N° d'ordre :

REPUBLIQUE ALGERIENNE DEMOCRATIQUE & POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR & DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DJILLALI LIABES
FACULTE DES SCIENCES EXACTES
SIDI BEL ABBES

THESE DE DOCTORAT

Présentée par

Monsieur **GASMI Laid**

Spécialité : Mathématiques
Option : Probabilités-Statistiques

Intitulée

Time Series forecasting using neural networks and Genetic algorithms

Soutenue le : 17/ 01/ 2019

Devant le jury composé de :

Président : BENCHIKH Tawfik , Pr., Univ. Sidi Bel Abbès

Examinateur : GUENDOUZI Toufik , Pr., Univ. Saida

Directeur de thèse : CHIKR EL MEZOUAR Zouaoui, M.C.A., Univ. King Khalid Abha, Arabie Saoudite
Co-directeur de thèse : ATTOUCH Med Kadi, Pr., univ. Sidi Bel Abbès

2018/2019

Remerciement

Je tiens à remercier tout premièrement Dieu le tout puissant pour la volonté, la santé et la patience, qu'il nous a donné durant toute longue

Ainsi, je tiens à ex
directeur de thè Zouaoui Chikr Elmezouar, pour m'avoir encadré et soutenu tout au long de
le remercie de m'avoir apporté toute son ex
sans le

J'offre également de sincère Mohammed Kadi Attouch mon codirectrice de thè
et de patience, pour sa disponibilité, son attention et son soutien qui sont sans doute de
cette thè

Je remercie chaleureusement Monsieur Benchikh Cawfik Docteur à l'Université du Sidi Bel Abbè
jury.

Je suis aussi trè Guendouzi Coufik Profe
à l'Université du Saida d'acce

Je voudrais remercier, de façon particulière, Monsieur Abdelghani Ouahab Profe
et de soutien tout au long de cette thè
me
Ahmed Draia Adrar.

Pour finir, je voudrais remercier le ma mère, me) pour leurs soutiens sans faille qu'il

Laid Gasmi

ملخص

التنبؤ في السلسلة الزمنية هو موضوع بحث نشط للغاية في مجال العلوم والهندسة. بسبب صعوبة تقييم الطبيعة الدقيقة لسلسلة زمنية ، غالباً ما يكون من الصعب للغاية توليد تنبؤات مناسبة لأن العديد من مشكلات العالم الحقيقي غير خطية. مؤخرا، تم اقتراح خوارزميات مستوحاة من البيولوجيا، مثل الشبكات العصبية الاصطناعية والخوارزميات الجينية كنهج بديل واعد للتنبؤ بالسلسلة الزمنية. في هذا البحث ، نناقش ونقارن بين هاتين الطريقتين و طريقة بوكس و جنكينز. بالنسبة إلى طريقة الشبكة العصبية ، استخدمنا معيار AIC لحل مشكلة اختيار النموذج (اختيار الشبكة) ، وأعطت هذه الطريقة تفوقاً للشبكات العصبية مقارنةً بنموذج $ARMA$. من أجل تحسين جودة التنبؤ اقترحنا التهجين بين الخوارزميات الجينية ونموذج $ARMA$.

الكلمات المفتاح : السلسلة الزمنية ، بوكس و جنكينز ، $ARMA$، الشبكات العصبية ، الخوارزميات الجينية ، AIC.

Abstract

Time series forecasting is a very active research subject in the field of science and engineering. Because of the difficulty of assessing the exact nature of a time series, it is often very difficult to generate appropriate forecasts because many real-world problems are non-linear. Recently, bio-inspired computational algorithms, such as neural networks and genetic algorithms have been proposed as an approach promising alternative for forecasting time series. In this research, we discuss and compare these two methods and that of Box-Jenkins. For the neural network method we used the AIC criterion to solve the problem of choice of the model (architecture selection), this method gave a superiority to the neural networks compared to the ARMA model. For the improvement of the quality of prediction, we have hybridized the genetic algorithms and the ARMA model.

Key words *: time series, Box-Jenkins, ARMA, neural networks, genetic algorithms, AIC.*

Résumé

La prévision des séries temporelles est un sujet de recherche très actif dans le domaine de la science et de l'ingénierie. En raison de la difficulté d'évaluer la nature exacte d'une série chronologique, il est souvent très difficile de générer des prévisions appropriées, car de nombreux problèmes du monde réel sont non-linéaires. Récemment, les algorithmes computationnels bio-inspirés, tels que les réseaux neuronaux artificiels et les algorithmes génétiques ont été proposés comme une approche alternative prometteuse pour la prévision des séries temporelles. Dans cette recherche, nous avons discuter et comparer ces deux méthodes et celle de Box-Jenkins. Pour la méthode des réseaux de neurones nous avons utiliser le critère de AIC pour résoudre le problème de choix du modèle (sélection d'architecture), cette méthode à donnée une supériorité au réseaux de neurones par rapport au modèle ARMA. Pour l'amélioration de la qualité de prévision nous avons proposé une hybridation entre les algorithmes génétiques et le modèle ARMA.

Mots clés *: séries temporelles, Box- Jenkins, ARMA, réseaux de neurones, algorithmes génétiques, AIC.*

Table des matières

Table des figures

Liste des tableaux

Liste des symboles

$BPTT$ Back Propagation Through Time (rétro-propagation dans le temps)

DNA Deoxyribonucleic acid

MLP multilayer perceptron (Perceptron Multi Couche)

$QQ-plotle$ le graphique Quantile-Quantile

SBC Critère bayesien de Schwartz(Schwartz' Bayesian Criterion)

$\mathcal{L}^2(\Omega,\, A,\, P)$ espace de Hilbert pour le produit scalaire $(X|Y)$

$\bar{\mathcal{L}}\left(\underline{X_{t-1}}\right)$ espace $\bar{\mathcal{L}}\left(1,\, X_{t-1},\, \cdots,\, X_{t-p}, \cdots\right)$

