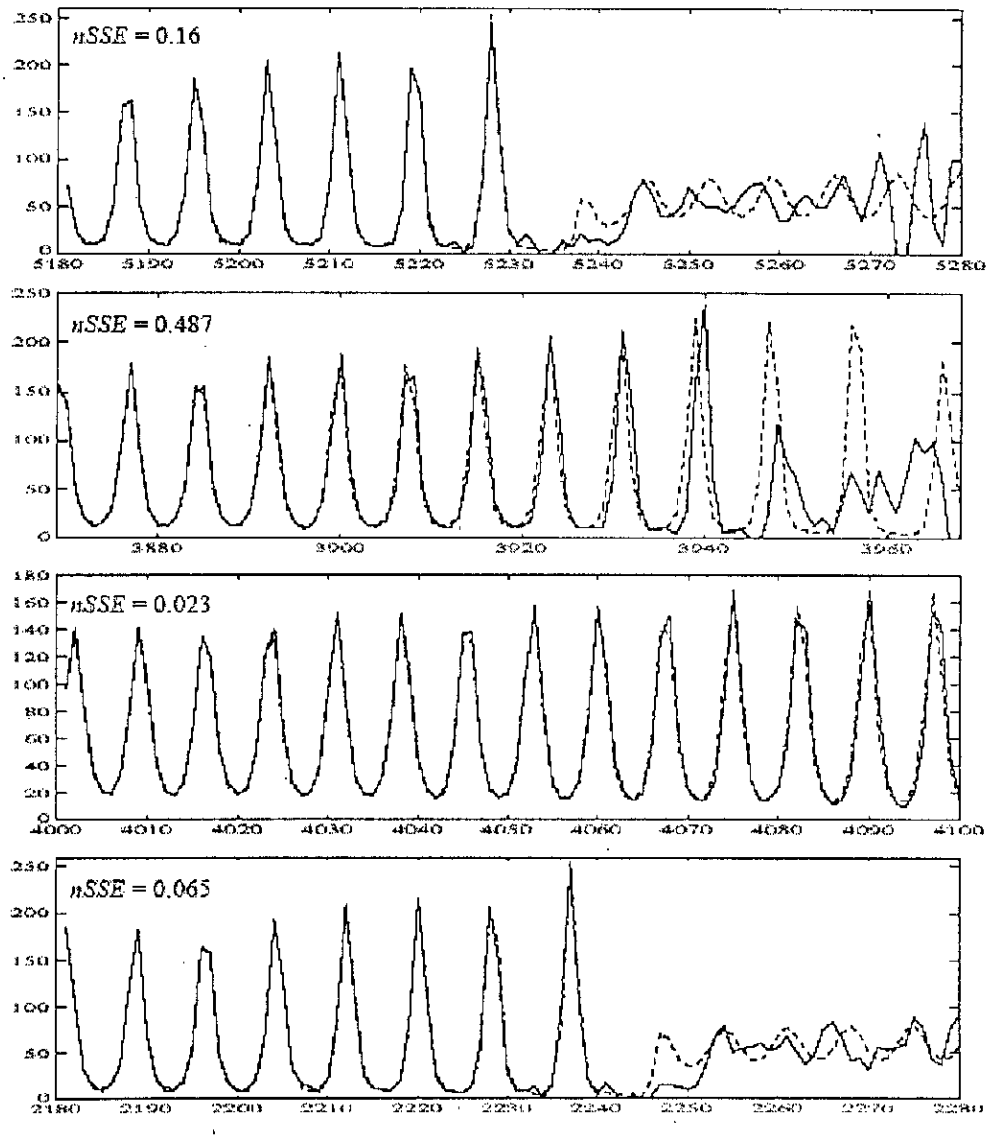


Figure 13: Le réseau d'origine, entraîné sur 1000 points, est utilisé pour produire des prédictions itérées en plusieurs lieux des données. Les courbes discontinues correspondent toujours à la série réelle.

5 Les autres études:

Nous venons de faire étude détaillée du réseau FIR et nous avons appliqué ce modèle aux données du Laser à des fins de prédiction, pour tenter de

trouver un modèle « acceptable ». Ce n'est bien sûr pas le meilleur, et il existe d'autres configurations qui pourront donner des résultats plus précis. Un meilleur modèle serait celui qui a besoin de moins de paramètres et qui donne le plus proche - de la réalité - description du système. On est donc confronté au problème classique de compromis entre simplicité et efficacité : on cherche, d'une part, le modèle le plus simple (moins de calculs et de paramètres), et d'autre part, celui qui donne des résultats exacts. Ce compromis est l'objet des études de la Théorie d'Information Algorithmique « Algorithmic Information Theory », développée par Chaitin 1966, Kolmogorov 1965, et Solomonoff 1964. Dans cette théorie, l'information est mesurée - à travers une chaîne unique de symboles - par le nombre de bits dont on a besoin pour spécifier le plus court algorithme pouvant les générer. Mais il n'existe pas un algorithme universel pour trouver le plus petit programme pouvant générer la séquence observée, car chaque problème – c'est-à-dire à chaque série temporelle – est rendu indépendant des autres par des caractéristiques spécifiques, et donc, même si la démarche de raisonnement est la même, les algorithmes et les programmes diffèrent. Aussi existe-il une profonde limitation sur le plan théorique, dans la modélisation des séries? Les contraintes associées à chaque domaine d'application, ne peuvent pas toujours permettre de trouver les résultats désirés.

Dans notre étude, nous avons ignoré plusieurs problèmes posés par les séries temporelles; cela ne signifie pas que ces problèmes ne sont pas l'objet de recherches qui tentent d'en venir à bout :

1. Construction des modèles paramétrés.
2. Contrôle des systèmes non linéaires.
3. Analyse des systèmes ayant une structure spatiale similaire à la structure temporelle.
4. Apprentissage "Temporal Difference" (TD learning).
5. Nettoyage des données.
6. Experts for prédiction (Gausian Mixture Moel GMM).
7. Etude des séries multivariables non stationnaires.
8. Les réseaux récurrents.

6 Conclusion:

Dans ce chapitre nous nous sommes focalisé sur le modèle de base autorégressif. Toutefois, il est clair que la méthodologie générale présentée ici peut être facilement étendue à d'autres configurations. L'extension la plus évidente est la méthode AutoRegressive Moving Average non linéaire :

$$y(k) = \aleph[y(k-1), y(k-2), ..., y(k-N), e(k-1), e(k-2), ..., e(k-N2)] + e(k). \tag{19}$$

où $\aleph$ peut être un réseau FIR, un réseau Feed-Forward standard, ou tout autre topologie de réseau. Le modèle ARMA est le plus appliqué pour la prédiction à pas singulier, où on peut ajouter une régression sur les valeurs passées des prédictions d'erreurs résiduelles qu'on connaît. Cependant, les deux modèles AR et ARMA ont été extrapolés des modèles traditionnels des équations différentielles linéaires.

Cet exposé sur les réseaux FIR nous a démontré l'efficacité des réseaux de neurones, nous allons d'aborder maintenant les question concrètes de la modélisation des données relatives au sujet de ce mémoire.

Chapitre 4: Modélisation de la durée d'insolation de l'Algérie

Nous allons procéder à une l'analyse des données de la durée d'insolation sur le territoire Algérien. Le but est de modéliser l'ensemble des données disponibles, issues des mesures des stations de la NASA. L'ensemble des données utilisées est disponible sur l'adresse Internet *http://eosweb.larc.nasa.gov /sse/*. Il s'agit donc de trouver les paramètres du modèle. La méthode utilisée est l'Analyse Non Linéaire de la Composante Principale, mieux connue en anglais par: NonLinear Principal Component Analysis NLPCA.

Dans les chapitres précédents, nous avons démontré la défaillance des modèles linéaires, et nous avons utilisé la méthode des réseaux de neurones pour une série de données (données du laser) à une fin de prédiction. Dans cette partie, la méthode réseaux de neurones est aussi utilisée, mais la différence avec ce qui a été fait précédemment est qu'ici, cette méthode, sera utilisée uniquement pour modéliser les données. L'autre aspect de l'analyse qui apparaît dans cette partie est que la série sur laquelle nous avons travaillé est multivariables; la durée d'insolation dépend de la latitude et de la longitude. Quoique la dépendance en longitude soit moins forte, mais ne pouvant ignorer cette dernière, nous avons préféré étudier les données en tant que multivariables.

1 Durée d'insolation et fraction d'insolation:

1.1 Définitions:

On parle d'**insolation**, dès que le rayonnement solaire est suffisant pour créer une ombre portée bien nette. Ce phénomène est observé à l'aide d'un héliographe, appareil conçu pour mesurer la durée de cette insolation. La **durée maximale théorique d'insolation** par ciel clair est le temps compris entre les heures de lever et de coucher du soleil. Pour un lieu donné cette durée maximale théorique évolue naturellement au cours de l'année, avec le cycle des saisons. Pour un jour donné, le rapport entre la durée d'insolation observée et la durée maximale théorique ainsi définie, est appelé **fraction d'insolation**. Ce nombre dépend de la seule couverture nuageuse de la journée ainsi que de l'ombrage (obstacles: montagnes,...), mais nos données ne prennent pas en considération l'ombrage, car pour effectuer ce genre d'étude, il faut s'intéresser au lieu précis.

1.2 Mesure:

Pour déterminer expérimentalement la fraction d'insolation, on mesure, pour une journée, le temps durant lequel a brillé le soleil. Cette durée, SS, est la " *durée d'insolation quotidienne* ". On appelle SS_0 la durée astronomique du jour (intervalle entre le lever et le coucher du Soleil). La " fraction d'insolation quotidienne ", notée σ , est par définition :

$$\sigma = \frac{SS}{SS_0} \tag{1}$$

L'intérêt de cette notion est surtout que sa détermination est facile. Il existe un appareil rudimentaire, l'héliographe, pour mesurer SS. Par ailleurs, la durée du jour SS_0 est calculable à partir des lois cosmographiques.

1.2.1 L'héliographe de Campbell-Stokes:

Selon la définition de l'encyclopédie, l'héliographe est " un instrument enregistrant la durée d'ensoleillement". Il a été inventé en 1853 par le Britannique J.F. Campbell, puis amélioré par Sir George Gabriel Stokes en 1879. Il est constitué d'une sphère de verre qui concentre le rayonnement solaire direct sur une bande de carton placée sur un support à l'arrière de la sphère. Ce carton, dont la nature et la structure sont normalisées, est décoloré ou brûlé par le rayonnement direct. Chaque jour, le carton est remplacé. La longueur de la brûlure, mesurée avec une règle, permet d'estimer en heures et en dixièmes d'heure la durée réelle d'insolation. Cet appareil, dont l'utilisation est simple, porte le nom de ses inventeurs (héliographe de Campbell-Stokes) ; il est encore employé, mais on a tendance à le remplacer par des instruments automatiques.

L'héliographe automatique: Il permet l'enregistrement de la durée d'insolation dans les stations météorologiques automatiques. Il existe deux de ces types d'héliographes, tous deux fonctionnant avec des cellules photovoltaïques : les modèles statiques et les modèles dynamiques.

L'héliographe à cellules photovoltaïques: Il s'agit d'un instrument statique mesurant la durée pendant laquelle l'éclairement solaire est supérieur à $120 W/m^2$, une valeur conseillée par l'OMM (l'Organisation Météorologique Mondiale). Il comporte deux cellules photovoltaïques dont l'une reçoit le rayonnement solaire global et l'autre uniquement le rayonnement diffus grâce à un écran. L'instrument délivre une tension électrique sensiblement proportionnelle à l'intensité du rayonnement direct. Lorsque le soleil est présent, les deux cellules délivrent un signal déséquilibré à partir d'un seuil donné dont la tension électrique est sensiblement proportionnelle à l'intensité du rayonnement direct.

L'héliographe dynamique: Il comporte un élément tournant qui permet d'analyser le contraste existant entre la luminance du ciel et celle du soleil.

L'héliographe à fibre optique: Il s'agit d'un variante du précédent équipé d'une fibre optique en rotation qui intercepte le rayonnement direct du soleil quelle que soit sa position.

1.2.2 Description de l'héliographe de Campbell-Stokes:

L'appareil est constitué d'une loupe sphérique en verre, fixée par ses deux pôles à un pied. A l'arrière de la sphère, on place un carton thermosensible sur un support métallique incurvé et concentrique à la lentille de verre. L'axe de la sphère est incliné exactement comme celui de la terre par rapport au soleil.

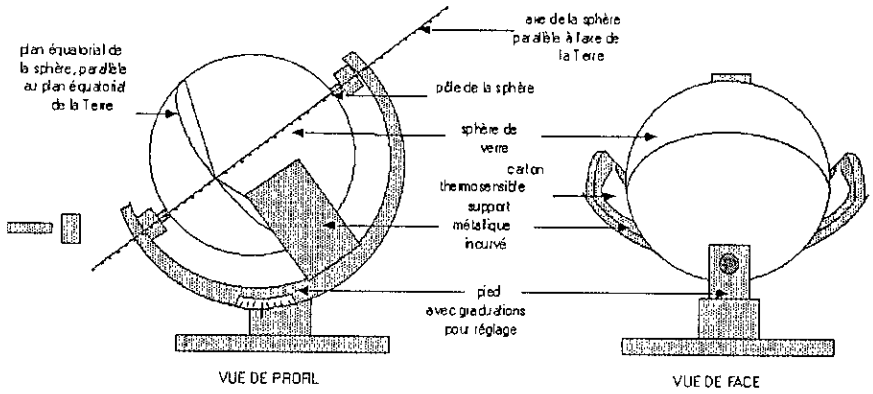


Figure 1: L'héliographe de Campbell-Stokes.

La sphère de verre, constituant une lentille convergente, concentre les rayons du soleil sur le carton gradué en heures placé sur le support. La hauteur du soleil et la durée du jour varient au cours de l'année, selon les saisons ; les cartons utilisés seront dès lors différents selon les saisons et placés différemment sur le support.

Exploitation : le fonctionnement de l'héliographe repose sur:

- la graduation des cartons (inversion de la graduation par rapport à la course du soleil : tracé du rayon passant par le centre de la sphère)
- la longueur des cartons utilisés en été est plus grande : relation avec la durée du jour
- la forme et la position des cartons sur le support : relation avec les saisons.

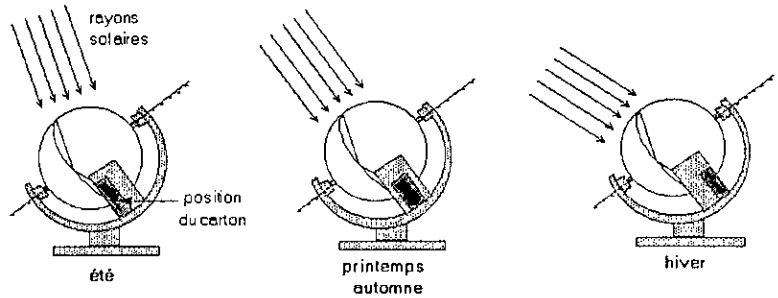


Figure 2: Positions de l'appareil selon les saisons.

2 Position du problème:

Notre tâche consiste donc à modéliser les données de la fraction d'insolation de l'Algérie, dont le territoire est compris entre 18° et 38° de latitude Nord, et entre 9° de longitude Ouest et 12° de longitude Est, (le méridien intervalle 0° Greenwich passant près de Mostaganem).

Comme nous l'avons expliqué à la fin du chapitre 2, les modèles contiennent beaucoup plus de paramètres, ce qui nous permet d'opter pour le choix d'une théorie pauvre et des données riches.

Dans notre cas, on dispose d'un fichier de données assez pauvre, et afin d'avoir le modèle le plus adapté à l'Algérie, nous avons fait le choix d'un ensemble de latitudes allant de 18° à 43° avec un pas de un degré, et nous avons supposé que le modèle trouvé est valable pour l'ensemble du pays. 80% de ces latitudes sont utilisées pour l'apprentissage, et seulement 20% pour le test et la validation du modèle. Quant au choix des longitudes, il sera discuté dans la section prochaine.

Les données utilisées dans notre étude sont celles de la NASA, qui sont disponibles sur le Web. Pour chaque latitude et longitude, on a la fraction d'insolation correspondante.

A première vue, et en comparaison à la dépendance en latitude, σ ne dépend que légèrement de la longitude, mais après illustration en 3D de σ en fonction de la latitude et de la longitude, cet aspect ne peut être ignoré, vu son apparition comme composante déterminante de la fraction d'insolation ; l'annexe B donne les illustrations en trois dimensions des données utilisées pour les 12 mois de l'année. Toutefois nous avons jugé utile, dans le but de réduire la dimension du problème, de ne pas prendre toutes les longitudes entre 9° Ouest et 12° Est. Les valeurs utilisées sont les suivantes : 8° , 5° et 2° Est, 0°, et 2°, 4°, 7°, et 11° Est. Le choix de ces valeurs est motivé essentiellement par le fait que, dans les intervalles choisis, la fraction d'insolation

est presque constante. Le programme – écrit en MATLAB, et auquel on consacrera une section – est capable de calculer les paramètres du modèle pour n'importe quelles données prises sur n'importe quel lieu géographique (c'est-à-dire le programme est indépendant des données).

Les données dont nous disposons sont des moyennes mensuelles; elles aussi prises comme une moyenne sur 10 ans de mesures (de 1983 à 1993). On dispose donc de ce qu'on appelle la fraction d'insolation mensuelle. Vu cette disposition, nous avons jugé utile de déterminer pour chaque mois séparément un modèle adéquat. Finalement, nous disposons de 12 séries de données multivariables, la donnée principale étant la fraction d'insolation, et les variables la latitude et la longitude. Après une organisation, le tableau des données pour chaque mois se présente ainsi :

		colonne 1	colonne 2	...	colonne 24	colonne 25
	$Lon \backslash Lat$	18° Nord	19° Nord	...	42° Nord	43° Nord
ligne 1	8° Est	$\sigma_{1,1}$	$\sigma_{1,2}$	...	$\sigma_{1,24}$	$\sigma_{1,25}$
ligne 2	5° Est	$\sigma_{2,1}$	$\sigma_{2,2}$	...	$\sigma_{2,24}$	$\sigma_{2,25}$
ligne 3	2° Est	$\sigma_{3,1}$	$\sigma_{3,2}$	...	$\sigma_{3,24}$	$\sigma_{3,25}$
ligne 4	0°	$\sigma_{4,1}$	$\sigma_{4,2}$	...	$\sigma_{4,24}$	$\sigma_{4,25}$
ligne 5	2° Ouest	$\sigma_{5,1}$	$\sigma_{5,2}$	...	$\sigma_{5,24}$	$\sigma_{5,25}$
ligne 6	4° Ouest	$\sigma_{6,1}$	$\sigma_{6,2}$	...	$\sigma_{6,24}$	$\sigma_{6,25}$
ligne 7	7° Ouest	$\sigma_{7,1}$	$\sigma_{7,2}$	...	$\sigma_{7,24}$	$\sigma_{7,25}$
ligne 8	11° Ouest	$\sigma_{8,1}$	$\sigma_{8,2}$	...	$\sigma_{8,24}$	$\sigma_{8,25}$

Tableau 1: Présentation des données pour chaque mois.

Nous avons donc pour chaque mois une série multivariable de 8x25 échantillons. Sous forme matricielle, les données de chaque mois se présentent comme suit:

$$\Sigma_i = \begin{pmatrix} \sigma_{1,1} & \sigma_{1,2} & \cdots & \sigma_{1,24} & \sigma_{1,25} \\ \sigma_{2,1} & \sigma_{2,2} & \cdots & \sigma_{2,24} & \sigma_{2,25} \\ \sigma_{3,1} & \sigma_{3,2} & \cdots & \sigma_{3,24} & \sigma_{3,25} \\ \sigma_{4,1} & \sigma_{4,2} & \cdots & \sigma_{4,24} & \sigma_{4,25} \\ \sigma_{5,1} & \sigma_{5,2} & \cdots & \sigma_{5,24} & \sigma_{5,25} \\ \sigma_{6,1} & \sigma_{6,2} & \cdots & \sigma_{6,24} & \sigma_{6,25} \\ \sigma_{7,1} & \sigma_{7,2} & \cdots & \sigma_{7,24} & \sigma_{7,25} \\ \sigma_{8,1} & \sigma_{8,2} & \cdots & \sigma_{8,24} & \sigma_{8,25} \end{pmatrix} ; i = 1, ..., 12.$$

3 Théorie de la méthode utilisée:

L'analyse non linéaire de la composante principale "Nonlinear principal component analysis" (NLPCA) peut être exécutée par un modèle de réseau neurologique qui étend la non linéarité à l'analyse classique par la mé-thode

d'analyse de la composante principale "principal component analysis " (PCA). La présence des minimums locaux dans la fonction coût rend le NLPCA légèrement instable, puisque l'optimisation commence à partir de paramètres initiaux différents; la méthode converge souvent à différents minimums. La régularisation en ajoutant des limites de pénalité de poids à la fonction coût pour but d'améliorer la stabilité du NLPCA. Avec l'approche linéaire, il y a une dichotomie entre le PCA et le PCA tournée, car il est généralement impossible d'avoir une solution pouvant simultanément

- ▶ expliquer le désaccord global maximum des données,
- ▶ approcher les faisceaux locaux des données.