$\beta_1^{1/2}$ le coefficient de Kurtoris

$\chi^2(K)$ loi du Khi-Deux avec K degrés de liberté

$\gamma\left(h\right)$ auto-covariance d'ordre h

$\gamma_X^i(h)$ l'auto-covariance inverse d'ordre h

$\hat{\rho}_n\left(h\right)$ la fonction d'autocorrélation empirique

$\hat{\sigma}_n\left(h\right)$ La fonction d'autocovariance empirique

μ_k le moment centré d'ordre k

∇^d l'opérateur de différence de degré d

$\rho_X^i\left(h\right)$ l'auto-corrélation inverse d'ordre h

ACF fonction d'auto corrélation

$ACPF$ fonction d'auto-corrélation partielle

ADF le test de Dickey-Fuller augmenté

AG Algorithme Génétique

AIC le critère d'Akaike

ANN Artificial Neural Networks

$AR(p)$ auto-régressif d'ordre p

$ARCH$ autoregressive conditional heteroskedasticity

$ARIMA(p,d,q)$ Autoregressive Integrated Moving Average d'ordre $p, d,$ et q

$B-J$ Box et Jenkins

BP Back-Propagation (Rétro-Propagation)

Df Degré de liberté(Degree of freedom)

DS non stationnarité stochastique(Stationary Difference)

$EL(X_t|\underline{X_{t-1}})$ la régression linéaire (ou affine) sur $\bar{\mathcal{L}}\left(\underline{X_{t-1}}\right)$

$EL\left(X|1,\, X_{t-1},\, \cdots,\, X_{t-p}\right)$ la régression affine théorique de X_t sur $X_{t-1},\, \cdots,\, X_{t-p}$

$EL\left(X|X_{t-1},\, \cdots,\, X_{t-p}\right)$ la régression linéaire théorique de X_t sur $X_{t-1},\, \cdots,\, X_{t-p}$

$FFNN$ Feed Forward Neural Networks (réseaux à propagation avant)

FIR Finite Impulse Response

FRC fréquences relatives cumulées

$GARCH$ generalized autoregressive conditional heteroskedastic

HQ Le critère d'information de Hannan - Quinn

IIR Infinite Impulse Response

$L\,(ou\,B)$ lopérateur retard (lag, ou backshift ou backward, operator)

$MA(q)$ moyenne mobile (Moving Average) d'ordre q

MAE l'erreur absolue moyenne (Mean Absolute Error)

$MAPE$ écart absolu moyen en pourcentage (Mean Absolute Percent Error)

MCG Moindres carrés généralisés

MCO Moidres carrés ordinaire

ML Machine Learning (apprentissage artificiel)

MSE l'erreur quadratique moyenne (Mean Squared Error)

PSO Particle Swarm Optimization

Q_{LB} la statistique de Ljung et Box (1978)

$r(p) = \tau(p)$ auto-corrélation partielle d'ordre p

$RBFN$ Radial Basis Function Networks (Réseaux à Fonction à Base Radiale)

$RMSE$ la racine carrée de l'erreur quadratique moyenne(Root Mean Square Error)

$RTRL$ Real Time Recurrent Learning (apprentissage récurrent temps réel)

SIC le critère de Schwarz

$STAR$ smooth transition autoregressive

TCL le théorème central limite

TS non stationnarité déterministe(Trend Stationary)

$\bar{x}_n$ la moyenne

f_X la densité spectral de $(X_t)_{t\in\mathbb{Z}}$

$\beta_2 = \frac{\mu_4}{\mu_2^2}$ le coefficient de Kurtoris

$\hat{\sigma}_n\,(0)$ la variance empirique

$\varepsilon_t \rightsquigarrow BB\left(0,\, \sigma^2\right)$ bruit blanc

Introduction générale

La prévision des séries chronologiques est un sujet de recherche très actif dans le domaine de la science et de l'ingénierie. L'objectif principal de l'analyse des séries chronologiques est de développer un modèle mathématique capable de prévoir les futures observations sur base des données disponibles. En raison de la difficulté d'évaluer la nature exacte d'une série chronologique, il est souvent très difficile de générer des prévisions appropriées. Il existe deux approches différentes pour modéliser les séries temporelles en fonction de la théorie ou de l'hypothèse concernant la relation dans les données. Les méthodes traditionnelles telles que la régression par séries temporelles, le lissage exponentiel et la moyenne mobile intégrée autorégressive (*ARIMA*) sont basées sur des modèles linéaires. En d'autres termes, ils supposent que la valeur future d'une série chronologique est liée linéairement aux observations passées. En particulier, le modèle *ARIMA* est représentatif des modèles linéaires et a acquis une grande popularité depuis la publication du livre classique de Box and Jenkins (1976). D'autre part, un certain nombre de modèles de séries temporelles non linéaires ont été développés au cours des deux dernières décennies, car la plupart des problèmes du monde réel sont non linéaires et l'approximation linéaire de la situation réelle complexe peut ne pas être appropriée.

Traditionnellement, les méthodes de prévision statistique linéaire ont été largement utilisées dans de nombreuses situations du monde réel. Les modèles linéaires sont faciles à développer et à mettre en œuvre. Ils sont aussi simples à comprendre et à interpréter. Pour les situations où un grand nombre d'éléments doit être fourni et / ou la précision n'est pas exigeante, ces caractéristiques sont particulièrement attrayantes. Les modèles linéaires, cependant, ont la limitation en ce que de nombreux problèmes du monde réel sont non-linéaires [Makridakis et al. (1982)]. L'utilisation de modèles linéaires pour approcher des problèmes non linéaires complexes n'est souvent pas satisfaisante, en particulier lorsque l'horizon de prévision est relativement long.

Une variété de modèles de séries temporelles non linéaires ont été proposés dans le but d'améliorer les performances de prévision pour les systèmes non linéaires ; tels que, le modèle *(STAR)* [Chan and Tong (1986)], le modèle *(ARCH)* [Engle (1982)], et le modèle *(GARCH)* [Bollerslev (1986)] reçoivent le plus d'attention.

Les formes pré-spécifiées de modèle limitent également l'utilité de ces modèles puisqu'il y a beaucoup de modèles non linéaires possibles et une forme spécifique peut ne pas capturer toutes les non-linéarités dans les données. Seul un succès (gain) limité a été trouvé au cours des deux dernières décennies dans l'utilisation de modèles non

linéaires [De Gooijer and Kumar (1992)].

Récemment, les algorithmes computationnels bio-inspirés, tels que les réseaux neuronaux artificiels et les algorithmes génétiques ont été proposés comme une approche alternative prometteuse pour la prévision des séries chronologiques[Haykin (1999b) ; Samanta and Nataraj (2008)].

Premièrement, Un grand nombre d'applications réussies ont montré que les réseaux de neurones peut être un outil très utile pour la modélisation et la prévision de séries temporelles [Zhang et al. (1998) ; Balkin and Ord (2000) ; Lapedes and Farber (1987)].

Les réseaux de neurones appartiennent à une famille de modélisation non linéaire généralisée. Théoriquement,Hornik et al. (1989), Cybenko (1989), et Hornik (1991) ont établi que les réseaux de neurones sont des approximateurs fonctionnels universels et peuvent approcher n'importe quelle fonction non linéaire avec une précision arbitraire. C'est une avancée très importante pour les réseaux neuronaux puisque le nombre de possible non linéaire modèles est énorme pour les problèmes du monde réel et un bon modèle devrait être en mesure de les rapprocher tous Granger (1993).

Empiriquement, les réseaux neuronaux se sont avérés efficaces pour modéliser et prévoir des séries temporelles non linéaires avec ou sans bruits [Saxén (1996)]. De nombreuses comparaisons ont été faites entre les réseaux neuronaux et les méthodes traditionnelles (linéaires) sur les performances prévisionnelles des séries chronologiques. Bien que la plupart des chercheurs trouvent que les réseaux neuronaux peuvent surperformer les méthodes linéaires dans une variété de situations, les conclusions ne sont pas cohérentes [Zhang et al. (1998)]. Bien que l'on s'attende en théorie à ce que les réseaux de neurones conviennent aux problèmes de structure non linéaire [Gorr (1994)], il est souvent difficile en réalité à déterminer si un problème à l'étude est linéaire ou non. Dans les années 70 les modèles auto régressifs linéaires *(AR)* ont été largement étudiés et expérimentes. Néanmoins l'hypothèse de linéarité se révèle souvent beaucoup trop restrictive. Depuis, de nombreuses tentatives pour introduire des modèles autorégressifs non-linéaire ont abouti dans certaines cas à de meilleurs modèles. Parmi ceux-ci plus particulièrement les perceptrons multicouches *(MLP)* ont connu un succès important. Cela pour des qualités théoriques comme la facilité d'approximation universelle, mais aussi pour des raisons pratiques comme la facilité du calcul de la dérivée de la fonction representée par le *MLP* par rapport à ses paramètres.

Deuxièmement, les algorithmes génétiques sont des méthodes stochastiques basées sur une analogie avec des systèmes biologiques. Ils reposent sur un codage de variables organisées sous forme de structures chromosomiques et prennent modèle sur les principes de l'évolution naturelle de *Darwin* pour déterminer une solution optimale au problème considéré. Ils ont été introduits par Holland (1975) pour des problèmes d'optimisation complexe. Contrairement aux méthodes d'optimisation classique, ces algorithmes sont caractérisés par une grande robustesse et possèdent la capacité d'éviter les minimums locaux pour effectuer une recherche globale. De plus, ces algorithmes n'obéissent pas aux hypothèses de dérivabilité qui contraignent pas mal de méthodes classiques destinées à traiter des problèmes réels.

Les algorithmes génétiques ont été utilisés dans des domaines aussi divers que l'ingénierie Goldberg (1989a), chimie (Brodmeier and Pretsch (1994), optimisation (Zeiri (1995), acoustique (Gingras and Gerstoft (1995)) reconnaissance de formes (Piper (1995)), et l'économie (Palmer et al. (1999), ou encore en théorie des jeux répétés

(Axelrod et al. (1987)), ainsi que pour l'optimisation de fonctions (De Jong (1980)). Pan et al. (1995) utilisent les AG afin de trouver les paramètres optimaux de régressions nonlinéaires. Dans un autre ordre d'idée, Boné et al. (1998) utilisent les AG pour trouver la forme économétrique théorique la mieux adaptée à la modélisation de séries temporelles. Chez eux, la population de chromosomes définit le type de modélisation (AR,MA ou $ARMA$) et la valeur des coefficients associés. Un résultat de convergence fort pour les AGs basé sur la théorie de Fredlin and Wentzell (1984) des perturbations stochastiques des systèmes dynamiques Cerf (1994). Cette dernière approche est la plus satisfaisante tant sur le plan mathématique, que sur celui de la modélisation, les différents opérateurs étant présentés comme «perturbant» un *processus Markovien* représentant la population à chaque étape. Ici encore il est démontré l'importance de l'opérateur de mutation, le croisement pouvant être totalement absent. L'investissement nécessaire pour la compréhension des démonstrations de *Cerf* est assez important.

Aujourd'hui encore, beaucoup de statisticiens considèrent les AGs comme une boîte noire aux pouvoirs magiques dont on est incapable de théoriser le fonctionnement et à laquelle il serait donc illusoire, voire dangereux, de se fier. Certes, les articles montrant le succés de l'application des AG à des problèmes d'optimisation de plus en plus complexes se sont multipliés. Mais il était clair que, tant que ces méthodes n'auraient pas donné la clef de leur fonctionnement, les statisticiens sérieux se refuseraient à les utiliser. Bien que parfois limitant, un tel raisonnement a eu l'immense avantage de pousser les scientifiques à chercher à comprendre ce qui se passe réellement dans un algorithme génétique du point de vue théorique (voir Cerf (1994)).

L'utilisations des réseaux de neurones ainsi que les algorithmes génétiques , rencontrent un réel succès dans le domaine des séries temporelles. Nous verrons dans la suite de ce travail que bien que ces méthodes soient issus d'une transposition intuitive d'un concept à l'efficacité démontrée, ils s'appuient maintenant, en outre, sur une modélisation qui les rend beaucoup moins mystérieux. Nous espérons que ce travail participera à la diffusion de ces résultats et ainsi qu'à leur extension.

Cette thèse est organisée en quatre chapitres traitant : le problème d'identification et modélisations les séries temporelles par la méthode de Box et Jenkins ; les réseaux de neurones ainsi que les algorithmes génétiques , puis nous avons comparé ces modèles.
Le premier chapitre est la présentation des séries temporelles, l'intérêt des séries temporelles, les domaines d'application et les sources des séries temporelles utilisées dans le chapitre des résultats.
Le deuxième chapitre est consacré à l'état de l'art des réseaux de neurones, entre autre le MLP adopté dans notre projet. Une description de la méthode d'apprentissage utilisée est aussi présentée.
troisième chapitre est une revue de littérature détaillée sur les algorithmes génétiques. Dans cette partie, nous expliquons les différentes étapes de l'algorithme génétique, les opérateurs de reproduction (croisement et mutation) et le choix des paramètres. Nous discutons également l'approche algorithme génétique simple versus l'algorithme génétique itératif, ainsi que le fondement mathématiques des algorithmes génétiques qui est basé sur la théorie Fredlin and Wentzell (1984).
le quatrième chapitre est composé de deux articles en anglais :

pour le premier article ; Nous avons hybridé la méthode Box-Jenkins et les algorithmes génétiques Les résultats ont montré l'efficacité de cette méthode d'hybridation par rapport à la méthode de Box-Jenkins,La série utilisée dans cet article était le nombre d'accidents de la route mortels au Koweït.