Avec le NLPCA, ces objectifs peuvent être atteints ensemble et ainsi la non linéarité dans le NLPCA unifie les approches PCA et PCA tournée. Avec le noeud circulaire du réseau à goulot d'étranglement, le NLPCA peut extraire les modes périodiques, ou les modes à ondes.

3.1 Introduction à l'analyse de la composante principale:

Dans l'analyse statistique classique des séries multivariables, la régression linéaire était la plus utilisée. Ensuite, de nouvelles méthodes ont été développées, essayant toujours d'affecter un comportement linéaire aux séries observées, comme PCA "Principale Component Analysis", et CCA "Canonical Correlation Analysis". Dans ce genre d'analyse, on essaye de trouver comment la réponse de la variable y est linéairement affectée par les variables de prédiction $\mathrm{x} \equiv [x_1, ..., x_l]$, i. e.

$$y = \mathrm{r} \cdot \mathrm{x} + r_o + \varepsilon \tag{2}$$

où ε est l'erreur (ou le résidu), et les coefficients de régression r et r_o sont trouvés en minimisant la moyenne de ε^2.

3.2 Analyse de la composante principale:

Pour analyser une grande variété de données, des images satellites en sortie des modèles numériques, météorologistes et océanographes ont adopté des méthodes statistiques multivariables classiques, telles que l'analyse de la composante principale (PCA) et l'analyse canonique de la corrélation (CCA). PCA (également connue comme analyse orthogonale empirique des fonctions) extrait les modes dans un ensemble de variables $\{x_l\}$. Elle est généralement employée pour deux buts:

★ réduire la dimensionnalité de l'ensemble des données en maintenant seulement les quelques premiers modes,

★ extraire des comportements (ou identifier des modèles) à partir de $\{x_i\}$, une tâche où elle est concurrencée par des méthodes dites PCA tournée "rotated PCA " (RPCA).

Dans plusieurs applications météorologiques et océanographiques, les données peuvent être exprimées sous la forme $\mathbf{x}(t) = [x_1, x_2, ..., x_l]$, où chaque variable x_i,$(i = 1, ..., l)$, est une série temporelle contenant n observations marquées par l'index t. Généralement t est tout simplement le temps, mais il peut être toute autre variable : température, pression, latitude,... etc. PCA est une méthode qui cherche u, une combinaison linéaire des x_i, et un vecteur associé $\mathbf{a}$, tel que :

$$u(t) = \mathbf{a} \cdot \mathbf{x}(t), \tag{3}$$

de façon que

$$\left\langle \|\mathbf{x}(t) - \mathbf{a}u(t)\|^2 \right\rangle \text{ soit minimisé} \tag{4}$$

où $\langle ... \rangle$ dénote une moyenne (en temps ou en échantillon). Ici u, appelé la composante principale (principal component PC), est une série temporelle, tandis que $\mathbf{a}$, le premier vecteur propre « eigenvector » de la matrice de covariance des données, appelé aussi fonction orthogonale empirique (empirical orthogonal function, EOF), décrit souvent un modèle spatial. Du résidu, $\mathbf{x} - \mathbf{a}u(t)$, et de la même manière; le deuxième mode PCA peut être extrait, et ainsi de suite pour les modes d'ordres supérieurs. En pratique, un algorithme commun peut extraire tous les modes simultanément.

La différence fondamentale entre PCA et NLPCA est que NLPCA accorde une architecture non linéaire pour passer de $\mathbf{x}$ à u, tandis que PCA accorde une architecture linéaire. En faisant un réapprentissage, uniquement sur les modes dominants, PCA a été largement utilisé pour réduire la dimension des données, et pour en extraire les modèles essentiels. PCA a été aussi étendu aux techniques de la SSA (Single Spectrum Analysis).

3.3 Le modèle réseau de neurones Feed-Forward:

Le modèle de réseau de neurones le plus utilisé dans ce genre d'application, est le modèle Feed-Forward, appelé aussi Perceptron Multicouches, qui accomplit une régression et une classification non linéaires. L'architecture de base (Figure 3), consiste en une couche de neurones d'entrés x_i, (un neurone est simplement une variable) reliée à une autre couche de neurones cachés, qui à leur tour, sont reliés à une couche de neurones de sortie y_j.

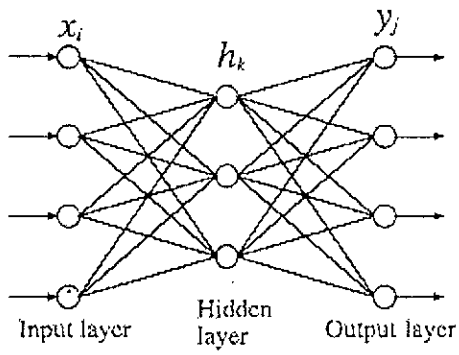


Figure 3: Un diagramme schématique du modèle réseau de neurones Feed-Forward, avec une seule couche cachée de neurones (c'est-à-dire variable), sandwiché entre la couche d'entrée et celle de sortie. Dans ce modèle, l'information vient en avant, en commençant par les neurones d'entrée. Augmenter le nombre de neurones cachés, augmente le nombre de paramètres.

Dans la figure 3, il y a uniquement une couche cachée h_k. Une fonction de transfert (fonction d'activation dans le langage des réseaux de neurones) trace des entrées aux neurones cachés. Il existe plusieurs fonctions de transfert, la plus utilisée est la tangente hyperbolique c'est-à-dire

$$h_k = \tanh(\sum_i \omega_{ki} x_i + b_k) \tag{5}$$

où ω_{ki} et b_k sont , respectivement, les paramètres poids et biais. La fonction $tanh(z)$ est une fonction profilée sigmoïde, où ses deux valeurs asymptotiques ± 1 quand $z \to \pm\infty$, peuvent être vues comme les deux états du neurone (en repos ou en activation), dépendants de la force d'excitation z. S'il y a plus d'une couche cachée, des équations de la forme (5) sont utilisées pour le calcul de la valeur suivante des neurones cachés à partir de la couche actuelle. Quand le réseau Feed-Forward est utilisé pour une régression non linéaire, les sorties des neurones y_j sont généralement calculées par combinaison linéaire des neurones de la couche précédente c'est-à-dire

$$y_j = \sum_k \tilde{\omega}_{kj} h_k + \tilde{b}_j \tag{6}$$

Etant donné les données observées y_{oj}, les valeurs optimales des paramètres poids et biais (ω_{ki}, $\tilde{\omega}_{jk}$, b_k et $\tilde{b}_j$) sont trouvées en faisant un **apprentissage** du réseau, c'est-à-dire accomplir une optimisation non linéaire, où la fonction coût (ou la fonction objective)

$$J = \left\langle \sum_j (y_j - y_{oj})^2 \right\rangle \tag{7}$$

est minimisée, J n'étant autre que l'erreur quadratique moyenne (Mean Squared Error **MSE**) de la sortie. Le réseau trouve un ensemble de relations de régressions non linéaires $y_j = f_j(\mathbf{x})$. Pour approximer un ensemble de fonctions continues f_j, une seule couche de neurones cachés suffit, le nombre suffisant de neurones dans cette couche est arbitraire, c'est la qualité de l'approximation qui entre en jeu. Un réseau à une seule couche cachée est appelé réseau 2-couches, car il y a deux couches de mappage (Equations 5 et 6), allant de l'entrée à la sortie. Toutefois, il existe une autre convention pour calculer le nombre de couches, et quelques auteurs parlent d'un réseau 2-couches comme étant un réseau 3-couches, car il y a en réalité 3 couches de neurones.

3.4 Minimums locaux et sur-ajustement:

La difficulté principale de la méthode NN[1] est que l'optimisation non linéaire rencontre souvent un multiple de minimums locaux dans la fonction coût. Cela signifie qu'à partir de différentes conjectures initiales des paramètres, l'algorithme d'optimisation peut converger vers différents minimums locaux. Beaucoup d'approches ont été proposées afin d'alléger ce problème. Une approche souvent utilisée implique une multitude d'optimisations à partir de différents paramètres initiaux aléatoires, de sorte que, si tout va bien, toutes les exécutions n'aboutissent pas à des faibles minimums locaux.

Un autre piège avec la méthode NN est "overfitting" ou sur-ajustement, c'est-à-dire s'adapter au bruit dans les données, chose due à la flexibilité énorme du NN à adapter les données. Avec assez de neurones cachés, le NN peut ajuster (ou adapter) les données, y compris le bruit, à une exactitude arbitraire. Ainsi pour un réseau avec beaucoup de paramètres, l'atteinte du minimum global peut signifier une mauvaise solution adaptée. Habituellement, seulement une partie des données est employée pour apprendre (c'est-à-dire adapter) le modèle du NN, l'autre partie est réservée à la validation du modèle. Si trop de neurones cachés sont utilisés, le modèle d'ajustement des données de l'apprentissage sera excellent, mais le modèle adapté aux données de validation sera pauvre; cette méthode de travail permet aux chercheurs d'estimer le nombre approprié de neurones cachés à utiliser, en choisissant un

[1] Dorénavant, nous utiliserons la notation NN pour faire référence à un réseau de neurones.

nombre qui donne une meilleure erreur, pour l'apprentissage et pour la validation. Pendant le processus d'optimisation, il est généralement commun de surveiller le MSE des données d'apprentissage et des données de validation séparément. Quand le nombre d'itérations de l'algorithme d'optimisation augmente, le MSE calculé pour les données d'apprentissage diminue; cependant, au delà d'un certain nombre d'itérations, le MSE des données de validation commence à augmenter, indiquant le début du "overfitting", et par conséquent, le moment approprié pour arrêter le processus d'optimisation. Une autre approche pour éviter l'overfitting est d'ajouter des limites de pénalité des poids à la fonction coût (voir l'annexe C). Une autre approche est encore de calculer un ensemble de modèles de NN à partir de valeurs aléatoires différentes des paramètres initiaux. La moyenne de l'ensemble des solutions du NN tend à donner une solution plus douce que les différentes solutions individuelles du NN.

Si des qualifications de prévision doivent être estimées, une autre partie inutilisée des données doit être réservée pour des essais indépendants afin d'estimer les qualifications de prévision, comme pour les données de validation qui ont été déjà utilisées pour déterminer l'architecture modèle. Quelques auteurs échangent la terminologie pour les données de validation et les données du test, nous utiliserons indifféremment les deux termes pour désigner les données du test. Pour une qualité inférieure de l'ensemble des données (par exemple des données courtes, ou bruitées), les problèmes des minimums locaux et d'overfitting pourraient rendre les méthodes non linéaires des NN incapables d'offrir un avantage sur les méthodes linéaires.

Le réseau Feed-Forward a été appliqué à une variété de régressions et de classifications non linéaires des problèmes en sciences environnementales telles que la météorologie et l'océanographie, et a été étudié par *Gardner et Dorling* [1998] et *Hsieh et Tang* [1998]. Quelques exemples des applications récentes : utilisation du NN pour le diagnostic de tornade [*Marzban*, 2000], pour le calcul radiatif efficace du transfert dans les modèles généraux atmosphériques de circulation [*Chevallier et al.*, 2000], pour le recouvrement multi-paramètres satellitaire de la sonde spatiale Microwave/Imager (SSM/I) [*Gemmill et Krasnopolsky*, 1999], pour la récupération de vent à partir d'un Scatterometer [*Richaume et al.*, 2000], pour l'adaptation non linéaire statistique des modèles de rendement (MOS) [*Yuval and Hsieh*, 2003], pour le calcul efficace de la densité de l'eau de mer ou de la salinité à partir d'une équation d'état non linéaire [*Krasnopolsky et al.*, 2000], pour la prévision de la température de surface de mer du Pacifique tropical [*Tang et al.*, 2000 ; *Yuval*, 2001], et pour une atmosphère empirique dans un modèle couplé hybride d'atmosphère-océan du Pacifique tropical [*Tang and Hsieh*, 2002]. Pour les applications de NN en géophysique (exploration séismique, détermination

de lithologie de bien-notation, exploration et séismologie électromagnétiques des tremblements de terre), voir *Sandham et Leggett* [2003].

3.5 Analyse non linéaire de la composante principale NLPCA:

Nous allons ici nous limiter à l'étude des courbes ouvertes. Les courbes closes sont plus utilisées dans le cas des données périodiques ou se présentant sous forme d'ondes.

Comme la PCA trouve une ligne droite qui traverse "le milieu" du faisceau des données; la prochaine étape évidente est de généraliser la ligne droite à une courbe. *Kramer* [1991] a proposé un réseau neuronal, le modèle non linéaire basé sur la PCA (NLPCA), où la ligne droite est remplacée par une courbe continue pour rapprocher les données.

La différence fondamentale entre NLPCA et PCA est que PCA permet seulement un traçage linéaire (Equation 3) entre $\mathbf{x}$ et le PC u, alors que NLPCA permet un traçage non linéaire. Pour exécuter NLPCA, le réseau Feed-Forward de le figure 4 contient 3 couches de neurones cachées entre la couche d'entrée et celle de sortie.

La NLPCA est fondamentalement un réseau Feed-Forward standard avec 4 couches de fonctions de transfert traçant des entrées aux sorties. On peut voir le réseau de NLPCA comme composé de deux réseaux standards Feed-Forward 2-couches, placés l'un après l'autre. Le premier réseau 2-couches trace des entrées $\mathbf{x}$ à travers les couches cachées à la couche du goulot d'étranglement qui a seulement un seul neurone u, c'est-à-dire un traçage non linéaire $u = f(\mathbf{x})$. Inversement, le deuxième réseau Feed-Forward 2-couches trace du PC non linéaire (NLPC) u aux sorties $\mathbf{x}'$ de dimension d'origine (espace $\mathbf{x}$), avec comme objectif les sorties $\mathbf{x}' = g(u)$ qui doivent être aussi étroites que possible aux entrées $\mathbf{x}$, (le NN serait ainsi auto-associative).

Noter que $g(u)$ produit non linéairement une courbe dans le $\mathbf{x} - space$, par conséquent une approximation à une dimension des données originales. Pour réduire au minimum le MSE de cette approximation, la fonction de coût $J = \left\langle \left\| \mathbf{x} - \mathbf{x}' \right\|^2 \right\rangle$ est réduite au minimum, et de là on trouve les poids et les biais du NN. Serrant l'information d'entrée à travers le goulot d'étranglement avec seulement un neurone, on accomplit la réduction dimensionnelle. Des détails de la NLPCA sont donnés dans l'annexe C.

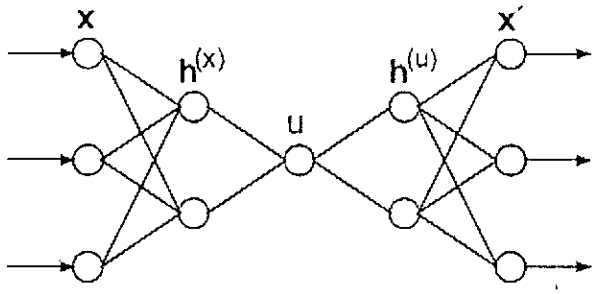


Figure 4: Diagramme schématique du modèle NN pour calculer la NLPCA. Il y a 3 couches de neurones cachés serrés entre la couche x d'entrée du côté gauche et la couche de sortie $x^{'}$ du côté droit. Du côté de l'entrée, la couche est appelée couche de codage, suivie de la couche dite 'bottleneck layer' ou en français' goulot d'étranglement ' (avec un seul neurone simple u), qui est alors suivi de la couche de décodage. Une fonction non linéaire fait passer de l'espace d'entrée de dimension élevée à l'espace du goulot d'étranglement à 1-dimension, suivi de la transformée inverse qui fait passer de nouveau de l'espace du goulot d'étranglement à l'espace original représenté par les sorties, qui doivent être aussi proches des entrées réduisant au minimum la fonction coût $J = \left\langle \left\| \mathbf{x} - \mathbf{x}^{'} \right\|^2 \right\rangle$.La compression de données est réalisée par le goulot d'étranglement, avec le neurone simple qui donne u, la composante principale non linéaire (NLPC).

En effet, la relation linéaire (Equation 3) dans la PCA est maintenant généralisée à $u = f(\mathbf{x})$, où f peut être fonction continue non linéaire représentable par un NN Feed-Forward traçant de la couche d'entrée à la couche du goulot d'étranglement, et au lieu de l'équation 4, $\left\langle \|\mathbf{x} - g(u)\|^2 \right\rangle$ est réduite au minimum. Le résidu $\mathbf{x} - g(u)$ peut être l'entrée du même réseau pour extraire le deuxième mode de NLPCA, et ainsi de suite pour les modes d'ordres supérieurs.

Le PCA classique est une version linéaire de ce NLPCA, ce qui peut être aisément vue après le remplacement de toutes les fonctions de transfert par la fonction identité, enlevant de ce fait le pouvoir de modélisation non linéaire du NLPCA. Alors le traçage en avant vers u comporte seulement une combinaison linéaire des variables originales comme dans le PCA.

4 Les paramètres du modèle et le programme de calculs:

4.1 Les paramètres du modèle :

Trouver les paramètres du modèle pour chaque mois du réseau de neurones, revient à spécifier :

1. Le type du réseau;
2. La dimension de l'entrée et de la sortie;
3. Le nombre de couches (suivant une convention);
4. Les fonctions de transfert à utiliser;
5. Les erreurs qui ont permis le choix de la configuration;
6. Le nombre de neurones dans chaque couche;
7. Les poids pour passer d'une couche à une autre;
8. Le biais de chaque neurone dans chaque couche.

En ce qui concerne notre application, les réponses sont les suivantes :

♦ Type du réseau : Réseau Feed-Forward.

♦ Dimension de l'entrée et de la sortie : pour l'apprentissage, l'entrée est une matrice 8×25, ainsi que la sortie. Pour le test, l'entrée et la sortie sont des matrices 8×5. A la fin de l'apprentissage et de la validation, notre réseau est capable de fournir, pour une latitude et une longitude donnée, la fraction d'insolation correspondante.

♦ Nombre de couches : pour la NLPCA, la configuration la plus utilisée, et qui a démontré son efficacité dans plusieurs travaux, est un réseau à 3 couches (sans compter les couches d'entrée et de sortie).

♦ Fonctions de transfert : voir la théorie de NLPCA et l'annexe 2.

♦ Erreurs et nombre de neurone: afin de trouver un minimum local acceptable, nous avons fait l'apprentissage et le test pour 08 neurones de 1 à 08 séparément (le même travail a été fait pour 1 neurone, puis 2 neurones, ...etc.). Les neurones dont on parle ici sont les neurones de la couche de codage et de la couche de décodage (ils sont égaux). Le nombre de neurones donnant le MSE le plus petit, donne la configuration du réseau pour le calcul du mode 1, le résidu de ce mode avec cette configuration est choisi pour effectuer l'adaptation, en calculant le mode 2, qui obéit au même raisonnement que pour le mode 1, c'est-à-dire que la configuration du réseau pour le mode 2 est celle qui donne le plus petit MSE du résidu du premier mode. Finalement, pour la couche centrale, il n'y a qu'un seul neurone simple.

♦ Poids et biais : ils sont les résultats de l'apprentissage et du test.

4.2 Le programme:

Le programme écrit en MATLAB, nous donne la main pour choisir le mois dont on veut modéliser les données :

1. La fonction **param** nous permet de fixer un certain nombre de paramètres : la pénalité, le pourcentage des données pour le test, les données à sauvegarder, la mise en échelle des données ou non, le nombre d'ensembles d'initialisation, le degré de non linéarité, la tolérance pour le sur ajustement, la valeur pour un arrêt prématuré de l'apprentissage en cas d'erreurs trop importantes. A la fin le nombre maximal des neurones dans les couches de codage et de décodage. Nos choix sont les suivants:

► La pénalité : le meilleur compromis entre désir de modélisation non linéaire efficace et erreur a été observé pour une pénalité = 0.2. Pour $P = 1$, les erreurs sont trop grandes, et on tend à une modélisation linéaire qui ne nécessite pas un réseau de neurones. Pour $P = 0$, le pouvoir non linéaire du réseau nous donne des résultats insignifiants.