Pour le deuxième article, nous avons étudié et modélisé une série temporelle qui représente la production d'électricité en utilisant l'énergie éolienne dans le champ de "Kabertene"(dans le sud de l'Algérie), du 01/01/2015 au 31/12/2015, en utilisant la méthode Box Jenkins et en utilisant des réseaux de neurones. On utiliser le critère de AIC pour choisir la meilleure architecture de réseaux de neurones. L'analyse des résultats a confirmé l'efficacité des réseaux neuronaux dans la prévision.

Nous finirons cette thèse par une conclusion et des recommandations et des suggestions sur les travaux futurs dans ce domaine de recherche. Pour les programmations, nous avons utilisé le logiciel R [1].

1. = http ://www.R-project.org,

CHAPITRE 1

Éléments d'analyse des séries temporelles

Nous étudions les caractéristiques statistiques- en terme de stationnarité-des séries temporelles en présentant les différents tests(Dickey-Fuller,corrélogramme, etc...) s'y rapportant. Puis, nous présentons différentes classes de modèles (AR, MA, ARMA) en étudiant leurs propriétés. Enfin, la méthode de Box et Jenkins qui systématise une démarche d'analyse des séries temporelles.

1.1 Introduction et premières définitions

L'étude des séries temporelles, ou séries chronologiques, correspond à l'analyse statistique d'observations régulièrement espacées dans le temps. Elles ont été utilisées en astronomie ('on the periodicity of sunspots', 1906), en météorologie ('time-seires regression of sea level on weather', 1968), en théorie du signal ('Noise in FM receivers', 1963), en biologie ('the autocorrelation curves of schizophrenic brain waves and the power spectrum', 1960), en économie (' time-series analysis of imports, exports and other economic variables ', 1971)...etc.(voir Charpentier (2006))
Pour la majorité de ces phénomènes, il existe souvent une dépendance temporelle entre les observations, ce qui a des modélisations de type autorégressif : on utilise le passé pour expliquer le présent et prédire le futur. Modéliser et prédire une série chronologique suppose, dans la plupart des cas, de faire des hypothèses sur son comportement ; Puiqu'il s'agit de séries non-déterministes, il va falloir que la comporsante aléatoire ' varie, mais pas trop'. Cette condition se traduira par la « stationnarité », qui implique une certaine régularité du processus et permet de dériver ses propriétés asymptotiques.
Dans toute la suite, on se place dans un cadre paramétrique : trouver un modèle pour les observations revient à déterminer les paramètres d'une fonction dont la forme est connue 'a priori' et qui vérifie

$$y_t = f(\epsilon_t, y_{t-1}, ...)$$

Où y_t représente l'observation à l'instant t et ϵ_t est le bruit aléatoire.
De puis les années 20(par exemple l'étude de **Yule** sur les taches solaires publiée en 1927) et jusqu 'aux années 80, les modèles linéaires à bruit gaussien AR ont tenu la

tête d'affiche. Ils modélisent le présent par une combinaison linéaire d'un nombre fini de retards plus un bruit :

$$y_t = a_0 + a_1 y_{t-1} + a_2 y_{t-2} + a_p y_{t-p} + \epsilon_t$$

Largement utilisés pendant ce temps. Cependant, de puis une vingtaine d'années, les limitations des modèles linéaires ont été soulignées. Les données réelles ayant souvent des caractéristiques non-linéaires qui ne sont pas prises en compte. Dans les séries financières, par exemple, on observe des périodes très fragiles, suivies de périodes plus stables. Ce phénomène appelé "volatility clustering".
Pour résoudre les problèmes évoqués, des modèles plus complexes ont été proposés à partir des années 80 : Les modèles à volatilité conditionnelle de type$ARCH$, $GARCH$ [Engle (1982), Bollerslev (1986)], les modèles à seuil ou linéaires par morceaux (Tong provenant de l'itelligence artificielle comme les perceptrons multicouches qui ont la propriété pratique d'approximateurs universels [Hornik et al. (1989)].

Définition 1.1.1. *(serie temporelle)*
Une série temporelle (ou série chronologique) à temps discret est une suite réelle finie $(x_t)_{1\leq t\leq n}$ *,où t représente le temps (en minute, jour, année...).*

Voici quelques exemples de séries temporelles [Jacques (2013)].

Exemple 1.1. *Nombre de passagers par mois (en milliers) dans les transports aériens, de 1949 à 1960 (la Figure(1.1)).*

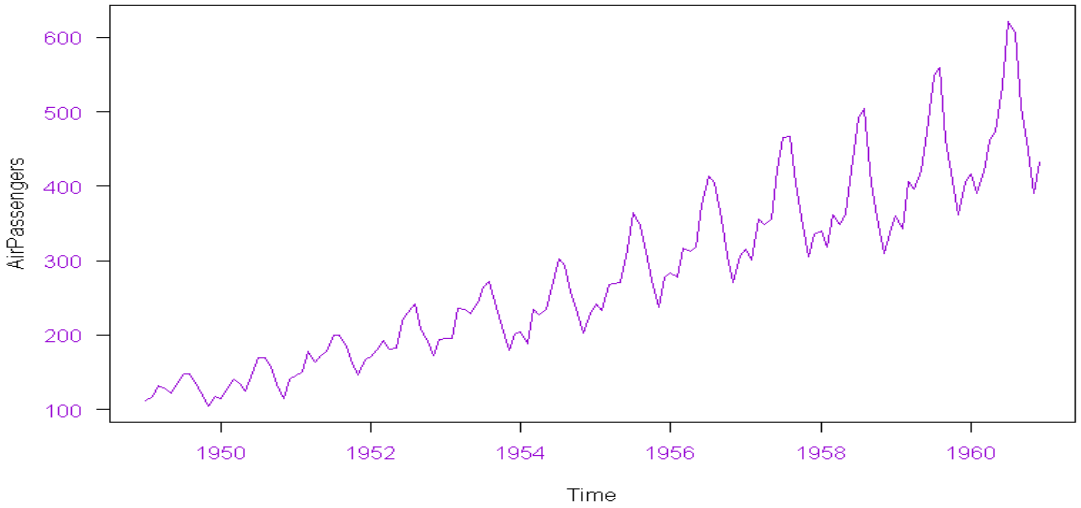

fig. 1.1 – Nombre de passagers (en milliers) dans les transports aériens

Un des objectifs principaux de l'étude d'une série temporelle est la prévision des réalisations futures, très souvent pour des raisons économiques (prévoir l'évolution de la vente d'un produit pour ajuster au mieux les moyens de production, prévoir l'évolution d'un marché financier ...). Bien entendu, aucun modèle ne correspond exactement à la réalité, et il est impossible de prévoir parfaitement le devenir d'une série temporelle. Lorsque cela sera possible, nous donnerons des intervalles de prévisions, afin de

pouvoir apporter une information quant à la précision de la prévision.
Pour ce faire, il existe un large choix de modèles utilisables :

- les modèles de régression, comme par exemple :

$$x_t = \alpha_1 t + \alpha_2 + \epsilon_t,\ t = 1, ..., n$$

  Une fois les coefficients de ce modèle estimés, la prévision de x_{t+1} sera

$$\hat{x}_{t+1} = \hat{\alpha}_1 (t+1) + \hat{\alpha}_2$$

  .
- les lissages exponentiels qui sont très simples à mettre en œuvre ;
- les modèles de type ARMA, qui consistent à enlever de la série les tendances et saisonnalités (ou périodicités) évidentes et à modéliser le résidu restant. Ces méthodes sont plus sophistiquées et plus lourdes numériquement (temps de calcul) que les précédentes, mais également plus performantes.

Exemple 1.2. *(Morts par accident) Le nombre mensuel de morts et blessés graves par accident de la route en France métropolitaine (la Figure(1.2) haut) montre d'importantes fluctuations saisonnières. En vérité elles sont peu explicites sur la série complète, alors qu'un zoom de la série sur quelques années [Yves (2011)] (la Figure(1.2) bas) montre que les mois de décembre et janvier présentent moins d'accidents.*

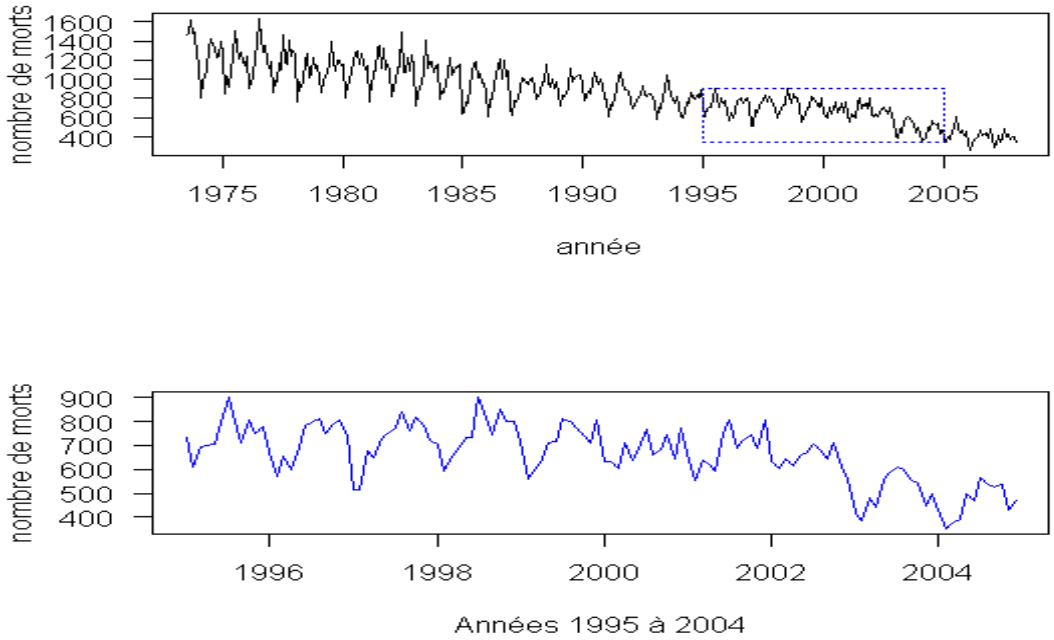

fig. 1.2 – Accidents de la route - morts à 30 jours

On note également une décroissance du niveau moyen de la série, qui s'accélère à partir de l'année 2000.

Parmi les exemples précédents [1], celui relatif au nombre de passagers dans les transports aériens (la Figure(1.1)) est une série assez typique de ce que l'on rencontre en économétrie, et elle donne lieu à de bonnes prévisions pour toutes les méthodes classiques.
Les défis que nous allons devoir relever sont les suivants :

- définir un modèle avec un nombre fini de paramètres ;
- estimer les paramètres de ce modèle ;
- vérifier la qualité d'ajustement du modèle, comparer différents modèles (partage de l'échantillon d'observations en 80% pour l'apprentissage et 20% pour le test) ;
- effectuer des prédictions.

1.1.1 Tendances et composantes saisonnières

On parle de tendance lorsque la série $(x_t)_{1\leq t\leq n}$ peut s'écrire, à une erreur d'ajustement ε_t près, comme une combinaison linéaire de m fonctions du temps, choisies a priori (par exemple fonction puissance, exponentielle, logarithmique...) :

$$x_t = \sum_{j=1}^{m} \alpha_j f_j(t) + \varepsilon_t \quad 1 \leq t \leq n\,.$$

Lorsque $x_t = \alpha t + \beta + \varepsilon_t$ la tendance est linéaire ($m = 1$ et $f(t) = \alpha t + \beta$). Une tendance polynomiale se traduira par

$$x_t = \alpha_1 t^p + \alpha_2 t^{p-1} + \cdots + \alpha_{p+1} + \varepsilon_t\,.$$

On parle de composante périodique lorsque la série $x_t = \alpha t + \beta + \varepsilon_t$ peut se décomposer en :

$$x_t = s_t + \varepsilon_t \; 1 \leq t \leq n\ ,$$

où s_t est périodique, c'est-à-dire $s_{t+T} = s_t$, avec T la période (supposée entière). Lorsque la période est de *6 mois ou 1 an*, on parle généralement de composante saisonnière.
Il est fréquent qu'une série comporte à la fois une tendance et une composante périodique (voir figure 1.1).