► Données d'apprentissage : matrice 8×20, c'est-à-dire 80% des données.

► Données du test : matrice 8×5, c'est-à-dire 20% des données.

► Pas de mise en échelle des données car elles sont toutes comprises entre 0 et 1.

► Ensemble des initialisations : 25 fois des initialisations différentes (aléatoires)

► La tolérance pour le sur ajustement : 0.

► La valeur pour un arrêt prématuré de l'apprentissage en cas d'erreurs trop importantes : 0.001.

Nombre maximal de neurones : $m = 08$.

La fonction **nlpca1** permet de choisir le mois pour lequel on fait la modélisation, c'est le mode 1. L'apprentissage et le test prend dans ce cas environ une heure pour chaque mois. Les résultats ainsi que tous les paramètres choisis, pour les 8 cas (1 neurone, 2 neurones, ..., etc.) sont sauvegardés dans des fichiers (.mat), ce qui fait 8 fichiers pour le mode 1, pour chaque mois.

3. La fonction **nlpca2**, le même travail que **nlpca1** mais pour le résidu de la configuration qui a effectué le minimum d'erreurs.

5 Les résultats:

5.1 Les erreurs:

Le tableau suivant (Tableau 2) donne les erreurs trouvées pour chaque mois, le nombre de neurones (dans les couches de codage et de décodage), ainsi que le numéro de l'ensemble d'initialisation qui a permis de donner le minimum

local qui est le plus réduit des 8×25 exécutions, pour chaque mois et pour chaque mode (2 modes). Le **MSE** est donné en pourcentage (%).

Janvier					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
1	25	0.1598	7	13	0.0345
Fevrier					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
3	11	0.1464	5	22	0.0393
Mars					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
2	22	0.2183	3	2	0.0357
Avril					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
2	21	0.2712	5	9	0.0657
Mai					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
6	6	0.1958	8	7	0.0603
Juin					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
7	12	0.1801	2	22	0.0458
Juillet					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
7	24	0.6346	6	23	0.1079
Aout					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
2	24	0.7514	4	1	0.1287
Septembre					
	Mode1			*Mode2*	
m	*ens*	MSE	m	*ens*	MSE
2	24	0.4238	1	13	0.1239

Octobre					
		*Mode*1	*Mode*2		
m	*ens*	MSE	m	*ens*	MSE
3	12	0.1502	7	12	0.0281
Novembre					
		*Mode*1	*Mode*2		
m	*ens*	MSE	m	*ens*	MSE
7	1	0.1289	7	19	0.0383
Décembre					
		*Mode*1	*Mode*2		
m	*ens*	MSE	m	*ens*	MSE
7	3	0.1269	4	22	0.0378

Tableau 2: Valeur de m, le numéro de l'ensemble, et le MSE pour chaque mois, et pour les deux modes.

Les figures suivantes (figures 5, 6 et 7) illustrent l'entrée du réseau et la sortie résultante du modèle trouvé pour le mode 1. Nous avons préféré une représentation en deux dimensions, afin de mieux visualiser les erreurs, nous donnons donc pour chaque mois, la fraction d'insolation (mesurée et calculée) en fonction de la latitude, pour une région donnée, c'est-à-dire un intervalle de longitudes, les courbes en + représentent les valeurs calculées.

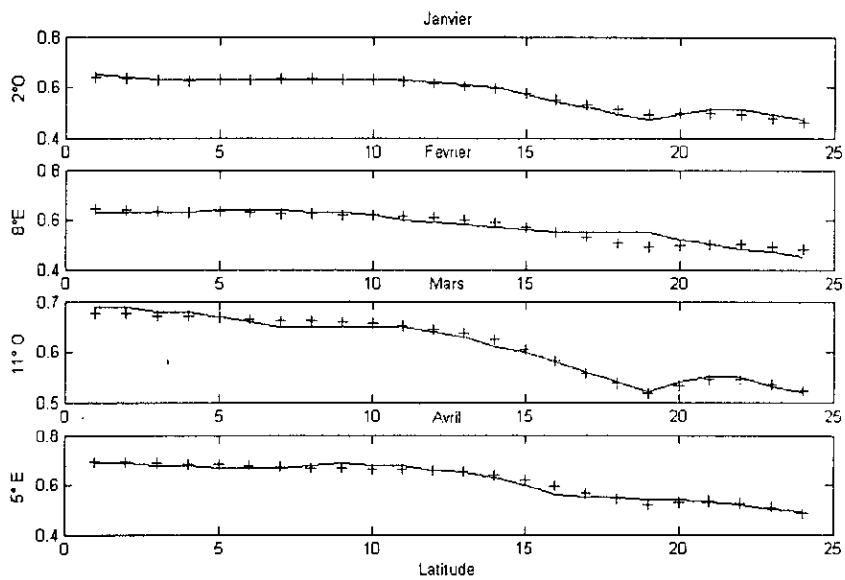


Figure 5: La fraction d'insolation, mesurée et calculée pour les mois de Janvier, Fevrier, Mars et Avril.

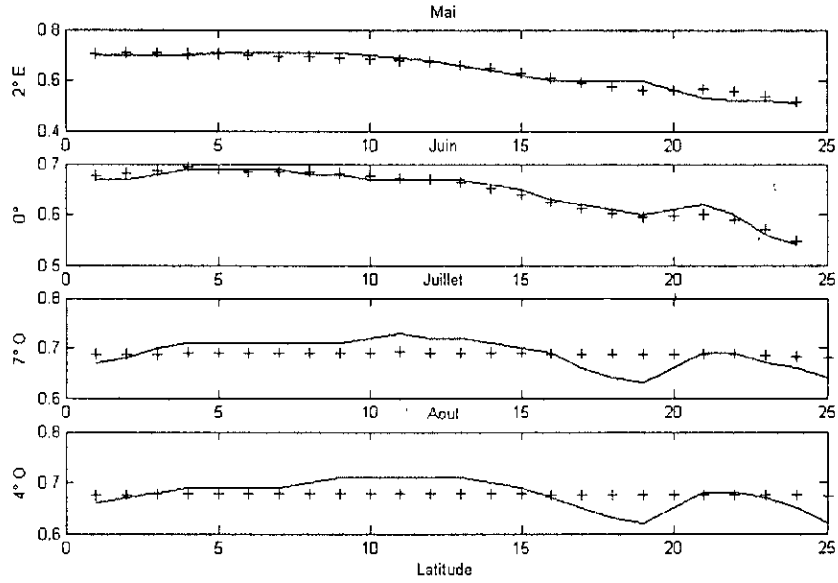


Figure 6: La fraction d'insolation, mesurée et calculée pour les mois de Mai, Juin, Juillet et Août.

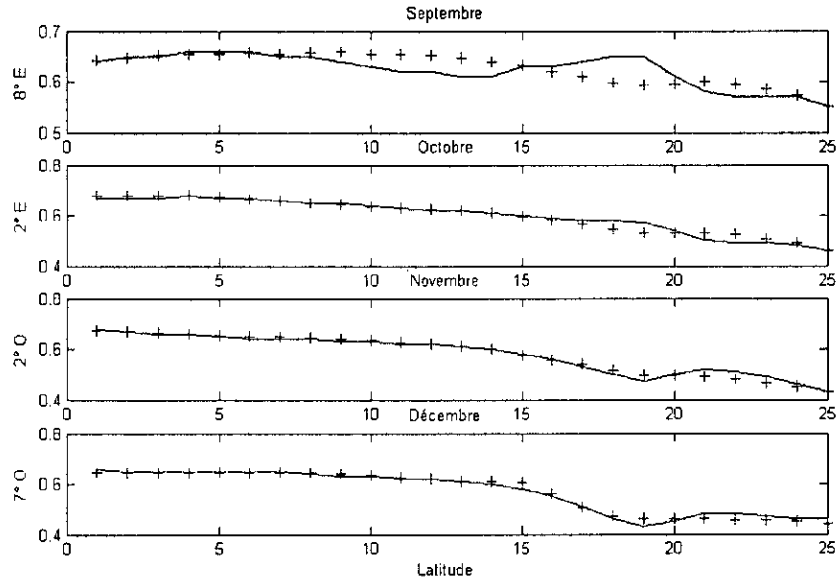


Figure 7: La fraction d'insolation, mesurée et calculée pour les mois de Septembre, Octobre, Novembre et Décembre.

Nous remarquons que la qualité des modèles est acceptable pour les mois de Janvier, Fevrier, Mars, Avril, Mai, Juin, Octobre, Novembre, et Décem-

bre; mais elle est médiocre, pour ne pas dire mauvaise, pour les trois mois restants: Juillet, Août et Septembre. Pour améliorer les modèles, on introduit le deuxième mode (mode 2) dont la sortie est additionnée au premier. Ainsi, nous obtenons des modèles remarquablement précis. La figure 8 illustre la fraction d'insolation mesurée et calculée pour les trois mois de Juillet, Août et Septembre; en prenant en considération le mode 2.

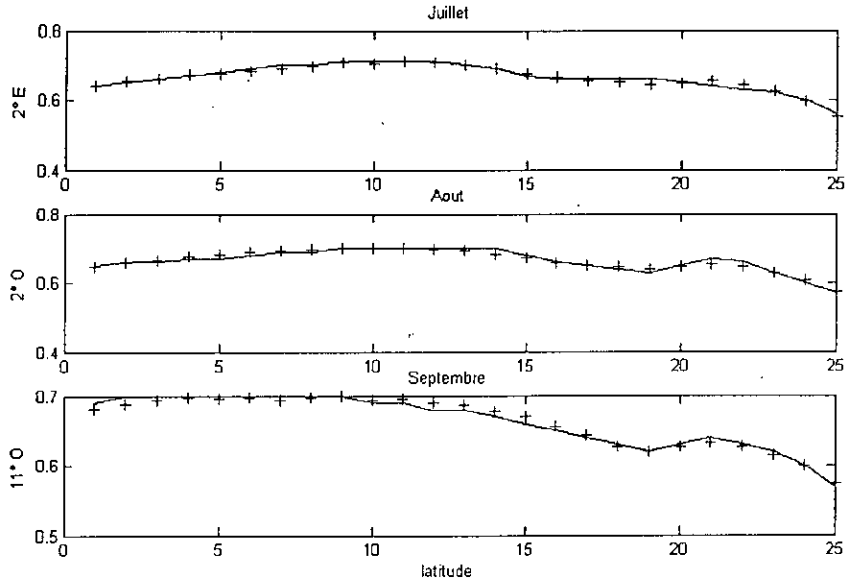


Figure 8: La fraction d'inolation mesurée et calculée pour les trois mois de Juillet, Août et Septembre; en prenant en considération le mode 2.

Nous constatons donc que l'introduction du mode 2 améliore considérablement la qualité des modèles. Cela pourra s'appliquer à tous les mois, les paramètres du modèle de chaque mois sont donnés dans les sections suivantes pour les deux modes.

On peut encore introduire des modes d'ordres supérieurs afin d'améliorer d'avantage les modèles; mais nous nous arrêtons à ce stade là, car aller au delà de deux modes exige plus de calculs, en plus, les résultats trouvés sont largement satisfaisants.

5.2 Les paramètres:

Nous donnons ici, l'algorithme de la NLPCA, afin de mieux situer chaque paramètre.

m = nombre de neurones dans les couches de codage et de décodage.

$l = 8$, nombre d'entrées.

$n = 25$, nombre d'échantillon de la variable x_i, dans notre cas c'est le nombre de latitudes pour chaque région.

k et i, variables muettes avec : $k = 1, ..., m$; $i = 1, ..., l$.

D'après la figure 4, les sorties de chaque neurone, les poids ainsi que les biais sont comme suit:

• sortie de neurone $h_k^{(x)}$:

$$h_k^{(x)} = f_1(\sum_{i=1}^{l} \omega_{ki}^{(x)} x_{ki} + b_k^{(x)}) \tag{8}$$

• sortie du neurone u :

$$u = f_2(\sum_{k=1}^{m} \omega_{1k}^{(x)} h_k^{(x)} + b_1^{(x)}) \tag{9}$$

• sortie du neurone $h_k^{(u)}$:

$$h_k^{(u)} = f_3(\omega_{k1}^{(u)} u + b_k^{(u)}) \tag{10}$$

• sortie du neurone $x_i^{'}$:

$$x_i^{'} = f_4(\sum_{k=1}^{m} \omega_{ik}^{(u)} h_k^{(u)} + b_i^{(u)}) \tag{11}$$

Sous forme matricielle, on écrit :

$$\mathbf{h}_k^{(x)} = f_1[\mathbf{W}^{(x)} \cdot \mathbf{x} + \mathbf{b}^{(x)}] \tag{12}$$

$$u = f_2[\mathbf{w}^{(x)} \cdot \mathbf{h}^{(x)} + \bar{b}^{(x)}] \tag{13}$$

$$h_k^{(u)} = f_3[\mathbf{w}^{(u)} \cdot u + \mathbf{b}^{(u)}] \tag{14}$$

$$x_i^{'} = f_4[\mathbf{W}^{(u)} \cdot \mathbf{h}^{(u)} + \bar{\mathbf{b}}^{(u)}] \tag{15}$$

Avec :

► $\mathbf{W}^{(x)}$: matrice $m \times l$ contenant les $\omega_{ki}^{(x)}$, les poids de la couche entrée à la couche codante.

► $\mathbf{b}^{(x)}$: vecteur $m \times 1$ contenant les $b_k^{(x)}$, les biais des neurones de la couche codante.

► $\mathbf{w}^{(x)}$: vecteur $1 \times m$ contenant les $\omega_{1k}^{(x)}$, les poids de la couche codante au neurone central.

► $\bar{b}^{(x)}$: scalaire, le biais du neurone central.

► $\mathbf{w}^{(u)}$: vecteur $m \times 1$ contenant les $\omega_{k1}^{(u)}$, les poids du neurone central à la couche décodante.

► $\mathbf{b}^{(u)}$: vecteur $m \times 1$ contenant les $b_k^{(u)}$, les biais des neurones de la couche décodante.

► $\mathbf{W}^{(u)}$: matrice $l \times m$ contenant les $\omega_{ik}^{(u)}$, les poids de la couche décodante à la couche de sortie.

► $\bar{\mathbf{b}}^{(u)}$: vecteur $l \times 1$ contenant les $\bar{b}_i^{(u)}$, les biais des neurones de la couche de sortie.

► $\mathbf{x}$: la matrice contenant les données x_{ki} (matrice $l \times n$).

► $\mathbf{h}^{(x)}$: le vecteur contenant les sorties des neurones de la couche codante, les $h_k^{(x)}$ (vecteur $m \times 1$).

► u : la sortie du neurone central.

► $\mathbf{h}^{(u)}$: le vecteur contenant les sorties des neurones de la couche décodante, les $h_k^{(u)}$ (vecteur $m \times 1$).

► $\mathbf{x}_i^{'}$: la sortie du neurone i de la couche de sortie.

On déduit donc que le nombre de paramètres est égal à :

► poids : $2 \times (l \times m) + 2 \times (m \times 1)$.

► biais : $2 \times (m \times 1) + (l \times 1) + 1$.

Le total est de $(2 \times l \times m + 4 \times m + l + 1)$ paramètres.

Valeurs des paramètres:

Les valeurs des paramètres du modèle de chaque mois, et pour les deux modes sont données sous forme matricielle à la fin de ce chapitre. Les vecteurs de la forme $x = [.....]'$, indiquent des vecteurs colonnes, une notation empruntée à MATLAB.

6 Interprétation des résultats et conclusions:

Dans cette étude, nous avons essayé de trouver une meilleure approximation des données de la durée d'insolation de l'Algérie. Nous avons procédé à l'aide d'une courbe ouverte faite par un réseau de neurones appliqué à l'analyse de la composante principale. Cette méthode est largement utilisée dans l'analyse des données météorologiques et océanographiques, afin de

décomposer les données en plusieurs composantes: la première est la composante principale, la seconde est le résidu de celle ci, et ainsi de suite pour extraire les modes d'ordre supérieur. Nous avons utilisé cette méthode à des fins de modélisation, c'est-à-dire trouver un filtre (boite noire) qui permet de fournir, pour les deux entrées: longitude et latitude, une sortie correspondante, qui est la fraction d'insolation. Les résultats trouvés pour les mois de Janvier, Février, Mars, Avril, Mai, Juin, octobre, Novembre, et Décembre, sont acceptables sans avoir recours au mode 2; donc on peut se limiter à la NLPC1, car l'erreur (ou le résidu) est suffisamment réduite pour la considérée comme une composante sans effet dans les données. Pour les mois de Juillet, Août, et Septembre, nous remarquons que la seule NLPC1 n'est pas suffisante pour représenter toute l'information contenue dans les données, et que l'erreur est importante, deux choix se présentent afin de venir à bout de ce problème :

1. Augmenter le nombre de neurones dans les couches de codage et de décodage; pour notre étude, nous nous somme arrêtés à 08 neurones, et l'erreur pour les trois mois sus-cités est restée considérable. Comme il est impossible d'augmenter indéfiniment ce nombre, nous avons choisi de nous arrêter à ce stade des simulations, qui seront probablement reprises dans l'avenir (augmentation du nombre de neurones). Il faut noter ici qu'un minimum local acceptable, pourra être trouvé, soit pour 09 neurones, soit pour plus ; c'est le problème classique des réseaux de neurones « trouver la meilleur configuration ».
2. Exploiter les pouvoirs de la méthode NLPCA: au lieu de se contenter de prendre une seule composante principale, on en prend deux ou plus. Ayant calculé la NLPC2 pour tous les mois, on dispose donc d'une deuxième composante qui s'ajoutera à la première, ce qui va permettre d'ajuster le modèle de ces mois afin de réduire l'erreur. Ce deuxième choix ; même s'il a l'inconvénient de présenter beaucoup de calculs, donne des résultats remarquablement précis. C'est cette deuxième solution que nous avons choisie, et nous avons donné les paramètres pour les deux modes. Pour les mois contenant deux modes, le résidu du premier est réinjectée dans un réseau de même nombre de couches, et avec apprentissage, on aura sa configuration, c'est-à-dire le nombre de neurones dans les couches de codage et de décodage. Finalement, la sortie de notre modèle est trouvée par le principe de superposition (mode1+mode2). La figure 9 illustre schématiquement cette procédure.

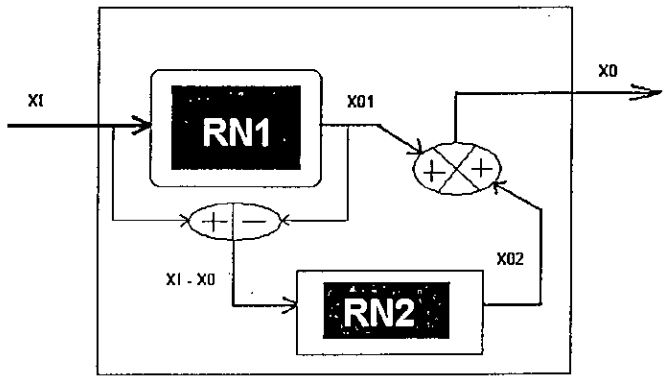


Figure 9: Illustration de la procedure de calcul de la sortie du modèle en cas de prise en compte du mode 2. Le résidu du mode 1, $XI - X0$ trouvé à partir du premier réseaux de neurone RN1, constitu une entrée au deuxième réseau RN2. La sortie finale est la somme des sorties des deux modes.

Malgré des résultats satisfaisants, la NLPCA présente quelques inconvénients :

► Le critère de validation étant le MSE, c'est-à-dire l'erreur quadratique moyenne, on peut la trouver suffisamment petite ; mais pour quelques échantillons des données, l'erreur reste considérable.

► Le temps de calcul est long, en raison des itérations et du nombre important d'initialisations, et de l'augmentation du nombre de neurones, cela rend la méthode lourde en calculs: le calcul de la NLPC1 pour chaque mois prend environ 40 minutes et pour la NLPC2, un peu plus.

► La nécessité d'aller parfois au delà du mode2, complique les calculs.

Toutefois, l'analyse non linéaire de la composante principale reste un moyen efficace pour la modélisation des données multivariables.