1.1.2 Indices descriptifs d'une série temporelle

Indices de tendances centrales

Considérons un ensemble d'observations $x_1, \ldots, x_n$.
Nous utilisons comme indicateur de la tendance centrale la moyenne :

$$\bar{x}_n = \frac{1}{n}\sum_{t=1}^{n} x_t.$$

1. Ces données sont disponibles dans le logiciel R.

Indices de dispersion

Nous utilisons comme indicateur de dispersion la variance empirique :

$$\hat{\sigma}_n(0) = \frac{1}{n}\sum_{1}^{n}(x_t - \bar{x}_n)^2$$

.

Indices de dépendance

La fonction d'autocovariance empirique est donnée par :

$$\hat{\sigma}_n(h) = \frac{1}{n-h}\sum_{t=1}^{n-h}(x_t - \bar{x}_n)(x_{t-h} - \bar{x}_n),$$

et **la fonction d'autocorrélation empirique** est donnée par

$$\hat{\rho}_n(h) = \frac{\hat{\sigma}_n(h)}{\hat{\sigma}_n(0)}.$$

Ce sont les auto-corrélations empiriques que nous utiliserons pour caractériser la dépendance entre les variables [Jacques (2013)].

1.2 Estimation et élimination de la tendance et de la saisonnalité

Une série temporelle $(x_t)_{1\leq t\leq n}$ est l'observation des n premières réalisations d'un processus stochastique $(X_t)_t$. C'est ce processus que l'on cherche désormais à modéliser. Pour cela, la démarche suivante doit être adoptée :

- représenter graphiquement la série afin de repérer les tendances et saisonnalités ;
- estimer et supprimer les tendances et saisonnalités (partie déterministe du processus stochastique) ;
- choisir un modèle pour les résidus (partie aléatoire du processus stochastique) et l'estimer ;
- prédire les réalisations futures à l'aide de ce modèle.

1.2.1 Processus stationnaire du second ordre

Dans toute la suite on considèrera $(X_t)_{t\in\mathbb{Z}}$ avec $X_t \in \mathcal{L}^2(\Omega, A, P)$ pour tout $t \in \mathbb{Z}$.

Définition 1.2.1. *$(X_t)_{t\in\mathbb{Z}}$ est un processus stationnaire au sens strict si :*
$\forall n \in \mathbb{N}, \forall (t_1, \cdots, t_n), \forall h \in \mathbb{Z}$, la loi de $(X_t, \cdots, X_{t_n})$ est identique à la loi de $(X_{t_1+h}, \cdots, X_{t_n+h})$

Théorème 1.2.1. *(théorème de Kolmogorov)*
$(X_t)_{t\in\mathbb{Z}}$ est un processus stationnaire au sens strict si et seulement si la loi de $(X_t)_{t\in\mathbb{Z}}$ est identique à la loi de $(Y_t)_{t\in\mathbb{Z}}$ où $Y_t = X_{t+h}$.

Définition 1.2.2. *$(X_t)_{t\in\mathbb{Z}}$ est un processus stationnaire du second ordre (ou un processus faiblement stationnaire) s'il vérifie :*

1. $\forall t \in \mathbb{Z},\ E(X_t) = m$;
2. $\forall t \in \mathbb{Z},\ \text{var}(X_t) = \sigma^2 = \gamma(0)$;
3. $\forall t \in \mathbb{Z},\ \forall h \in \mathbb{Z},\ cov(X_t,\ X_{t+h}) = \gamma(h)$ *(ne depend que de h)* ;
   $\gamma(h)$ *est l'auto-covariance d'ordre* h *de* X_t.

Remarque 1.1.

— *si un processus est stationnaire au sens strict alors il est faiblement stationnaire ;*

— *dans la suite on considère le cas de la définition 1.2.2 (processus stationnaires du second ordre) ;*

— *si* $(X_t)_{t\in\mathbb{Z}}$ *est un processus gaussien alors il y'a équivalence entre staionnarités faible et forte ;*

— $E\begin{pmatrix} X_t \\ \vdots \\ X_{t_n} \end{pmatrix} = \begin{pmatrix} m \\ \vdots \\ m \end{pmatrix}$ *et* $\text{var}\begin{pmatrix} X_t \\ \vdots \\ X_{t_n} \end{pmatrix} = \begin{pmatrix} \gamma(0) & & \gamma(t_j - t_i) \\ & \ddots & \\ & & \gamma(0) \end{pmatrix}$.

Exemple 1.3. *Le processus* $(\varepsilon_t)_{t\in\mathbb{Z}}$ *(bruit blanc faible) si et seulement si :*

— $E(\varepsilon_t) = 0,\ \forall t \in \mathbb{Z}$;

— $\text{var}(\varepsilon_t) = \sigma^2,\ \forall t \in \mathbb{Z}$;

— $\text{cov}(\varepsilon_t,\ \varepsilon_\tau) = 0,\ \forall t \neq \tau,\ \varepsilon_t \rightsquigarrow BB\left(0,\ \sigma^2\right)$.

Remarque 1.2.

ε_t *est un bruit blanc fort si et seumement si les* ε_t *sont i.i.d.,* $E(\varepsilon_t) = 0$ *et* $\text{var}(\varepsilon_t) = \sigma^2$.

Exemple 1.4 (Marche aléatoire (processus non stationnaire))**.** *Soit* $\varepsilon_t \rightsquigarrow BB\left(0,\ \sigma^2\right)$, $(X_t)_{t\in\mathbb{Z}}$ *est une marche aléatoire (random wolk) si et seulement si*

i) $X_t = X_{t-1} + \varepsilon_t,\ \forall\, t \geq 0$;

ii) $\text{cov}(\varepsilon_t,\ X_{t-k}) = 0,\ \forall\, 0 < k \leq t$.

Même si $E(X_t) = E(X_{t-1}) \Rightarrow E(X_t) = m,\ \forall t \in \mathbb{Z},\ (X_t)_t$, *n'est pas stationnaire :*

$$\left.\begin{array}{rl} X_t & = X_{t-1} + \epsilon_t \\ X_{t-1} & = X_{t-2} + \epsilon_t \\ & \vdots \\ X_1 & = X_0 + \epsilon_0 \end{array}\right\} \Rightarrow X_t = X_0 + \sum_{k=1}^{t} \epsilon_k$$

D'où

$$v(X_t) = v(X_0) + 2\sum_{k=1}^{t} \text{cov}(\varepsilon_t,\ X_0) + v\left(\sum_{k=1}^{t} \varepsilon_k\right) = v(X_0) + t\sigma^2$$

Le processus n'est pas stationnaire en variance.