★★★★★★★★★★★★★★★★★★★★★★★★★★★★
★★ Les paramètres des deux modes ★★
★★★★★★★★★★★★★★★★★★★★★★★★★★★★

Paramètres du Mode 01

Janvier

- $\mathbf{W}^{(x)} = [0.0075\ 0.0075\ 0.0071\ 0.0074\ 0.0073\ 0.0085\ 0.0094\ 0.0078]$
- $\mathbf{b}^{(x)} = [0.1054]'$
- $\mathbf{w}^{(x)} = [3.5410]$
- $\bar{b}^{(x)} = [-1.2016]$
- $\mathbf{w}^{(u)} = [2.6543]$
- $\mathbf{b}^{(u)} = [1.8821]$
- $\mathbf{W}^{(u)} = [0.5518\ 0.5480\ 0.5540\ 0.5562\ 0.5550\ 0.5529\ 0.5421\ 0.5540]'$
- $\bar{\mathbf{b}}^{(u)} = [1.5140\ 1.6485\ 1.7348\ 1.6962\ 1.6552\ 1.7593\ 2.0692\ 1.8108]'$

Fevrier

- $\mathbf{W}^{(x)} = \begin{pmatrix} 0.0018 & 0.0082 & 0.0045 & 0.0017 & 0.0093 & 0.0054 & 0.0019 & 0.0097 \\ 0.0063 & 0.0018 & 0.0102 & 0.0061 & 0.0019 & 0.0105 & 0.0064 & 0.0019 \\ 0.0108 & 0.0065 & 0.0024 & 0.0130 & 0.0076 & 0.0021 & 0.0126 & 0.0076 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [0.3721\ 0.0376\ 0.0992]'$
- $\mathbf{w}^{(x)} = [-0.4971\ -2.5303\ -1.5191]$
- $\bar{b}^{(x)} = [-0.1131]$
- $\mathbf{w}^{(u)} = [1.8925\ -2.6289\ 2.2439]'$
- $\mathbf{b}^{(u)} = [1.0083\ -1.6311\ -0.2034]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} -0.4959 & -0.8288 & -0.0094 \\ -0.4919 & -1.2610 & -1.0744 \\ -0.7018 & -0.8527 & 0.7829 \\ 0.7492 & 1.2700 & 0.9863 \\ 0.5300 & 0.6807 & 1.1331 \\ 0.9176 & 0.0739 & 0.4934 \\ 0.0527 & 0.0462 & -0.1133 \\ -0.3942 & 0.0107 & 0.1743 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [0.3007\ 0.6062\ 0.3506\ 0.2528\ -0.0392\ -0.2767\ 0.1173\ 0.2789]'$

Mars

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0001 & -0.0044 & -0.0003 & -0.0046 & -0.0003 & -0.0066 & -0.0002 & -0.0050 \\ -0.0001 & -0.0056 & -0.0001 & -0.0057 & -0.0001 & -0.0072 & -0.0001 & -0.0076 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [0.1913\ 0.0595]'$
- $\mathbf{w}^{(x)} = [0.0733\ 3.7541]$

- $\bar{b}^{(x)} = [0.5763]$
- $\mathbf{w}^{(u)} = [-2.8177\ -0.6410]'$
- $\mathbf{b}^{(u)} = [1.9364\ 2.1281]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} 1.7195 & 2.1125 \\ 2.1073 & 2.2617 \\ 0.9219 & -0.2364 \\ 0.0494 & 0.6046 \\ 1.9720 & 2.0280 \\ 2.1479 & 1.9046 \\ 0.0481 & 0.9357 \\ 0.4323 & -0.4470 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.1530\ 0.6772\ 0.9543\ -0.1461\ 0.6800\ 0.3224\ 0.1585\ 1.1295]'$

==========

Avril

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0014 & -0.0139 & -0.0009 & -0.0141 & -0.0018 & -0.0114 & -0.0013 & -0.0116 \\ -0.0013 & -0.0119 & -0.0016 & -0.0118 & -0.0009 & -0.0151 & -0.0012 & -0.0106 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [-0.8001\ 0.0510]'$
- $\mathbf{w}^{(x)} = [0.6078\ 3.5211]$
- $\bar{b}^{(x)} = [-0.2603]$
- $\mathbf{w}^{(u)} = [2.4342\ -0.4657]'$
- $\mathbf{b}^{(u)} = [1.4789\ 1.9098]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} -1.2374 & -1.3237 \\ -1.2797 & -1.2527 \\ -1.2126 & -1.1860 \\ -1.2707 & -1.0947 \\ 1.1357 & 0.6323 \\ 1.1868 & -0.4544 \\ 0.2163 & 1.2650 \\ 1.1935 & -0.0778 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.7889\ -0.3199\ -0.8353\ 0.7743\ 0.1279\ -0.8878\ -0.8441\ 0.4516]'$

==========

Mai

- $\mathbf{W}^{(x)} = \begin{pmatrix} 0.0109 & 0.0068 & 0.0062 & -0.0039 & 0.0076 & 0.0012 & 0.0129 & 0.0081 \\ 0.0074 & -0.0045 & 0.0092 & 0.0014 & 0.0140 & 0.0086 & 0.0080 & -0.0050 \\ 0.0098 & 0.0015 & 0.0131 & 0.0082 & 0.0076 & -0.0046 & 0.0094 & 0.0016 \\ 0.0122 & 0.0077 & 0.0071 & -0.0045 & 0.0088 & 0.0013 & 0.0108 & 0.0068 \\ 0.0064 & -0.0042 & 0.0078 & 0.0012 & 0.0110 & 0.0069 & 0.0064 & -0.0040 \\ 0.0080 & 0.0012 & 0.0085 & 0.0054 & 0.0049 & -0.0032 & 0.0062 & 0.0010 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [-0.0617\ 0.0002\ 0.0178\ -0.1956\ -0.2045\ -0.2224]'$
- $\mathbf{w}^{(x)} = [1.5141\ 0.9273\ 0.8566\ -0.5674\ 1.0955\ 0.1805]$
- $\bar{b}^{(x)} = [-0.6858]$

► $\mathbf{w}^{(u)} = [0.8619\ -0.0776\ 1.8703\ 0.6068\ -0.6547\ -2.1591]'$

► $\mathbf{b}^{(u)} = [0.3796\ 0.5865\ 1.0488\ 0.4064\ -0.4292\ -1.3572]'$

► $\mathbf{W}^{(u)} = \begin{pmatrix} -0.0362 & 0.8151 & 0.0662 & 0.7194 & -0.0652 & 0.1341 \\ 0.4434 & 0.7591 & 0.4721 & 0.2465 & 0.0070 & 0.0182 \\ 0.2541 & -0.0788 & -0.1845 & 0.1311 & 0.6843 & 1.1019 \\ 0.9519 & 0.1163 & 0.4957 & 0.7596 & 0.2813 & 0.5279 \\ 0.0638 & 0.2178 & 0.0650 & 0.5437 & 0.8548 & 1.0444 \\ 0.4399 & 0.4712 & -0.4718 & 0.5867 & -0.0178 & -0.0204 \\ -0.0283 & 0.0264 & -0.2447 & 0.2413 & -0.5342 & -0.4687 \\ -0.7113 & -1.0630 & -0.7874 & -0.4178 & -0.6658 & -0.3111 \end{pmatrix}$

► $\tilde{\mathbf{b}}^{(u)} = [0.6349\ 1.0184\ 1.0102\ 0.9990\ 0.7523\ 0.9982\ 1.0303\ 0.9233]'$

==========

Juin

► $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0135 & -0.0060 & -0.0015 & -0.0180 & -0.0060 & -0.0045 & 0.0018 & -0.0128 \\ -0.0057 & -0.0016 & -0.0173 & -0.0071 & -0.0022 & 0.0021 & -0.0144 & -0.0055 \\ -0.0019 & -0.0192 & -0.0080 & -0.0049 & 0.0024 & -0.0126 & -0.0055 & -0.0017 \\ -0.0159 & -0.0069 & -0.0041 & 0.0020 & -0.0108 & -0.0048 & -0.0013 & -0.0148 \\ -0.0061 & -0.0036 & 0.0011 & -0.0094 & -0.0039 & -0.0012 & -0.0119 & -0.0041 \\ -0.0030 & 0.0017 & -0.0084 & -0.0040 & -0.0009 & -0.0105 & -0.0041 & -0.0036 \\ 0.0010 & -0.0078 & -0.0034 & -0.0009 & -0.0097 & -0.0037 & -0.0024 & 0.0010 \end{pmatrix}$

► $\mathbf{b}^{(x)} = [0.0843\ -0.2251\ -0.0482\ -0.0120\ 0.0724\ 0.3203\ 0.2966]'$

► $\mathbf{w}^{(x)} = [1.6101\ 0.7662\ 0.2136\ 2.1932\ 0.8960\ 0.6003\ -0.2868]$

► $\tilde{b}^{(x)} = [-0.4886]$

► $\mathbf{w}^{(u)} = [-0.9841\ -0.2458\ 0.7461\ 0.4248\ 1.5751\ 0.9526\ 0.8629]'$

► $\mathbf{b}^{(u)} = [-0.3005\ 0.0578\ 0.1614\ -0.1767\ 0.8504\ 0.5191\ 0.4211]'$

► $\mathbf{W}^{(u)} = \begin{pmatrix} -0.2133 & 0.1635 & 0.6857 & 0.2773 & 0.5268 & 0.3224 & -0.0048 \\ 0.0662 & 0.3301 & -0.4051 & 0.0549 & -0.0925 & 0.9713 & -0.4576 \\ -0.2020 & -0.0965 & -0.6442 & -0.4427 & 0.7363 & -0.6533 & -0.6147 \\ -0.1071 & -0.6940 & 0.4070 & 0.7412 & -0.6370 & -0.4668 & -0.6961 \\ -0.2712 & -0.0270 & 0.6991 & -0.6241 & -0.7144 & -1.2469 & -0.7669 \\ -0.9032 & 0.3332 & -0.7962 & -0.2409 & -0.9912 & -0.8259 & 0.3993 \\ -0.5133 & 0.3225 & -0.8676 & -0.3636 & -0.7093 & 0.2204 & -0.5290 \\ -0.0591 & -1.0801 & -0.4528 & -0.2924 & 0.5408 & -0.5013 & -0.0023 \end{pmatrix}$

► $\tilde{\mathbf{b}}^{(u)} = [0.3311\ -0.0429\ -0.0576\ -0.2496\ -0.2118\ 0.4233\ 0.5058\ 0.3160]'$

==========

Juillet

- $\mathbf{W}^{(x)} = \begin{pmatrix} 0.0061 & -0.0019 & 0.0042 & -0.0007 & 0.0002 & 0.0024 & -0.0042 & 0.0054 \\ -0.0017 & 0.0038 & -0.0006 & 0.0002 & 0.0019 & -0.0036 & 0.0075 & -0.0024 \\ 0.0061 & -0.0008 & 0.0003 & 0.0031 & -0.0051 & 0.0066 & -0.0021 & 0.0046 \\ -0.0007 & 0.0003 & 0.0028 & -0.0045 & 0.0057 & -0.0018 & 0.0041 & -0.0007 \\ 0.0002 & 0.0024 & -0.0038 & 0.0039 & -0.0012 & 0.0026 & -0.0005 & 0.0001 \\ 0.0017 & -0.0027 & 0.0048 & -0.0015 & 0.0031 & -0.0005 & 0.0002 & 0.0020 \\ -0.0032 & 0.0056 & -0.0018 & 0.0040 & -0.0007 & 0.0002 & 0.0024 & -0.0038 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [-0.2655\ 0.0255\ -0.0147\ -0.6787\ 0.8377\ -0.6136\ -0.6441]'$
- $\mathbf{w}^{(x)} = [-0.8485\ 0.2663\ -0.6158\ 0.1494\ -0.0804\ -0.4744\ 0.8533]$
- $\bar{b}^{(x)} = [-0.5179]$
- $\mathbf{w}^{(u)} = [-0.3402\ 0.0139\ 0.2921\ -1.0085\ -0.7105\ -0.1945\ 0.2736]'$
- $\mathbf{b}^{(u)} = [-0.0275\ -0.6023\ -0.6502\ 0.0474\ -0.8064\ -0.2088\ 0.3051]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} 0.7766 & 1.6637 & -1.1620 & -1.1132 & 0.6425 & -1.0932 & -0.6610 \\ 1.6989 & 0.4768 & -1.0008 & -1.5976 & -0.6051 & -1.3271 & 0.3464 \\ -1.1340 & -1.3550 & -0.4770 & -1.0589 & -0.7304 & -1.6842 & 0.9101 \\ -0.9597 & 0.5849 & -1.2495 & 1.3400 & -0.7263 & 0.9751 & 0.3470 \\ -0.0564 & 1.6273 & 1.3104 & 0.9919 & -0.3977 & 1.1816 & 0.1691 \\ -0.2267 & 0.1073 & 0.0267 & -0.2590 & 1.4548 & 1.0751 & 0.3055 \\ 0.1699 & -1.2779 & -0.0162 & -0.0240 & 1.3504 & 1.5997 & -0.6067 \\ 0.0718 & -1.2506 & 1.6214 & -0.6791 & 0.9744 & -1.6199 & 0.2129 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.4159\ -0.1415\ -0.8949\ -1.1650\ 0.5843\ -0.7118\ -0.0259\ -1.3809]'$

==========

Août

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0010 & 0.0044 & -0.0012 & 0.0048 & -0.0018 & 0.0075 & -0.0019 & 0.0075 \\ -0.0019 & 0.0067 & -0.0013 & 0.0054 & -0.0016 & 0.0063 & -0.0015 & 0.0056 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [-0.6069\ -0.6670]'$
- $\mathbf{w}^{(x)} = [-0.3342\ 1.3451]$
- $\bar{b}^{(x)} = [-0.1349]$
- $\mathbf{w}^{(u)} = [0.2169\ 0.8718]'$
- $\mathbf{b}^{(u)} = [0.5563\ 0.9526]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} -1.7375 & 1.1942 \\ 0.0253 & 0.9746 \\ 0.8471 & 0.7209 \\ 0.8619 & -0.9440 \\ 1.4707 & 0.1087 \\ 1.4080 & -1.2209 \\ 0.5587 & 1.0192 \\ 1.7121 & 1.4982 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [0.8353\ 0.1559\ 0.1909\ 0.6799\ 0.1574\ 0.0705\ -0.2017\ 0.5715]'$

==========

Septembre

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0067 & -0.0020 & -0.0064 & -0.0031 & -0.0066 & -0.0048 & -0.0094 & -0.0044 \\ -0.0077 & -0.0035 & -0.0093 & -0.0026 & -0.0092 & -0.0016 & -0.0062 & -0.0019 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [0.0740\ 0.0991]'$
- $\mathbf{w}^{(x)} = [-2.9368\ -1.1928]$
- $\bar{b}^{(x)} = [-0.5062]$
- $\mathbf{w}^{(u)} = [0.7540\ 0.8436]'$
- $\mathbf{b}^{(u)} = [1.1741\ 1.1937]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} 1.7471 & 0.1379 \\ -1.0635 & 1.6231 \\ 0.5413 & 1.5980 \\ 1.7462 & 1.5715 \\ 0.6847 & -0.8793 \\ 2.3373 & 0.5535 \\ 1.9550 & -0.6097 \\ 1.2781 & 1.7092 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [0.8141\ 0.5423\ 0.8222\ 0.8042\ 0.8893\ 0.6872\ 0.9038\ 0.9510]'$

==========

Octobre

- $\mathbf{W}^{(x)} = \begin{pmatrix} 0.0021 & 0.0112 & 0.0062 & 0.0022 & 0.0126 & 0.0068 & 0.0021 & 0.0129 \\ 0.0071 & 0.0023 & 0.0120 & 0.0063 & 0.0019 & 0.0119 & 0.0062 & 0.0020 \\ 0.0120 & 0.0062 & 0.0020 & 0.0144 & 0.0078 & 0.0020 & 0.0140 & 0.0073 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [0.4180\ 0.0360\ 0.1454]'$
- $\mathbf{w}^{(x)} = [-0.5120\ -2.5816\ -1.4108]$
- $\bar{b}^{(x)} = [-0.0058]$
- $\mathbf{w}^{(u)} = [1.6909\ -2.5880\ 1.9353]'$
- $\mathbf{b}^{(u)} = [0.8428\ -1.5652\ -0.2901]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} -0.5436 & -0.8695 & 0.0183 \\ -0.4177 & -1.1426 & -0.9478 \\ -0.5147 & -0.7307 & 0.8745 \\ 0.8038 & 1.2565 & 0.9180 \\ 0.4864 & 0.5746 & 0.9430 \\ 0.8518 & 0.1224 & 0.5434 \\ 0.0789 & 0.0388 & -0.1274 \\ -0.3970 & -0.0516 & 0.1612 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [0.2876\ 0.5832\ 0.3453\ 0.2513\ -0.0325\ -0.2395\ 0.1332\ 0.2944]'$

==========

Novembre

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0446 & 0.0329 & -0.0467 & -0.0490 & 0.0648 & 0.0257 & -0.0308 & -0.0457 \\ 0.0332 & -0.0480 & -0.0500 & 0.0663 & 0.0279 & -0.0321 & -0.0442 & 0.0318 \\ -0.0453 & -0.0480 & 0.0622 & 0.0265 & -0.0307 & -0.0396 & 0.0308 & -0.0434 \\ -0.0456 & 0.0601 & 0.0258 & -0.0297 & -0.0397 & 0.0287 & -0.0408 & -0.0428 \\ 0.0559 & 0.0250 & -0.0274 & -0.0387 & 0.0277 & -0.0396 & -0.0416 & 0.0565 \\ 0.0256 & -0.0286 & -0.0423 & 0.0318 & 0.0456 & -0.0478 & 0.0636 & 0.0268 \\ -0.0308 & -0.0394 & -0.0407 & 0.0285 & 0.0411 & 0.0548 & 0.0240 & -0.0278 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [0.3766\ -0.2587\ 0.0580\ 0.1005\ -0.2823\ 0.5234\ 0.5314]'$
- $\mathbf{w}^{(x)} = [5.4696\ -3.9849\ 6.4152\ 6.5606\ -7.8390\ -5.5771\ 4.6185]$
- $\bar{b}^{(x)} = [1.4615]$
- $\mathbf{w}^{(u)} = [-0.1847\ 0.1499\ 0.1144\ -0.4696\ -0.0906\ 0.1984\ -0.3114]'$
- $\mathbf{b}^{(u)} = [-0.7699\ 0.0372\ -1.2142\ 2.7065\ -0.6968\ -1.1667\ -0.4106]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} -0.0877 & 0.0013 & 0.5578 & -1.1838 & -0.0861 & 0.0303 & 0.9330 \\ 0.1550 & -0.3226 & -0.7623 & 0.2533 & -1.0327 & -0.0582 & -0.0260 \\ 0.7148 & 0.2240 & -0.1435 & -0.7462 & 0.3929 & -1.4041 & -0.1137 \\ -0.2186 & 0.2216 & 0.4339 & 0.6132 & 0.8233 & -0.3463 & 2.0017 \\ 0.1088 & -0.1045 & -0.6245 & 0.2313 & -0.0814 & 0.0451 & 0.3883 \\ 0.2122 & 0.3342 & -0.8813 & -0.1341 & -0.0408 & -0.0655 & 0.4397 \\ -1.0723 & 1.2962 & -0.2849 & 0.0610 & 0.0610 & -0.6559 & 0.0852 \\ -0.1727 & -0.0560 & 0.2013 & 0.1417 & 0.4721 & 0.2275 & 0.2403 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.2912\ -0.5306\ 0.9232\ -2.0343\ 0.3386\ 0.1064\ 1.2795\ 0.3282]'$