Définition 1.2.3 (Fonction d'auto-covariance)**.**
L'auto-covariance d'un processus stationnaire $(X_t)_{t\in\mathbb{Z}}$ *est définie par :*

$$\begin{aligned} \gamma\ :\ & \mathbb{Z} \to \mathbb{R} \\ & h \mapsto \gamma(h) = \text{cov}(X_t,\ X_{t-h}) \end{aligned}$$

Proposition 1.2.1.

1. *γ est une fonction paire : $\gamma(-h) = \gamma(h)\ \forall h$;*
2. *γ est de type positif : $\forall n \in \mathbb{N}, \forall (t_1, \cdots, t_n), \forall (a_1, \cdots, a_n) \in \mathbb{R}^n$*

$$\sum_{1 \leq i,\, j \leq n} a_i a_j \gamma(t_i - t_j) > 0.$$

Démonstration.

1.

$$\begin{aligned}\gamma(h) &= \operatorname{cov}(X_t,\, X_{t+h}) = \operatorname{cov}\left(X_{t-h},\, X_{(t-h)+h}\right) = \operatorname{cov}(X_{t-h},\, X_t) \\ &= \operatorname{cov}(X_t,\, X_{t-h}) = \gamma(-h)\end{aligned}$$

2.

$$\begin{aligned}var(\sum a_i X_{t_i}) &= cov(\sum_i a_i X_{t_i}, \sum_j a_j X_{t_j}) \\ &= \sum_{i,j} a_i a_j cov(X_{t_i}, X_{t_j}) \\ &= \sum_{i,j} a_i a_j \gamma(t_i - t_j) \geq 0\end{aligned}$$

On suppose toujours qu'il n'y a pas de relations linéaires entre les X_t. En effet, si on avait $\operatorname{var}(\sum a_i X_{t_i}) = 0$ alors $\sum a_i X_{t_i} = cst.$ presque sûrement.

□

Définition 1.2.4 (Fonction d'auto-corrélation)**.**
La fonction d'auto-corrélation d'un processus stationnaire $(X_t)_{t \in \mathbb{Z}}$ est définie par :

$$\forall h \in \mathbb{Z},\ \ \rho(h) = \frac{\gamma(h)}{\gamma(0)} = corr(X_t,\, X_{t+h})$$

Proposition 1.2.2. *$\rho : h \to \rho(h)$ est une fonction paire, de type positif, à valeurs dans $]-1,\, 1[$.*

Démonstration.

$$corr(X_t,\, X_{t+h}) = \frac{\operatorname{cov}(X_t,\, X_{t+h})}{\sqrt{\operatorname{var}(X_t) \operatorname{var}(X_{t+h})}} = \frac{\gamma(h)}{\gamma(0)}$$

où γ est paire de type positif. □

1.2.2 Rappels sur $\mathcal{L}^2(\Omega,\, A,\, P)$

$\mathcal{L}^2(\Omega,\, A,\, P)$ est une espace de Hilbert pour le produit scalaire $(X|Y) = E(XY)$.

$$X_n \underset{\mathcal{L}^2}{\longrightarrow} X \Leftrightarrow \lim_{n \to +\infty} \|X_n - X\|_2 = 0$$

si $\sum_{j \in \mathbb{Z}} \|a_j X_j\|_2 = \sum_{j \in \mathbb{Z}} |a_j| \|X_j\|_2 < +\infty$ alors la série $\sum_{j \in \mathbb{Z}} a_j X_j$est définie $p.s.$ et :

$$\sum_{j=-p}^{q} a_j X_j \xrightarrow[p,q \to +\infty]{} \sum_{j \in \mathbb{Z}} a_j X_j$$

Théorème 1.2.2. *Projection sur un s.e.v. fermé H de $\mathcal{L}^2(\Omega, A, P)$*
$\forall X \in \mathcal{L}^2(\Omega, A, P), \exists! X^* \in H / \|X - X^*\|_2 = \min_{Y \in H} \|X - Y\|_2$
$P_H(X) = X^*$ est caractérisé par $X^* \in H$ et $X - X^* \in H^\perp$.

Théorème 1.2.3 (Théorème des trois perpendiculaires)**.** *Soit H un s.e.v. fermé de $\mathcal{L}^2(\Omega, A, P)$, G un s.e.v. fermé de H, alors :*

$$\forall X \in \mathcal{L}^2(\Omega, A, P), P_G(P_H(X)) = P_G(X)$$

1.2.3 Outils pour l'étude des processus stationnaires

Transformée d'un processus stationnaire par $MA(\infty)$

Proposition 1.2.3. *Soient $(X_t)_{t\in\mathbb{Z}}$ un processus stationnaire et $(a_j)_{j\in\mathbb{Z}}$ une suite de réels tels que $\sum_j |a_j| < +\infty$. Alors $Y_t = \sum_{j\in\mathbb{Z}} a_j X_{t-j}$ est defini (p.s.) pour tout t . On a les proprieties suivantes :*

1. *$Y_t \in \mathcal{L}^2(\Omega, A, P), \forall t \in \mathbb{Z}$*
2. *$(Y_t)_t$ est un processus stationnaire tel que*

$$E(Y_t) = m_Y = \left(\sum_{j\in\mathbb{Z}} a_j\right) m_X$$

$$\gamma_Y(h) = \sum_{j,k} a_j a_k \gamma(h + k - j) = \sum_{j,k} a_j a_k \gamma(h + j - k), \forall h \in \mathbb{Z}$$

On dit que $(Y_t)_{t\in\mathbb{Z}}$ est la transformée de $(X_t)_{t\in\mathbb{Z}}$ par la moyenne mobile infinite associée aux $(a_j)_{j\in\mathbb{Z}}$.

Démonstration.