==========

Décembre

- $\mathbf{W}^{(x)} = \begin{pmatrix} 0.0340 & 0.0372 & -0.0631 & 0.0148 & 0.0338 & 0.0170 & 0.0225 & 0.0361 \\ 0.0404 & -0.0697 & 0.0158 & 0.0361 & 0.0182 & 0.0242 & 0.0361 & 0.0407 \\ -0.0687 & 0.0164 & 0.0373 & 0.0179 & 0.0236 & 0.0324 & 0.0364 & -0.0659 \\ 0.0164 & 0.0352 & 0.0171 & 0.0231 & 0.0313 & 0.0346 & -0.0589 & 0.0143 \\ 0.0345 & 0.0171 & 0.0159 & 0.0306 & 0.0357 & -0.0615 & 0.0144 & 0.0311 \\ 0.0163 & 0.0215 & 0.0312 & 0.0377 & -0.0593 & 0.0116 & 0.0326 & 0.0163 \\ 0.0207 & 0.0316 & 0.0343 & -0.0497 & 0.0144 & 0.0278 & 0.0146 & 0.0202 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [-0.1071\ -0.1255\ 0.2562\ 0.0258\ -0.1201\ -0.0634\ -0.1278]'$
- $\mathbf{w}^{(x)} = [-6.3045\ -7.1425\ 12.0391\ -2.9106\ -6.5155\ -3.2601\ -4.2715]$
- $\bar{b}^{(x)} = [1.2860]$
- $\mathbf{w}^{(u)} = [-0.0965\ 1.1623\ -1.7652\ 2.2180\ 2.2957\ -0.1541\ 1.3979]'$
- $\mathbf{b}^{(u)} = [1.6650\ -0.1305\ 0.0160\ 0.0954\ 0.0557\ 0.4117\ -0.1029]'$

$$\blacktriangleright\ \mathbf{W}^{(u)} = \begin{pmatrix} 0.1734 & 0.0225 & 0.5192 & 0.5414 & 0.7603 & 0.4089 & 0.2981 \\ 0.1854 & -0.1589 & 0.1535 & -0.0478 & 0.0112 & -0.0892 & 0.3555 \\ 0.1098 & -0.1301 & 0.1129 & 0.2210 & 0.4398 & 0.0312 & 0.7508 \\ 0.6395 & 0.8280 & 0.2120 & -0.2614 & 0.1890 & 0.2943 & -0.1784 \\ 0.3654 & 0.4751 & 0.7299 & -0.0741 & 0.3074 & -0.0898 & -0.0686 \\ 0.1765 & -0.0766 & -0.1031 & -0.2061 & 0.1327 & 0.6357 & 0.7066 \\ 0.3961 & 0.3302 & 0.3640 & 0.5340 & 0.0721 & 0.3516 & 0.2381 \\ 0.0012 & 0.2391 & -0.0061 & 0.5430 & -0.0626 & 0.1344 & 0.2474 \end{pmatrix}$$

$$\blacktriangleright\ \bar{\mathbf{b}}^{(u)} = [0.1479\ 0.2696\ -0.0626\ -0.0540\ -0.2657\ -0.0028\ -0.6734\ 0.2601]'$$

Paramètres du Mode 02

Janvier

$$\blacktriangleright\ \mathbf{W}^{(x)} = \begin{pmatrix} 0.0005 & 0.0005 & 0.0686 & 0.0393 & -0.0024 & -0.0001 & -0.0019 & 0.0015 \\ 0.0006 & 0.0517 & 0.0328 & 0.0051 & -0.0007 & 0.0024 & 0.0010 & -0.0011 \\ 0.0416 & 0.0239 & 0.0027 & -0.0012 & -0.0013 & 0.0006 & -0.0004 & 0.0036 \\ 0.0006 & -0.0003 & 0.0011 & 0.0000 & -0.0003 & 0.0000 & -0.0245 & -0.0138 \\ -0.0004 & 0.0006 & 0.0013 & -0.0015 & 0.0013 & -0.0529 & -0.0327 & -0.0029 \\ -0.0000 & 0.0005 & -0.0016 & -0.0010 & -0.0430 & -0.0290 & -0.0014 & 0.0015 \\ -0.0019 & 0.0014 & -0.0003 & -0.0300 & -0.0187 & -0.0038 & 0.0001 & -0.0008 \end{pmatrix}$$

$$\blacktriangleright\ \mathbf{b}^{(x)} = [-3.8660\ 2.9153\ -0.5199\ -0.1511\ 0.8066\ 3.6169\ -2.2200]'$$

$$\blacktriangleright\ \mathbf{w}^{(x)} = [1.9980\ -0.3776\ -3.2534\ -1.5822\ -0.2913\ 1.1518\ -0.7127]$$

$$\blacktriangleright\ \bar{b}^{(x)} = [-0.3637]$$

$$\blacktriangleright\ \mathbf{w}^{(u)} = [0.5153\ 0.6307\ -0.2319\ 1.8130\ 0.0496\ 0.7273\ -0.3083]'$$

$$\blacktriangleright\ \mathbf{b}^{(u)} = [-0.1108\ -0.2051\ 0.3937\ -0.3528\ -0.6927\ 0.4337\ -0.0810]'$$

$$\blacktriangleright\ \mathbf{W}^{(u)} = \begin{pmatrix} -1.6578 & -0.4399 & -0.8260 & -0.2058 & 0.5987 & 0.7132 & 1.0624 \\ -0.0921 & 2.0194 & -1.0211 & 2.0065 & -0.6298 & -0.2355 & -0.7416 \\ -1.7906 & 1.3778 & 0.2194 & -0.0018 & 0.5901 & -0.2583 & 0.0255 \\ -0.3944 & -0.2367 & -0.2827 & -2.2504 & -0.0613 & -2.0356 & 0.3409 \\ 0.5640 & 1.1120 & 1.9080 & -0.5875 & -0.3438 & -0.0549 & -0.6981 \\ 0.2651 & 0.0326 & 0.6555 & 0.2045 & 0.4572 & -0.5197 & 0.1553 \\ -0.6329 & 0.2689 & 0.0348 & 0.1515 & 0.5990 & -0.2046 & -1.6960 \\ 0.4201 & -1.4643 & 0.3174 & 0.4210 & 0.5310 & 1.4150 & -0.6530 \end{pmatrix}$$

$$\blacktriangleright\ \bar{\mathbf{b}}^{(u)} = [1.2197\ 0.3731\ 0.7235\ 0.0559\ -0.3939\ -0.3043\ -1.0943\ 0.4133]'$$

Fevrier

$$\blacktriangleright\ \mathbf{W}^{(x)} = \begin{pmatrix} -0.0151 & 0.0429 & 0.0785 & -0.0032 & 0.0009 & -0.0057 & 0.0134 & 0.0279 \\ -0.0011 & 0.0003 & -0.0099 & 0.0247 & 0.0520 & -0.0009 & 0.0004 & -0.0009 \\ 0.0014 & 0.0021 & -0.0000 & -0.0001 & 0.0057 & -0.0132 & -0.0273 & 0.0009 \\ -0.0004 & 0.0121 & -0.0276 & -0.0466 & -0.0018 & -0.0004 & 0.0118 & -0.0243 \\ -0.0525 & 0.0012 & -0.0006 & -0.0025 & 0.0060 & 0.0118 & -0.0003 & 0.0001 \end{pmatrix}$$

- $\mathbf{b}^{(x)} = [0.9689\ 0.4858\ 0.0430\ 2.0384\ 1.6678]'$
- $\mathbf{w}^{(x)} = [1.4343\ -2.4226\ -3.4319\ 0.9590\ -0.2607]$
- $\bar{b}^{(x)} = [-1.2228]$
- $\mathbf{w}^{(u)} = [-0.4191\ 0.4215\ -0.3848\ 0.0541\ -0.8612]'$
- $\mathbf{b}^{(u)} = [-0.2762\ 0.1145\ -1.0641\ -0.4356\ -0.1682]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} 1.9822 & -0.0714 & 1.1916 & -0.0850 & -0.0413 \\ -1.4648 & -1.5247 & -0.2304 & -0.6532 & -0.3674 \\ -0.6420 & 0.2052 & 0.0846 & 0.3598 & 0.6653 \\ -0.1837 & -0.9941 & 0.5904 & 0.6818 & -0.3057 \\ 0.0471 & 1.1834 & 1.9576 & -0.0412 & 1.1679 \\ 1.4208 & 2.2249 & -0.3076 & -0.1219 & -0.0171 \\ -0.8088 & -1.4037 & 0.3640 & 0.3029 & -0.1374 \\ 0.2841 & -0.5239 & -0.2660 & -0.3885 & 0.1662 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.4473\ 0.8260\ 1.3495\ -0.4163\ 0.2094\ 0.9764\ 1.2454\ -0.7434]'$

==========

Mars

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0402 & -0.0550 & -0.0166 & -0.0098 & -0.0128 & -0.0052 & -0.0221 & -0.0289 \\ -0.0085 & -0.0017 & -0.0031 & -0.0003 & 0.0105 & 0.0155 & 0.0047 & 0.0215 \\ 0.0300 & 0.0105 & 0.0241 & 0.0335 & 0.0103 & -0.0033 & -0.0044 & -0.0001 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [-0.0658\ -0.0610\ -0.1681]'$
- $\mathbf{w}^{(x)} = [2.0124\ -2.7812\ -0.8746]$
- $\bar{b}^{(x)} = [0.2602]$
- $\mathbf{w}^{(u)} = [-0.6883\ 0.8619\ -0.1973]'$
- $\mathbf{b}^{(u)} = [0.3298\ -0.5303\ 0.0622]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} -1.0969 & -0.2471 & -0.5394 \\ 0.0205 & 0.2700 & 0.6049 \\ 0.7286 & -0.0126 & 1.4217 \\ 0.3362 & 0.7614 & 0.0636 \\ -0.4274 & -0.8155 & -0.8918 \\ 0.0736 & 0.1783 & -0.0029 \\ 0.1578 & -0.1317 & 0.0805 \\ -0.1120 & -0.1990 & -0.1281 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.2698\ -0.0655\ -0.1318\ -0.0122\ 0.0824\ 0.1494\ 0.1702\ -0.0185]'$

==========

Avril

- $\mathbf{W}^{(x)} = \begin{pmatrix} 0.0497 & -0.0044 & -0.0577 & 0.0012 & 0.0291 & 0.0128 & 0.0011 & -0.0151 \\ 0.0009 & 0.0074 & 0.0358 & -0.0046 & -0.0425 & 0.0009 & 0.0226 & 0.0024 \\ 0.0001 & -0.0027 & -0.0000 & 0.0004 & -0.0217 & 0.0027 & 0.0270 & -0.0042 \\ -0.0132 & -0.0362 & 0.0035 & 0.0429 & -0.0010 & -0.0221 & -0.0377 & 0.0041 \\ 0.0445 & -0.0011 & -0.0214 & 0.0007 & -0.0005 & -0.0007 & -0.0009 & 0.0005 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [0.2575\ 0.7351\ 0.2025\ -0.2126\ 0.0128]'$

- $\mathbf{w}^{(x)} = [1.3456\ -0.1919\ -1.5423\ 0.0333\ 0.7684]$
- $\bar{b}^{(x)} = [0.0626]$
- $\mathbf{w}^{(u)} = [0.5507\ -0.3616\ -0.7955\ -0.4664\ 0.2181]'$
- $\mathbf{b}^{(u)} = [0.0607\ 0.2206\ -0.3700\ 0.2615\ 0.0014]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} 0.7809 & 0.0871 & 0.2307 & 0.0907 & -0.3418 \\ -0.5376 & -0.6212 & -0.0652 & -0.7843 & -0.1192 \\ -0.5057 & -0.0453 & 0.3327 & 0.5313 & 0.6132 \\ 0.1175 & -1.3953 & -0.3709 & -1.3381 & 0.0815 \\ 0.6984 & 1.1257 & 1.1603 & -0.1798 & -0.4494 \\ -0.2592 & -0.2529 & -0.1604 & 0.3392 & 0.4552 \\ 0.3124 & 0.0533 & 0.4437 & 0.1356 & 0.4226 \\ 0.0502 & -0.0033 & -0.1280 & -0.1790 & -0.0243 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [1.1553\ 0.3056\ 0.7441\ 0.1269\ -0.5638\ -0.8743\ -0.8751\ -0.0701]'$

==========

Mai

- $\mathbf{W}^{(x)} = \begin{pmatrix} 0.0113 & -0.0409 & 0.0502 & 0.0406 & 0.0046 & -0.0316 & -0.0188 & 0.0497 \\ 0.0013 & -0.0073 & 0.0109 & 0.0070 & 0.0003 & -0.0053 & -0.0037 & 0.0090 \\ 0.0080 & -0.0282 & 0.0356 & 0.0284 & 0.0022 & -0.0191 & -0.0146 & 0.0347 \\ -0.0001 & 0.0001 & -0.0046 & -0.0008 & -0.0005 & -0.0005 & 0.0006 & 0.0026 \\ -0.0047 & 0.0129 & -0.0169 & -0.0128 & -0.0019 & 0.0092 & 0.0060 & -0.0154 \\ -0.0091 & 0.0341 & -0.0413 & -0.0335 & -0.0050 & 0.0231 & 0.0153 & -0.0404 \\ -0.0082 & 0.0309 & -0.0385 & -0.0315 & -0.0042 & 0.0218 & 0.0142 & -0.0372 \\ -0.0005 & 0.0025 & -0.0006 & -0.0022 & 0.0033 & 0.0018 & 0.0012 & -0.0005 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [0.0601\ -0.0261\ -0.0267\ 0.0155\ 0.0579\ 0.0793\ 0.0940\ -0.0486]'$
- $\mathbf{w}^{(x)} = [-0.2460\ 0.9326\ -1.1516\ -0.9337\ -0.1263\ 0.6437\ 0.4295\ -1.1220]$
- $\bar{b}^{(x)} = [0.5873]$
- $\mathbf{w}^{(u)} = [0.4566\ -0.2490\ -0.7194\ -0.2698\ 0.2089\ -0.4104\ 0.5580\ -0.4232]'$
- $\mathbf{b}^{(u)} = [-0.1891\ 0.1428\ 0.1851\ 0.0189\ 0.5950\ 0.1324\ -0.1270\ 0.0009]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} -0.5239 & -0.2315 & -0.3625 & 0.0415 & 0.1410 & 0.5519 & 0.5953 & 0.0016 \\ 0.5002 & 0.3622 & 0.1627 & -0.1619 & -0.3304 & -0.5780 & -0.4938 & -0.0086 \\ 0.6622 & 0.0070 & 0.5157 & 0.0733 & -0.3215 & -0.5074 & -0.4350 & -0.0894 \\ 0.2955 & -0.1285 & 0.4254 & 0.2038 & -0.1630 & -0.0728 & -0.2216 & 0.2160 \\ -0.0215 & -0.0569 & -0.0480 & -0.0043 & 0.0514 & 0.1080 & -0.0039 & -0.0283 \\ 0.5367 & 0.1921 & 0.3079 & -0.0988 & -0.2419 & -0.5023 & -0.4811 & 0.0382 \\ -0.7429 & -0.2605 & -0.3830 & 0.1103 & 0.3275 & 0.6738 & 0.6151 & -0.0053 \\ 0.2103 & -0.2665 & 0.2617 & 0.2159 & 0.1576 & 0.1669 & 0.0531 & -0.1561 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [0.6905\ 0.0779\ 0.5382\ 0.0672\ -0.2656\ -0.5328\ -0.4811\ -0.0122]'$

==========

Juin

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0305 & 0.0143 & -0.0151 & 0.0065 & -0.0264 & 0.0108 & -0.0026 & -0.0003 \\ 0.0180 & -0.0067 & 0.0406 & -0.0166 & 0.0469 & -0.0195 & 0.0210 & -0.0078 \end{pmatrix}$

- $\mathbf{b}^{(x)} = [-0.3458\ 0.4076]'$
- $\mathbf{w}^{(x)} = [3.8223\ -1.6094]$
- $\bar{b}^{(x)} = [1.1871]$
- $\mathbf{w}^{(u)} = [-0.0101\ 1.0147]'$
- $\mathbf{b}^{(u)} = [0.0779\ 0.5597]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} 0.3798 & -0.4671 \\ -0.2237 & -0.0776 \\ 0.3827 & -0.6912 \\ -1.2358 & -1.5960 \\ -1.2386 & -0.6104 \\ -1.0878 & -0.1053 \\ 0.7047 & 1.5301 \\ 1.8426 & 0.7923 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.1609\ -0.0554\ -0.1507\ -0.0085\ 0.0771\ 0.2963\ 0.3917\ 0.2592]'$

==========

Juillet

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0088 & -0.0003 & -0.0216 & 0.0390 & -0.0023 & 0.0008 & -0.0050 & 0.0006 \\ -0.0223 & 0.0337 & -0.0002 & 0.0005 & -0.0160 & -0.0004 & -0.0423 & 0.0793 \\ -0.0006 & 0.0013 & -0.0064 & -0.0004 & -0.0363 & 0.0699 & -0.0005 & 0.0011 \\ -0.0122 & -0.0003 & -0.0325 & 0.0638 & -0.0019 & 0.0010 & -0.0083 & -0.0002 \\ -0.0185 & 0.0357 & -0.0037 & 0.0006 & -0.0081 & -0.0011 & -0.0239 & 0.0444 \\ 0.0008 & 0.0008 & -0.0048 & -0.0013 & -0.0145 & 0.0272 & -0.0002 & 0.0005 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [-0.2905\ -2.7538\ 0.2845\ -0.0264\ 2.9293\ -2.7269]'$
- $\mathbf{w}^{(x)} = [-0.7537\ -1.3273\ -1.9176\ 3.6977\ -2.5155\ 3.7827]$
- $\bar{b}^{(x)} = [4.6420]$
- $\mathbf{w}^{(u)} = [-0.9316\ -0.6373\ 0.1270\ -1.6031\ 0.9271\ -0.3219]'$
- $\mathbf{b}^{(u)} = [-3.9030\ 0.0330\ 0.7850\ -2.1507\ 0.2219\ -1.0901]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} 0.3299 & 0.5851 & 0.1148 & 0.5832 & 0.1356 & 0.4684 \\ 0.0783 & 0.0675 & 0.5434 & 0.2285 & -0.0512 & -0.1882 \\ -0.0093 & -0.0689 & -0.3042 & 0.4227 & -0.4910 & 0.2168 \\ -0.0305 & -0.2579 & -0.4833 & -0.7522 & -0.2652 & 0.0011 \\ -1.2278 & -0.9433 & -0.0589 & -0.1040 & -0.0887 & 0.3620 \\ 0.6229 & -1.0285 & -0.0665 & -0.2019 & 0.6808 & 0.6935 \\ 0.6684 & 0.8529 & 0.9422 & -0.1734 & 0.3444 & -0.0650 \\ -0.6415 & -0.0107 & -0.0992 & -0.1951 & -0.0633 & 0.0256 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.3499\ -0.5000\ -0.0594\ 1.0181\ 0.5682\ 1.4411\ 1.1934\ -0.9631]'$

==========

Août

- $\mathbf{W}^{(x)} = \begin{pmatrix} 0.0062 & 0.0032 & -0.0174 & -0.0172 & 0.0077 & 0.0037 & -0.0201 & -0.0184 \\ 0.0119 & 0.0060 & -0.0366 & -0.0342 & 0.0128 & 0.0066 & -0.0369 & -0.0336 \\ 0.0086 & 0.0053 & -0.0293 & -0.0257 & 0.0075 & 0.0040 & -0.0226 & -0.0207 \\ 0.0093 & 0.0048 & -0.0284 & -0.0247 & 0.0077 & 0.0037 & -0.0239 & -0.0203 \end{pmatrix}$

- $\mathbf{b}^{(x)} = [0.0858\ 0.0993\ -0.0563\ -0.0402]'$
- $\mathbf{w}^{(x)} = [0.8055\ 0.4316\ -2.4472\ -2.2068]$
- $\bar{b}^{(x)} = [0.3652]$
- $\mathbf{w}^{(u)} = [1.5574\ 0.8940\ 1.3267\ 0.2963]'$
- $\mathbf{b}^{(u)} = [-0.7526\ -0.9770\ -0.8773\ 0.0503]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} 0.3913 & 0.3136 & 0.3379 & 0.4771 \\ 0.3670 & 0.1817 & 0.1149 & 0.3358 \\ -0.1968 & -0.0082 & 0.0953 & 0.0365 \\ 0.0543 & 0.2253 & 0.2814 & 0.0531 \\ 0.1890 & 0.2003 & 0.4324 & 0.3874 \\ 0.3459 & 0.2576 & 0.3568 & 0.2554 \\ 0.0564 & 0.0693 & 0.1595 & 0.1467 \\ 0.0761 & 0.0988 & 0.1712 & 0.0343 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.2208\ -0.1338\ -0.1438\ -0.2032\ -0.1403\ -0.0243\ -0.0071\ -0.1154]'$

==========

Septembre

- $\mathbf{W}^{(x)} = [-0.0103\ 0.0585\ 0.0371\ 0.0187\ 0.0028\ 0.0256\ -0.0081\ -0.0103]$
- $\mathbf{b}^{(x)} = [0.2010]'$
- $\mathbf{w}^{(x)} = [2.8366]$
- $\bar{b}^{(x)} = [0.1470]$
- $\mathbf{w}^{(u)} = [1.0884]'$
- $\mathbf{b}^{(u)} = [-0.6072]'$
- $\mathbf{W}^{(u)} = [-0.5521\ 3.2966\ 2.1390\ 1.0097\ 0.2669\ 1.5068\ -0.4863\ -0.5246]'$
- $\bar{\mathbf{b}}^{(u)} = [0.0890\ -0.5393\ -0.3510\ -0.1645\ -0.0420\ -0.2467\ 0.0794\ 0.0849]'$

==========

Octobre

- $\mathbf{W}^{(x)} = \begin{pmatrix} -0.0001 & 0.0044 & -0.0453 & -0.0731 & -0.0622 & -0.0111 & -0.0378 & 0.0013 \\ 0.0018 & -0.0130 & -0.0133 & -0.0171 & -0.0051 & -0.0087 & -0.0010 & 0.0061 \\ -0.0334 & -0.0536 & -0.0518 & -0.0081 & -0.0299 & 0.0002 & -0.0010 & -0.0031 \\ -0.0099 & -0.0061 & 0.0002 & -0.0036 & -0.0001 & -0.0042 & 0.0252 & 0.0376 \\ 0.0332 & 0.0053 & 0.0195 & 0.0014 & -0.0048 & 0.0294 & 0.0431 & 0.0360 \\ 0.0092 & 0.0262 & -0.0004 & -0.0033 & 0.0426 & 0.0510 & 0.0483 & 0.0074 \\ 0.0286 & -0.0012 & -0.0014 & 0.0199 & 0.0241 & 0.0236 & 0.0034 & 0.0120 \end{pmatrix}$
- $\mathbf{b}^{(x)} = [0.9274\ 1.9909\ -0.0768\ 0.2984\ 0.5130\ 1.2523\ -0.1598]'$
- $\mathbf{w}^{(x)} = [0.0047\ 1.6264\ -1.0848\ -1.6316\ -1.7384\ -0.9063\ -0.8021]$
- $\bar{b}^{(x)} = [0.9854]$
- $\mathbf{w}^{(u)} = [-0.3555\ -1.1244\ -0.4832\ -0.6061\ -0.4429\ 2.0471\ -0.2271]'$
- $\mathbf{b}^{(u)} = [0.4674\ 0.8752\ 0.1905\ -0.3586\ -0.3532\ -0.3839\ -0.5296]'$

$$\blacktriangleright\ \mathbf{W}^{(u)} = \begin{pmatrix} -0.5405 & -0.6218 & -0.4394 & -1.0059 & -0.0690 & 0.9312 & 0.5813 \\ 0.1962 & -0.9911 & -0.1604 & -0.4866 & 0.2004 & 0.5161 & 0.3582 \\ 0.8659 & -0.1664 & -0.1325 & -0.4250 & -0.2989 & -0.7992 & -0.6746 \\ 1.0348 & 0.7630 & 0.5871 & -0.4495 & -0.5948 & 0.1642 & -0.3975 \\ -0.3978 & 0.5473 & 0.7799 & 0.4143 & 0.4150 & -0.0482 & 0.0396 \\ 0.1271 & -0.0548 & 0.4817 & 0.3666 & 0.5876 & -0.2936 & -0.5602 \\ 0.3073 & -0.5377 & -0.5746 & 1.0169 & 1.3839 & 0.3020 & -0.1426 \\ -0.2014 & 0.5382 & 0.0124 & -0.2648 & 0.3316 & 0.8427 & 0.8013 \end{pmatrix}$$

$\blacktriangleright\ \bar{\mathbf{b}}^{(u)} = [0.2722\ -0.0149\ 0.3020\ 0.3217\ -0.1304\ -0.0369\ 0.0596\ 0.9140]'$

==========

Novembre

$$\blacktriangleright\ \mathbf{W}^{(x)} = \begin{pmatrix} 0.0320 & -0.0477 & -0.0379 & 0.0043 & 0.0017 & 0.0069 & 0.0204 & 0.0151 \\ -0.0248 & -0.0228 & 0.0000 & 0.0014 & 0.0005 & 0.0021 & 0.0267 & -0.0398 \\ -0.0380 & -0.0005 & -0.0052 & 0.0015 & 0.0133 & -0.0100 & -0.0004 & 0.0076 \\ -0.0014 & -0.0009 & -0.0003 & -0.0021 & -0.0170 & 0.0370 & 0.0354 & -0.0017 \\ 0.0044 & -0.0020 & -0.0168 & -0.0224 & 0.0453 & 0.0332 & -0.0048 & -0.0030 \\ -0.0010 & -0.0233 & -0.0214 & 0.0248 & 0.0257 & -0.0044 & 0.0026 & 0.0031 \\ -0.0123 & -0.0062 & 0.0097 & 0.0129 & 0.0016 & 0.0035 & 0.0021 & 0.0027 \end{pmatrix}$$

$\blacktriangleright\ \mathbf{b}^{(x)} = [0.5401\ 0.2282\ -0.2133\ 0.4673\ 0.6097\ 0.5264\ -0.3406]'$

$\blacktriangleright\ \mathbf{w}^{(x)} = [-0.9662\ 1.2914\ 1.0005\ -0.0604\ 0.1426\ -0.0248\ -0.6575]$

$\blacktriangleright\ \bar{b}^{(x)} = [-0.5646]$

$\blacktriangleright\ \mathbf{w}^{(u)} = [0.1513\ 0.4736\ 0.2911\ 0.0810\ -0.7305\ 0.9185\ 1.3783]'$

$\blacktriangleright\ \mathbf{b}^{(u)} = [-0.1387\ 0.1744\ 0.2455\ 0.3125\ -1.7125\ 0.5365\ 1.2396]'$

$$\blacktriangleright\ \mathbf{W}^{(u)} = \begin{pmatrix} -0.2023 & -0.0635 & -0.0824 & 0.0629 & 0.1137 & 0.3235 & 0.0368 \\ 0.3963 & -0.5692 & -0.2320 & -0.0567 & 0.1051 & 0.2012 & 0.7442 \\ 0.0605 & 0.0287 & 0.8273 & -2.3701 & 2.9439 & 0.4326 & -1.2104 \\ 2.4892 & -2.3135 & -0.8169 & 0.4187 & -1.1178 & 1.4582 & 0.0808 \\ -0.4209 & 0.9687 & -1.0720 & -0.6268 & 0.7857 & -1.4931 & 2.0504 \\ 0.1850 & -0.8033 & 1.0784 & -1.3477 & -0.5699 & -0.9340 & -0.0714 \\ -1.1482 & 0.1034 & 0.6729 & 0.2840 & 0.8994 & 0.5949 & -0.8794 \\ -0.1359 & -0.8503 & 0.0083 & 0.7751 & 0.4040 & 0.7037 & -0.0032 \end{pmatrix}$$

$\blacktriangleright\ \bar{\mathbf{b}}^{(u)} = [0.5021\ -0.8935\ 1.2888\ 0.1567\ -0.5824\ 0.6797\ -0.8244\ -0.1159]'$

==========

Décembre

$$\blacktriangleright\ \mathbf{W}^{(x)} = \begin{pmatrix} -0.0186 & -0.0466 & -0.0206 & 0.0358 & -0.0231 & -0.0404 & -0.0208 & 0.0289 \\ -0.0293 & -0.0573 & -0.0489 & 0.0492 & -0.0019 & 0.0028 & -0.0018 & -0.0060 \\ 0.0225 & 0.0376 & 0.0185 & -0.0295 & 0.0301 & 0.0666 & 0.0368 & -0.0413 \\ 0.0260 & 0.0504 & 0.0324 & -0.0354 & 0.0150 & 0.0385 & 0.0214 & -0.0278 \end{pmatrix}$$

$\blacktriangleright\ \mathbf{b}^{(x)} = [0.0177\ 0.4699\ -0.7156\ -0.7929]'$

$\blacktriangleright\ \mathbf{w}^{(x)} = [-1.5260\ -3.9832\ -3.8989\ 4.2589]$

$\blacktriangleright\ \bar{b}^{(x)} = [1.4930]$

- $\mathbf{w}^{(u)} = [5.0550\ 0.2450\ 4.0468\ -4.2544]'$
- $\mathbf{b}^{(u)} = [2.6885\ 1.6217\ 2.3640\ -2.5713]'$
- $\mathbf{W}^{(u)} = \begin{pmatrix} -0.5025 & -0.4782 & 0.1780 & 0.1660 \\ 0.2969 & -0.1498 & 0.0178 & 0.7766 \\ 0.1024 & -0.9996 & -0.0769 & 0.9009 \\ -0.5185 & -0.3754 & 1.5781 & 1.3225 \\ 1.0598 & 0.0471 & 0.4786 & 0.0111 \\ -0.6873 & -1.1399 & 0.4832 & 0.7628 \\ 0.4451 & -0.4344 & 0.4396 & 0.1501 \\ -0.3202 & -1.0152 & 0.5904 & 1.3555 \end{pmatrix}$
- $\bar{\mathbf{b}}^{(u)} = [-0.1493\ 0.7500\ 0.2268\ -0.7450\ 0.4923\ 0.1284\ -1.4166\ -0.8412]'$

Conclusion générale

L'inefficacité de la modélisation linéaire nous est apparue à travers quelques exemples. La modélisation des séries temporelles par les méthodes non linéaires est venue combler les lacunes de la modélisation linéaires. Nous avons étudié une méthode d'analyse non linéaire qui est la méthode réseaux de neurones, et nous avons choisi deux applications différentes, pour cerner les deux buts essentiels de l'analyse des séries temporelles : la modélisation et la prédiction.

Pour la prédiction, nous avons choisi comme application une série mono-variable, et comme configuration, un réseau FIR de dimension ($1\times12\times12\times1$), de retards ($25 : 5 : 5$), ce sont les données d'un laser dans un état chaotique, une série temporelle qui a fait l'objet de plusieurs applications afin de valider des modèles appliquées. Les résultats sont satisfaisants et concordent avec ceux déjà obtenus.

Pour la modélisation, nous avons choisi une série multi-variable, c'est les données de la fraction d'insolation de l'Algérie, qui a été prise comme dépendante de la latitude et de la longitude; cette étude est la première du genre. La méthode choisie pour cette tâche est l'analyse de la composante principale par réseau de neurones, une méthode qui largement appliquée dans la modélisation des données météorologiques et océanographiques. Pour chaque mois, nous avons trouvé un modèle correspondant. Les résultats sont satisfaisants, vu les erreurs trouvées (entre 0.1% et 0.8 % pour le mode 1 et moins pour le mode 2). Toutefois, toutes les ressources offertes par la méthode n'ont pas été utilisées, car nous avons arrêté notre travail à la modélisation en nous contentant d'extraire uniquement deux modes.

Ce qui serait intéressant, mais demande plus de travail et de calcul, est d'aller au-delà de deux modes, ce qui donnera des résultats plus précis et des modèles plus près de la réalité. Une fois les modèles du genre trouvés, nous pourrions concevoir une boite noire sous forme de logiciel indépendant de tout espace de travail, qui aurait comme entrées, la longitude d'une région et la latitude d'un lieu de cette région, et comme sortie la fraction d'insolation correspondante.

Annexe A: La complète dérivation de l'algorithme de rétropropagation temporelle

Nous donnons ici une complète dérivation de l'algorithme de rétropropagation temporelle. Nous voulons minimiser la fonction coût $C = \sum_k e^2(k)$ (c'est-à-dire la somme des erreurs quadratiques instantanées). Le gradient de la fonction coût avec respect du filtre synaptique est expliqué en utilisant la règle de chaîne :

$$\frac{\partial C}{\partial \mathbf{w}_{ij}^l} = \sum_k \frac{\partial C}{\partial s_j^{l+1}(k)} \cdot \frac{\partial s_j^{l+1}(k)}{\partial \mathbf{w}_{ij}^l} \tag{1}$$

où

$$s_j^{l+1}(k) = \sum_i s_{ij}^{l+1}(k) = \sum_i \mathbf{w}_{i,j}^l \cdot \mathbf{x}_i^l(k). \tag{2}$$

spécifie l'entrée du neurone j dans la couche l en temps k. Noter que cette expansion est différente de l'approche traditionnelle qui écrit le gradient total comme une somme des gradients instantanés: $\partial C/\partial s_j^{l+1}(k) \cdot \partial s_j^{l+1}(k)/\partial \mathbf{w}_{ij}^l \neq \partial e^2(k)/\partial \mathbf{w}_{ij}^l$. C'est seulement la somme sur tout les k qui est équivalente.

De l'équation 1, un algorithme stochastique est formé :

$$\mathbf{w}_{ij}^l(k+1) = \mathbf{w}_{ij}^l(k) - \eta \frac{\partial C}{\partial s_j^{l+1}(k)} \cdot \frac{\partial s_j^{l+1}(k)}{\partial \mathbf{w}_{ij}^l} \tag{3}$$

De l'équation 2, on conclut immédiatement que $s_j^{l+1}(k)/\mathbf{w}_{ij}^l = \mathbf{x}_i^l(k)$ pour toutes les couches du réseau. On définit $\partial C/\partial s_j^l(k) \equiv \delta_j^l(k)$ qui nous permet d'écrire l'équation 3 dans une forme notationnelle plus familière :

$$\mathbf{w}_{ij}^l(k+1) = \mathbf{w}_{ij}^l - \eta \delta_j^{l+1}(k).\mathbf{x}_i^l(k). \tag{4}$$

Pour montrer cette influence sur toutes les couches du réseau, une formule explicite pour $\delta_j^l(k)$ doit être trouvée. En commençant par la couche de sortie, on a simplement :

$$\delta_j^L(k) \equiv \frac{\partial C}{\partial s_j^L} = \frac{\partial e^2(k)}{\partial s_j^L} = -2e_j(k)f'(s_j^L(k)). \tag{5}$$

où $e_j(k)$ est l'erreur dans un nœud de sortie. Pour une couche cachée, on utilise encore la règle de la chaîne, en expansant sur tous les temps et sur les N_{l+1} entrées $s^{l+1}(k)$ dans la couche suivante :

$$\delta_j^l(k) \equiv \frac{\partial C}{\partial s_j^l}$$

$$= \sum_{m=1}^{N_{l+1}} \sum_t \frac{\partial C}{\partial s_m^{l+1}(t)} \frac{\partial s_m^{l+1}(t)}{\partial s_j^l(k)}$$

$$= \sum_{m=1}^{N_{l+1}} \sum_t \delta_m^{l+1}(t) \frac{\partial s_m^{l+1}(t)}{\partial s_j^l(k)}$$

$$= f'(s_j^l(k)) \sum_{m=1}^{N_{l+1}} \sum_t \delta_m^{l+1}(t) \frac{\partial s_{jm}^{l+1}(t)}{\partial x_j^l(k)}. \quad (6)$$

Mais rappelons que

$$s_{jm}^{l+1}(t) = \sum_{k'=0}^{N^l} w_{jm}^l(k') x_j^l(t - k'). \quad (7)$$

Donc

$$\frac{\partial s_{jm}^{l+1}(t)}{\partial x_j^l(k)} = \begin{cases} w_{jm}^l(t-k) & \text{pour } 0 \leq t-k \leq N^l \\ 0 & \text{si non} \end{cases} \quad (8)$$

d'où de nouveaux rendements

$$\delta_j^l(k) = f'(s_j^l(k)). \sum_{m=1}^{N_{l+1}} \sum_{t=k}^{N^l+k} \delta_m^{l+1}(t) w_{jm}^l(t-k).$$

$$= f'(s_j^l(k)). \sum_{m=1}^{N_{l+1}} \sum_{t=0}^{N^l} \delta_m^{l+1}(k+n) w_{jm}^l(n).$$

$$= f'(s_j^l(k)). \sum_{m=1}^{N_{l+1}} \boldsymbol{\delta}_m^{l+1}(k) \cdot \mathbf{w}_{jm}^l, \quad (9)$$

où nous avons déjà défini

$$\boldsymbol{\delta}_m^l(k) = [\delta_m^l(k), \delta_m^l(k+1), ..., \delta_m^l(k+N^{l-1})]. \quad (10)$$

En résumé, l'algorithme complet adapté, peut être exprimé comme suit :

$$\mathbf{w}_{ij}^l(k+1) = \mathbf{w}_{ij}^l(k) - \eta \delta_j^{l+1}(k) \cdot \mathbf{x}_i^l(k). \quad (11)$$

$$\delta_j^l(k) = \begin{cases} -2e_j(k)f'(s_j^L(k)) & l = L \\ f'(s_j^l(k)) \cdot \sum_{m=1}^{N_{l+1}} \delta_m^{l+1}(k) \cdot \mathbf{w}_{jm}^l & 1 \leq l \leq L-1 \end{cases} \quad (12)$$

La condition de causalité : une inspection attentive des équations précédentes relève que le calcul de $\delta_j^l(k)$, est non causal. La source de ce filtrage non causal peut être vue en considérant la définition de $\delta_j^l(k) = \partial C / \partial s_j^l(k)$. Quand on prend un temps pour la sortie de n'importe quel neurone interne pour compléter la propagation à travers le réseau, le changement dans l'erreur totale, dû au changement dans l'état interne, est une fonction des valeurs futures dans le réseau. Comme le réseau est un FIR seulement un nombre fini des valeurs futures doivent être considérées, et un simple ré-indexation nous permet d'écrire l'algorithme sous une forme causale :

$$\mathbf{w}_{ij}^{L-1-n}(k+1) = \mathbf{w}_{ij}^{L-1-n}(k) - \eta \delta_j^{L-n}(k - nN) \cdot \mathbf{x}_i^{L-1-n}(k - nN) \quad (13)$$

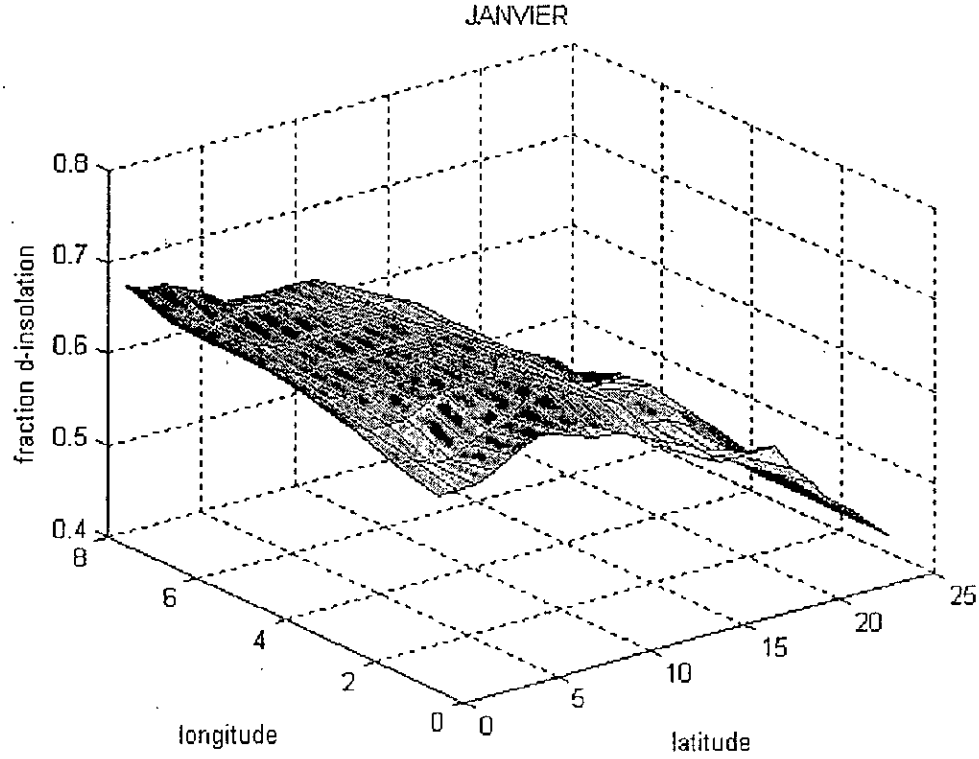
$$\delta_j^{L-n}(k-nN) = \begin{cases} -2e_j(k)f'(s_j^L(k)) & n = 0 \\ f'(s_j^{L-n}(k-nT)) \cdot \sum_{m=1}^{N_{l+1}} \delta_m^{L+1-n}(k-nN) \cdot \mathbf{w}_{jm}^{L-n} & 1 \leq n \leq L-1. \end{cases} \quad (14)$$

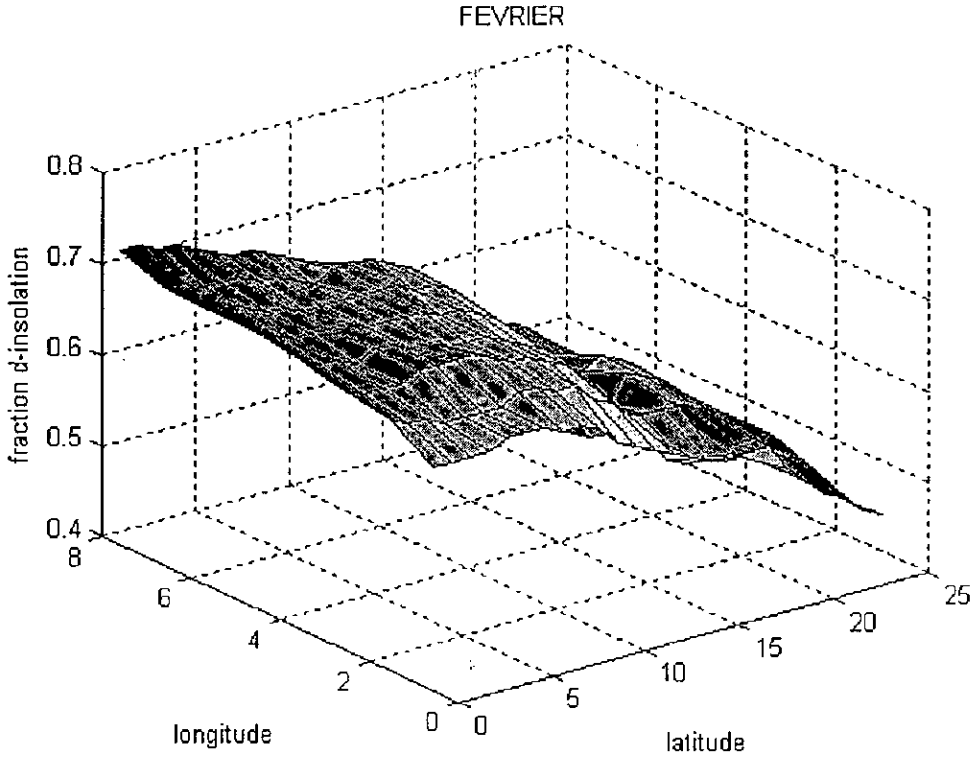
Esthétiquement plus satisfaisantes que les premières équations, elles diffèrent seulement en termes de changement d'indices. Ces équations sont implémentées en propageant les termes delta continuellement, en arrière, sans retard. Toutefois, par définition, cela force les valeurs internes des deltas à être changées dans le temps. Donc, on doit mémoriser les états $\mathbf{x}(k)$ appropriés pour former les termes adéquats pour la propagation. Un emmagasinement additionnel des retards est nécessaire uniquement pour les états $\mathbf{x}(k)$. La propagation en arrière des termes delta n'exige pas des retards additionnels et reste symétrique pour la propagation en avant. L'effet de cela est de retarder le gradient actuel mis à jour par quelques pas de temps. Cela peut résulter en des taux de convergences légèrement différents et un mauvais ajustement comme dans le cas linéaire analogue à l'algorithme *Delayed LMS*.

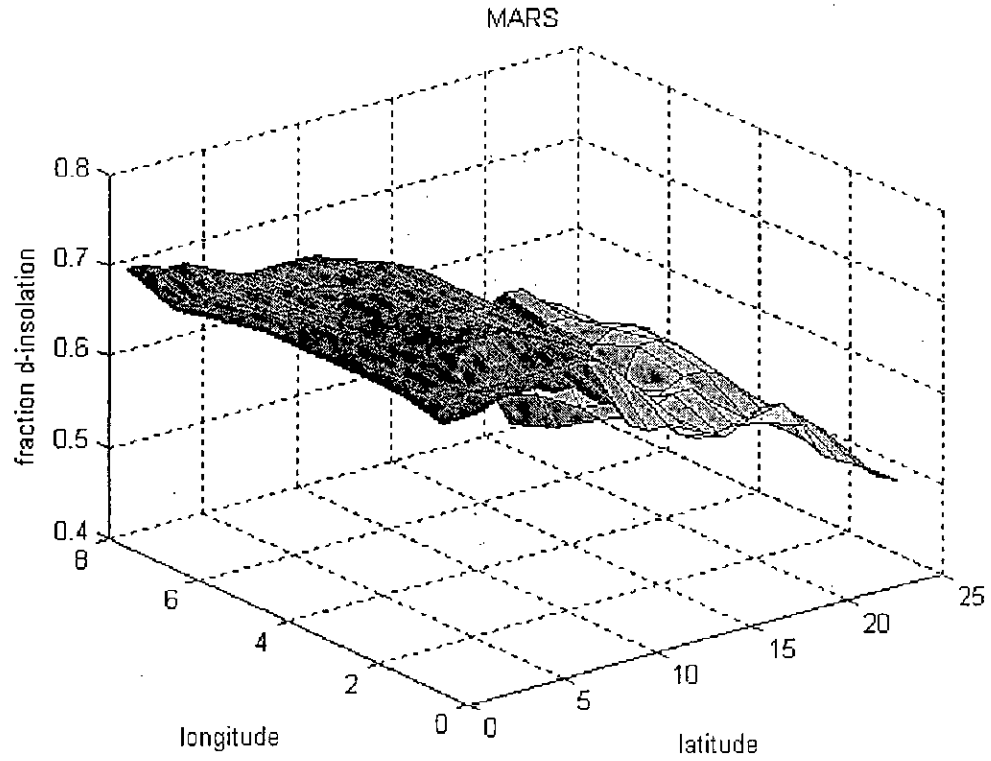
Par souci de simplicité, nous avons supposé que l'ordre de chaque filtre synaptique, N, est le même dans chaque couche. Cela est bien sûr non nécessaire. Dans le cas général, soit N_{ij}^l l'ordre du filtre synaptique connectant le neurone i de la couche l au neurone j de la couche suivante. Donc, dans les équations 13 et 14, nous remplaçons nN par $\sum_{l=L-n}^{L-1} \max_{ij} \{N_{ij}^l\}$. La règle

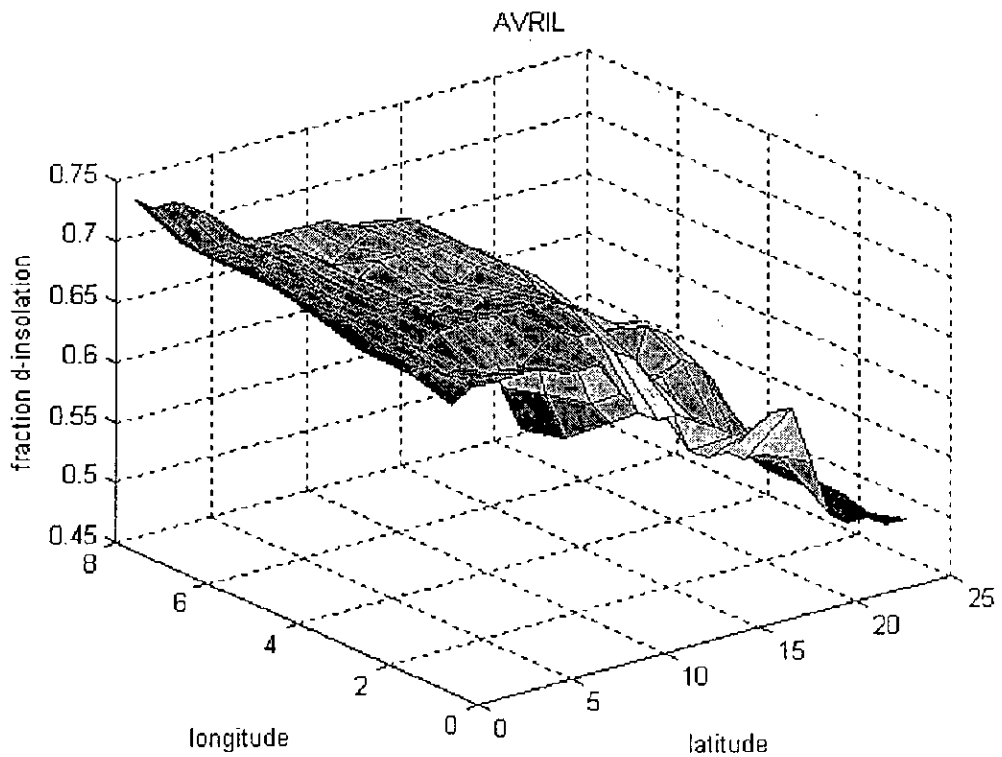
de base est que le changement de temps pour le delta associé à un neurone donné doit être produit égal au nombre total des retards le long du plus long chemin menant à la sortie du réseau.

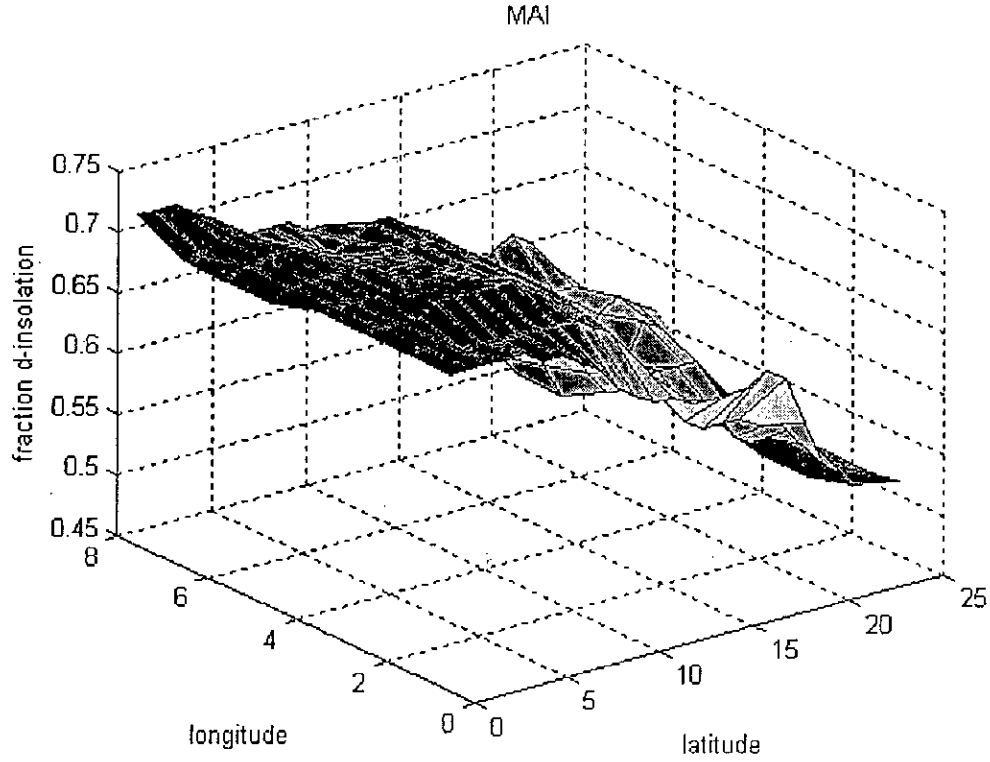
Annexe B : Figures illustratives des données de la fraction d'insolation utilisées

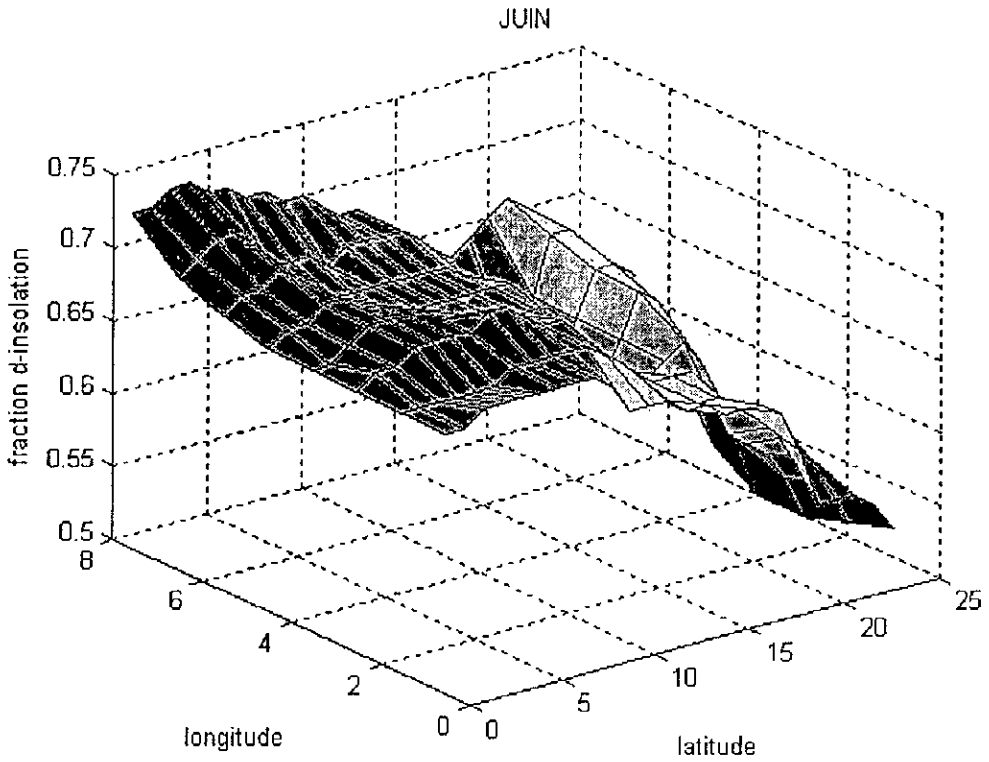


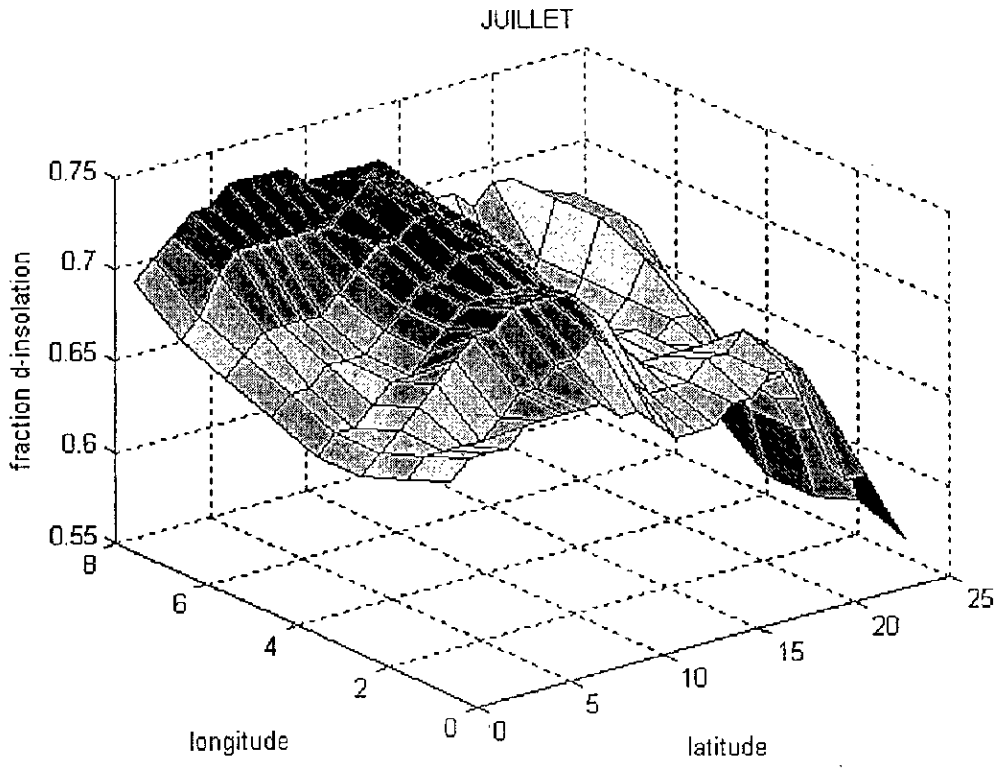


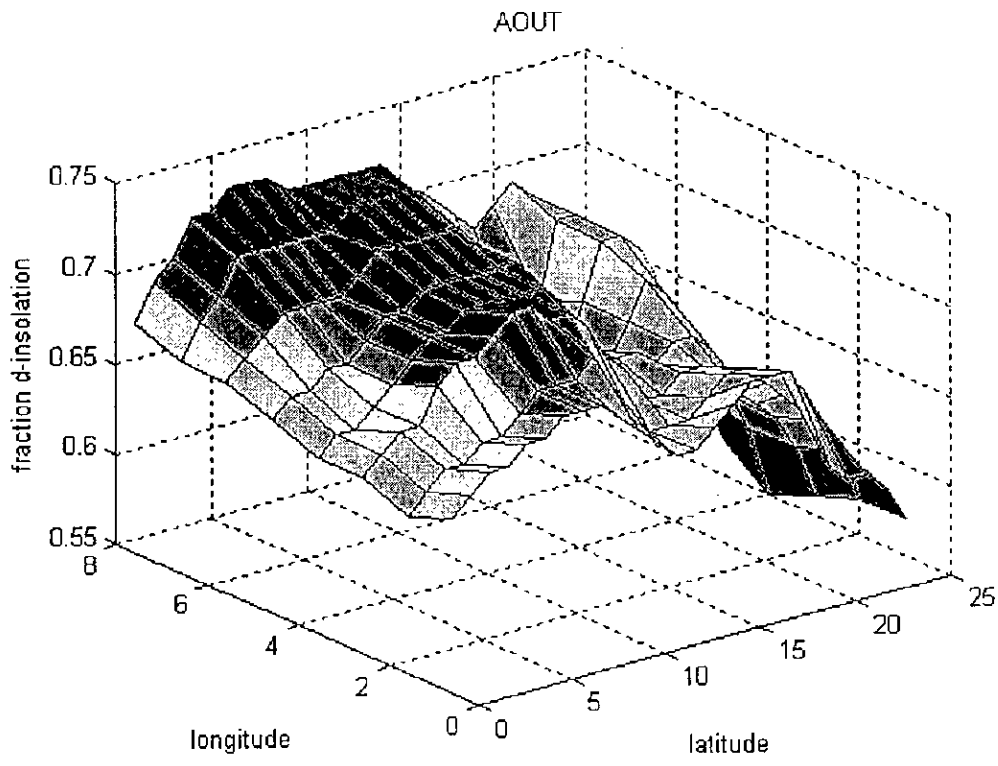

MARS
fraction d-insolation
0.8
0.7
0.6
0.5
0.4
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude

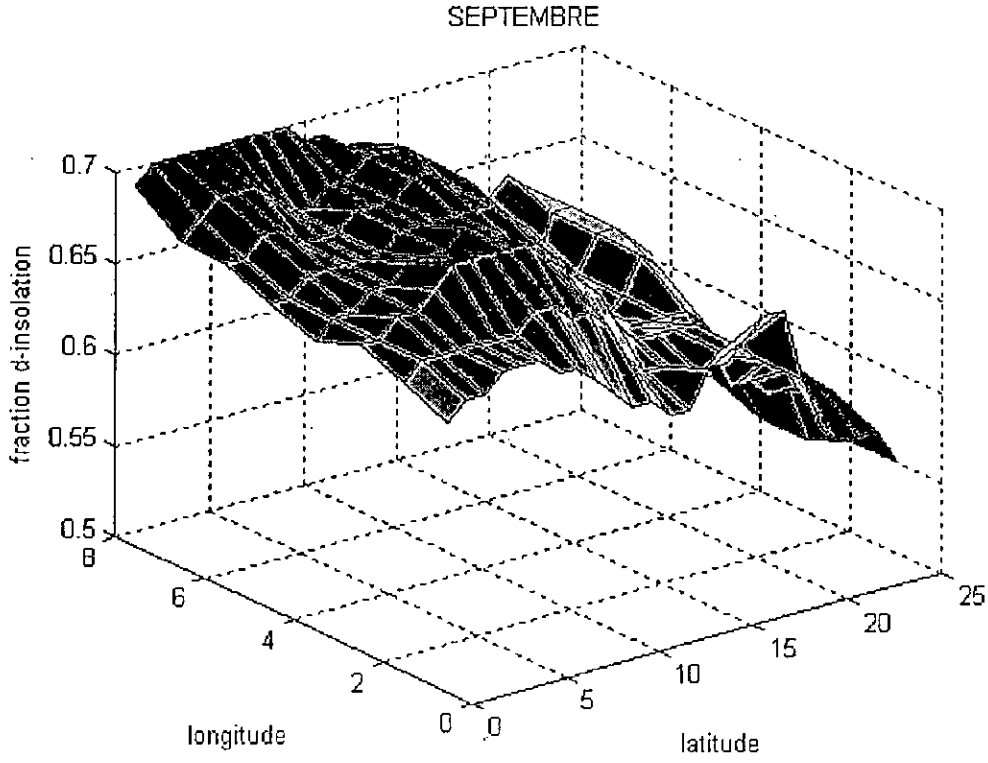

AVRIL
fraction d-insolation
0.75
0.7
0.65
0.6
0.55
0.5
0.45
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude

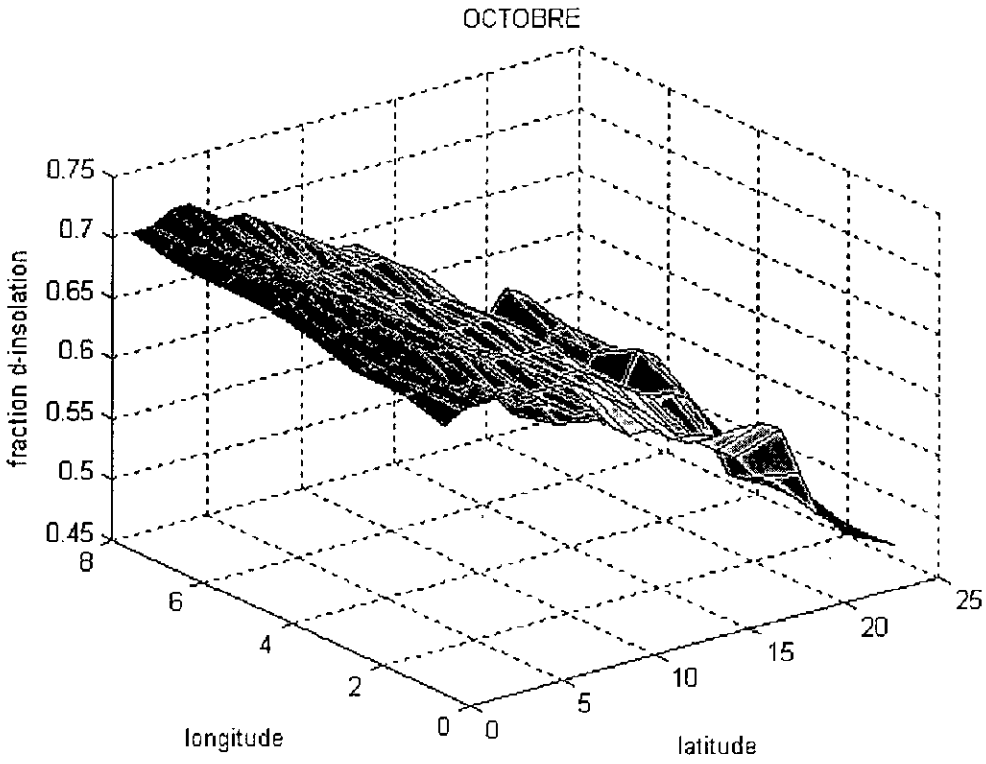

MAI
fraction d-insolation
0.75
0.7
0.65
0.6
0.55
0.5
0.45
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude

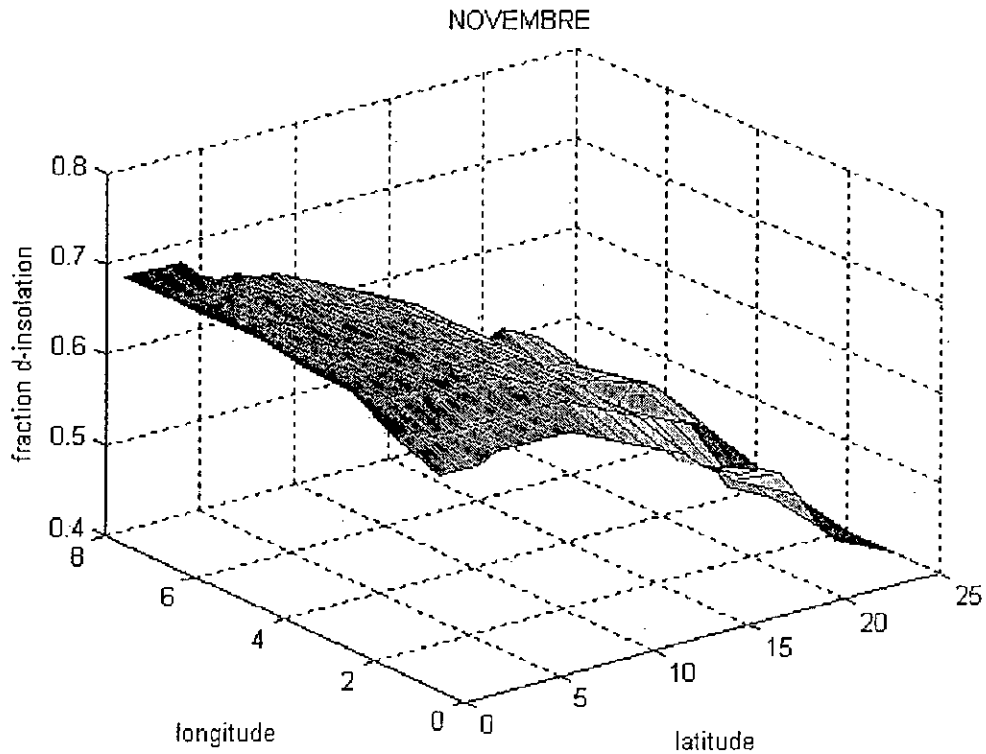

JUIN
fraction d-insolation
0.75
0.7
0.65
0.6
0.55
0.5
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude

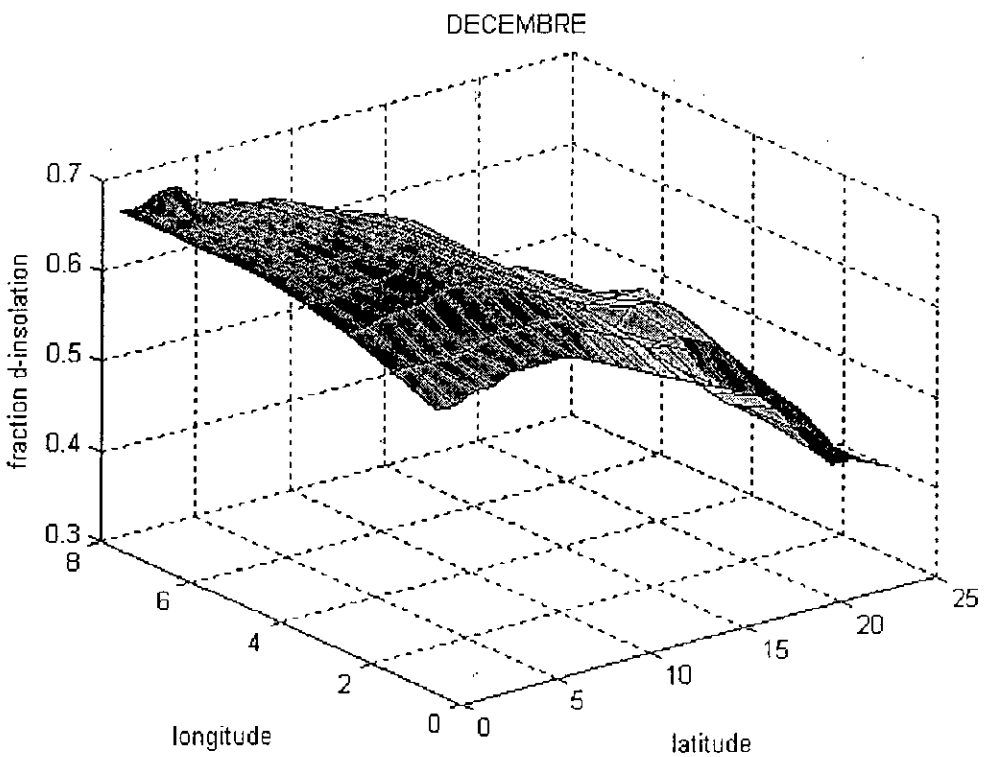

JUILLET
fraction d-insolation
0.75
0.7
0.65
0.6
0.55
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude


AOUT
fraction d-insolation
0.75
0.7
0.65
0.6
0.55
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude


SEPTEMBRE
fraction d-insolation
0.7
0.65
0.6
0.55
0.5
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude


OCTOBRE
fraction d-insolation
0.75
0.7
0.65
0.6
0.55
0.5
0.45
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude


NOVEMBRE
fraction d-insolation
0.8
0.7
0.6
0.5
0.4
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude


DECEMBRE
fraction d-insolation
0.7
0.6
0.5
0.4
0.3
8
6
4
2
0
longitude
0
5
10
15
20
25
latitude

Annexe C: Algorithme du modèle NLPCA

Dans la figure 4 du chapitre 4, la fonction de transfert f_1 part de $\mathbf{x}$, vecteur d'entrée colonne de longueur l, à la première couche cachée (la couche de codage), représentée par $\mathbf{h}^{(x)}$, un vecteur colonne de longueur m, dont les éléments:

$$h_k^{(x)} = f_1((\mathbf{W}^{(x)}\mathbf{x} + \mathbf{b}^{(x)})_k) \tag{1}$$

où (avec la police "*gras majuscule*" réservée pour des matrices et la petite police "*gras miniscule*" pour des vecteurs), $\mathbf{W}^{(x)}$ est la matrice des poids de dimension $m \times l$, $\mathbf{b}^{(x)}$, un vecteur colonne de longueur m contenant les paramètres biais, et $k = 1, ..., m$. De même, une deuxième fonction de transfert f_2 trace de la couche de codage à la couche du goulot d'étranglement (la couche contenant un neurone simple), qui représente la composante principale non linéaire u;

$$u = f_2(\mathbf{w}^{(x)} \cdot \mathbf{h}^{(x)} + \bar{b}^{(x)}). \tag{2}$$

La fonction de transfert f_1 est généralement non linéaire (habituellement la tangente hyperbolique ou la fonction sigmoïdale, bien que la forme exacte ne soit pas critique), tandis que f_2 est habituellement prise fonction d'identité.

Après, une fonction de transfert f_3 trace de u à la couche cachée finale (la couche de décodage) $\mathbf{h}^{(u)}$,

$$h_k^{(u)} = f_3((\mathbf{w}^{(u)}u + \mathbf{b}^{(u)})_k) \tag{3}$$

$(k = 1, ..., m)$; suivie par f_4, qui trace de u à $\mathbf{x}'$, le vecteur sortie colonne de longueur l, avec

$$x_i' = f_4((\mathbf{W}^{(u)}\mathbf{h}^{(u)} + \bar{\mathbf{b}}^{(u)})_i) \tag{4}$$

La fonction coût $J = \left\langle \left\| \mathbf{x} - \mathbf{x}' \right\|^2 \right\rangle$ est minimisée, en trouvant les valeurs optimales de $\mathbf{W}^{(x)}$, $\mathbf{b}^{(x)}$, $\mathbf{w}^{(x)}$, $\bar{b}^{(x)}$, $\mathbf{w}^{(u)}$, $\mathbf{b}^{(u)}$, $\mathbf{W}^{(u)}$ et $\bar{\mathbf{b}}^{(u)}$. La NLPCA a été implémentée en utilisant la fonction hyperbolique pour f_1 et f_3, et la fonction identité pour f_2 et f_4, donc

$$u = \mathbf{w}^{(x)} \cdot \mathbf{h}^{(x)} + \bar{b}^{(x)}, \tag{5}$$

$$x_i' = (\mathbf{W}^{(u)}\mathbf{h}^{(u)} + \bar{\mathbf{b}}^{(u)})_i. \tag{6}$$

En outre, on adopte les conditions de normalisation : $\langle u \rangle = 0$ et $\langle u^2 \rangle = 1$. Ces conditions sont approximativement satisfaites en modifiant la fonction coût:

$$J = \left\langle \left\| \mathbf{x} - \mathbf{x}' \right\|^2 \right\rangle + \langle u \rangle^2 + \left(\langle u^2 \rangle - 1 \right)^2 . \quad (7)$$

Le nombre de paramètres employés (poids et biais) par le NLPCA est $2lm + 4m + l + 1$. Cependant le nombre effectif de paramètres indépendants est deux fois moins en raison des contraintes sur $\langle u \rangle$ et $\langle u^2 \rangle$.

Le choix de m, nombre de neurones cachés dans les couches de codage et de décodage suit un principe général de parcimonie. Une valeur élevée de m augmente les possibilités de modélisation non linéaires du réseau, mais pourrait mener à des solutions sur-ajustées (c'est-à-dire les solutions ondulées qui ajustent le bruit dans les données). Si f_4 est la fonction d'identité, et $m = 1$, alors l'équation implique que tous les $\mathbf{x}_i'$ sont linéairement reliés à un neurone caché simple, par conséquent il peut seulement y avoir une relation linéaire entre variables $\mathbf{x}_i'$. Ainsi, pour les solutions non linéaires, nous devons avoir $m \geq 2$. Il est également possible d'avoir plus d'un neurone dans la couche du goulot d'étranglement. Par exemple, avec deux neurones dans le goulot d'étranglement, le mode extrait enjambera une courbe à deux dimensions au lieu d'une courbe à une seule dimension.

L'optimisation non linéaire a été effectuée par la fonction MATLAB '**fminu**', un algorithme quasi-newtonien. En raison des minimums locaux dans la fonction coût, il n'y a aucune garantie que l'algorithme d'optimisation atteigne le minimum global. Par conséquent, un certain nombre d'exécutions avec des paramètres (poids et biais) initiaux aléatoires ont été effectués. En outre, 20% des données ont été aléatoirement choisies comme données de validation. Les exécutions pour lesquelles le MSE du test est plus grand que celui de l'apprentissage sont rejetées pour éviter le sur-ajustement. L'exécution pour laquelle le MSE est le plus petit a été choisie comme solution.

En général, le problème le plus sérieux avec NLPCA est la présence des minimums locaux dans la fonction coût. En conséquence, les optimisations commençant à partir de différents paramètres initiaux convergent souvent vers différents minimums, rendant la solution instable ou non unique. Pour une régularisation de la fonction coût, on ajoute des limites de pénalité.

Le but des limites de pénalité des poids est de limiter le pouvoir non linéaire du NLPCA, qui vient des fonctions de transfert non linéaires dans le réseau. Le fonction de transfert *tanh* a la propriété qu'étant donné un x dans l'intervalle $[-L, L]$, on peut trouver un poids assez petit ω, de sorte que le $tanh(\omega x) = \omega x$, c'est-à-dire la fonction de transfert est presque linéaire. De même manière, on peut choisir ω assez grand, de sorte que le tanh approche une fonction échelon, rapportant ainsi les solutions en forme de Z (courbe Z). Si nous pouvons pénaliser l'utilisation des poids excessifs, nous pouvons

limiter le degré de non linéarité dans la solution NLPCA. Cela est réalisé avec une fonction de coût modifiée

$$J = \left\langle \left\| \mathbf{x} - \mathbf{x}' \right\|^2 \right\rangle + \langle u \rangle^2 + \left(\langle u^2 \rangle - 1 \right)^2 + P \sum_{ki} (W_{ki}^{(x)})^2, \tag{8}$$

où P est le paramètre de pénalité des poids. Un P assez large augmente la concavité de la fonction coût, et force les poids $\mathbf{W}^{(x)}$ à être petits en valeur, donnant ainsi des solutions moins non linéaires, contrairement à des valeurs de P petites ou nulles. Par conséquent, l'augmentation de P réduit également le nombre effectif des paramètres indépendants du modèle.

Le pourcentage de variance expliqué par le modèle de NLPCA est simplement

$$100\% \times \left(1 - \frac{\left\langle \left\| \mathbf{x} - \mathbf{x}' \right\|^2 \right\rangle}{\left\langle \left\| \mathbf{x} - \bar{\mathbf{x}} \right\|^2 \right\rangle} \right), \tag{9}$$

avec $\bar{\mathbf{x}}$ est la moyenne de $\mathbf{x}$.

Liste des Figures

Liste des tableaux

Bibliographie

[1] Aoki M. *"State Space Modelling of Time Series"* Springer-Verlag, Nerlin, 1987.

[2] Box G. E. P. & Jenkins J. M. "*Time Series Analysis : Forcasting and Control" 2nd ed.* Holden-Day, San Francusco, 1976.

[3] Brown R. G. "*Smoothing, Forcasting and Prediction*" Prentice Hall, Englewwod Cliffs, NJ, 1963.

[4] Canu Stéphane, Gascuel Olivier, Lechevalier Yves & Thiria Sylvie. "*Statistique et Méthodes Neuronales*" DUNOD 1995.

[5] Capderou M. *"Atlas Solaire de l'Algérie Tome 1 et 2"*, Office des Publications Universitaires (OPU) 1985.

[6] Chatfield Chris. "*Time-series forecasting*" Chapman & Hall/CRC, Boca Raton, Florida, 2001.

[7] Duda R. o. & Hart P. E. "*Pattern Classification and Scene Analysis*" Jhon Wiley and Sons, 1973.

[8] Granger C. W. J. & Andersen A. P. "*Introduction to Bilinear Time Series Models*" Vandenhoek & Ruprect, Goittingen, 1978.

[9] Haykin S. *"Neural Networks : A Comprehensive Foundation"* Prentice Hall, HJ, 2 edition, 1999.

[10] Hérault Jeanny & Jutten Christian. "*Réseaux de Neurones et Traitement du Signal"* HERMES 1994.

[11] Hsieh W. W. *"Nonlinear canonical correlation analysis by neural networks"* Neural Networks, 13, 1095-1105, 2000.

[12] Hsieh W. W. *"Nonlinear principal component analysis by neural networks"* Tellus, 53A, 599-615, 2001a.

[13] Hsieh W. W. *"Nonlinear canonical correlation analysis of the tropical Pacific climate variability using a neural network approach"* J. Clim., 14 , 2528-2539, 2001b.

[14] Hsieh W. W., and K. Hamilton, *"Nonlinear singular spectrum analysis of the tropical stratospheric wind"*, Quart. J. Roy. Met. Soc., 129 , 2367-2382., 2003.

[15] Hsieh W. W., and B. Tang, *"Applying neural network models to prediction and data analysis in meteorology and oceanography"*, Bull. Amer. Meteor. Soc., 79 , 1855-1870, 1998.

[16] Hsieh W. W., and A. Wu, *"Nonlinear multichannel singular spectrum analysis of the tropical Pacific climate variability using a neural network approach"*, J. Geophys. Res., 107(C7), 3076, DOI: 10.1029/2001JC000957, 2002.

[17] Hsieh, W.W. 2002. *"Nonlinear multivariate and time series analysis by neural network methods"* Reviews of Geophysics (submitted).

[18] Jain A. K., Murty M. N., and Flynn P. J. Data clustering : *A review. ACM Computing Surveys*, 31:3:264-323, 1999.

[19] Kramer, M. A. *"Nonlinear principal component analysis using autoassociative neural networks"* AIChE Journal, 37 , 233-243, 1991.

[20] Maafi, A. Thèse de magistère en Electronique appliquée: *Traitement stochastique des données d'ensoleillement en vue de l'optimisation des systèmes photovoltaïques.* Ecole Nationale Polytechnique d'Alger, Avril 1986.

[21] Nicholls D. F. & Pagan A. R. Varying coefficient regression. In E. J. Hannan, P. R. Krishnaiah, and M. M. Rao, editors, "*Handbook of Statistics.*" pages 413-449. North-Holland, Amsterdam, 1985.

[22] Quinlan J. r. *Introduction of decision trees.* Machine Learning, 1:81-106, 1986.

[23] Tong H. "*Nonlinear Time Series : A Dynamical System Approach.*" Oxford University Press, Oxford,1990.

[24] Wah Benjamin W. & Qian Minglun. "*Chapter 8 : Constraint-Based Neural Network Learning for Time Series Prédiction.*" Department of Electrical and Computer Engineering and the Coordinated Laboratory. University of Illinois, Urbana-Champaign, USA.

[25] Wah B. W & Shang Y. "*Global optimization for neural network training.*" IEEE computer, 29:45-54, March 1996.

[26] Wah B. W. & Qian M. L. Constrained formulations for neural network training and their applications to solve the two-spiral problem. In *Proc. Fifth Int'l Conf. on Computer Science and Informatics*, volume 1, pages 598-601, February 2000.

[27] Wan E. *"Temporal backpropagation for FIR neural networks"*, International Joint Conference on Neural Networks, vol. 1. San Diego, 1990, pp. 575-580.

[28] Wan E. *"Finite Impulse Response Neural Networks for Autoregressive Time Series Prediction"* Department of Electrical Engineering, Stanford University, Stanford, CA 94305-4055.

[29] Watkins C. J. "*Models of Delayed Reinforcement Learning*" Ph.D thesis, Cambridge University, Cambridge, UK, 1989.

[30] Weigend, A. S. September 1994. "*Time Series Analysis and Prediction.*" Department of Computer Science and Institute of Cognitive Science. University of Colorado, Boulder CO 80309-0430.

[31] Weigend, A., and Gershenfeld N., edition of *Proceeding of the NATO Advanced Research Workshop on Time Series Prediction and Analysis*, (Santa Fe, New Mexico, May 14-17 1992), Addison-Wesley, 1993.