1. $\sum_j \|a_j X_{t-j}\|_2 = \sum_j |a_j| \|X_{t-j}\|_2 = \left(\sum_j |a_j|\right)\left(m_X^2 + \gamma_X(0)\right)^{\frac{1}{2}} < +\infty$, Y_t est donc défini *p.s.* et $Y_t \in \mathcal{L}^2(\Omega, A, P)$
2. On a alors :

$$\begin{aligned}
E(Y_t) &= \int_\Omega Y_t dP = \int_\Omega \left(\sum_{j\in\mathbb{Z}} a_j X_{t-j}\right) dP \\
&= \sum_{j\in\mathbb{Z}} \left(\int_\Omega X_{t-j} dP\right) \text{(Fubini)} \\
&= E(\sum_{j\in\mathbb{Z}} a_j X_{t-j}) \\
&= \sum_{j\in\mathbb{Z}} E(X_{t-j}) \\
&= m_X \left(\sum_j a_j\right)
\end{aligned}$$

Enfin :

$$\begin{aligned} cov\left(Y_t,\, Y_{t-h}\right) &= cov\left(\sum_{j\in\mathbb{Z}} a_j X_{t-j},\, \sum_{k\in\mathbb{Z}} a_k X_{t-h-k}\right) \\ &= \sum_j \sum_k a_j a_k \underbrace{cov\left(X_{t-j},\, X_{t-h-k}\right)}_{\gamma_X(h+k-j)} \end{aligned}$$

□

Définition 1.2.5. *Si* $X_t = \varepsilon_t \rightsquigarrow \mathcal{BB}\left(0,\, \sigma^2\right)$ *alors* $Y_t = \sum_{j\in\mathbb{Z}} a_j \varepsilon_{t-j}$ *et on dit que* $Y_t \rightsquigarrow \mathcal{MA}\left(\infty\right)$.

Définition 1.2.6. *Soit* $(X_t)_{t\in\mathbb{Z}}$ *un processus stationnaire.*

- *La régression linéaire théorique de* X_t *sur* $X_{t-1},\, \cdots,\, X_{t-p}$ *est la projection orthogonale dans* $\mathcal{L}^2\left(\Omega,\, A,\, P\right)$ *de* X_t *sur* $H = vect\left(X_{t-1},\, \cdots,\, X_{t-p}\right)$.
  $EL\left(X|X_{t-1},\, \cdots,\, X_{t-p}\right)$ *représente la régression linéaire théorique de* X_t *sur* $X_{t-1},\, \cdots,\, X_{t-p}$.
- *La régression affine théorique de* X_t *sur* $X_{t-1},\, \cdots,\, X_{t-p}$ *est la projection orthogonale dans* $\mathcal{L}^2\left(\Omega,\, A,\, P\right)$ *de* X_t *sur* $H^* = vect\left(X_{t-1},\, \cdots,\, X_{t-p}\right)$.
  $EL\left(X|1,\, X_{t-1},\, \cdots,\, X_{t-p}\right)$ *est la régression affine théorique de* X_t *sur* $X_{t-1},\, \cdots,\, X_{t-p}$.

Remarque 1.3.

i) Si $E\left(X_t\right) \neq 0$, *on calculera toujours la régression affine. On la note souvent* $EL\left(X|X_{t-1},\, \cdots,\, X_{t-p}\right)$.

ii) Si $\left(X_t\right)_t$ *est gaussien, alors* $EL\left(X_t\,|.\right) = E\left(X_t|.\right)$.

Définition 1.2.7. *On appelle auto-corrélation partielle d'ordre p*

$$\begin{aligned} r(p) &= cor(X_t - EL(X_t|X_{t-1},\cdots,X_{t-p+1}), X_{t-p} - EL(X_{t-p}|X_{t-1},\cdots,X_{t-p+1})) \\ &= \frac{cov(X_t - EL(X_t|X_{t-1},\cdots,X_{t-p+1}), X_{t-p} - EL(X_{t-p}|X_{t-1},\cdots,X_{t-p+1}))}{[var(X_t - EL(X_t|X_{t-1},\cdots,X_{t-p+1}))var(X_{t-p} - EL(X_{t-p}|X_{t-1},\cdots,X_{t-p+1}))]^{1/2}} \end{aligned}$$

Avec $r\left(p\right) = a_p$ *coefficient de* X_{t-p} *dans* $EL\left(X_t|X_{t-1},\, \cdots,\, X_{t-p}\right)$, *et*

$$EL\left(X_{t-p}|X_{t-1},\, \cdots,\, X_{t-p+1}\right) = \sum_{j=1}^{p} a_j X_{t-j}$$

Pour la démonstration, on utilise le théorème de Frisch-Waugh.

Définition 1.2.8. *L'auto-corrélogramme partiel de* $(X_t)_{t\in\mathbb{Z}}$ *est le graphe de :*

$$\begin{aligned} r : \mathbb{N} &\to]-1, 1[\\ p &\mapsto r(p) \end{aligned}$$

Définition 1.2.9. *Soit* $(X_t)_{t\in\mathbb{Z}}$*un processus stationnaire.*

i) La régression linéaire théorique de X_t *sur* $X_{t-1},\, \cdots,\, X_{t-p}, \cdots$ *est la projection orthogonale dans* $\mathcal{L}^2\left(\Omega,\, A,\, P\right)$ *de*X_t *sur* $H = vect\left(X_{t-1},\, \cdots,\, X_{t-p}, \cdots\right)$.

ii) La régression affine théorique de X_t *sur* $1, X_{t-1}, \cdots, X_{t-p}, \cdots$ *est la projection orthogonale dans* $\mathcal{L}^2(\Omega, A, P)$ *de* X_t *sur* $H^* = vect(1, X_{t-1}, \cdots, X_{t-p}, \cdots)$.

On note aussi $\bar{\mathcal{L}}\left(\underline{X_{t-1}}\right)$ *l'espace* $\bar{\mathcal{L}}(1, X_{t-1}, \cdots, X_{t-p}, \cdots)$ *et :*

$$\begin{aligned} EL(X_t|\underline{X_{t-1}}) &= EL(X_t|X_{t-1}, \cdots, X_{t-k}, \cdots) \\ &= EL(X_t|1, X_{t-1}, \cdots, X_{t-k}, \cdots) \end{aligned}$$

la régression linéaire (ou affine) sur $\bar{\mathcal{L}}\left(\underline{X_{t-1}}\right)$.

Théorème 1.2.4. *Soient* $(X_t)_{t\in\mathbb{Z}}$ *un processus stationnaire et* $X_t^* = EL\left(X_t|\underline{X_{t-1}}\right)$ *la régression affine de* X_t *sur* $\bar{\mathcal{L}}(1, X_{t-1}, \cdots, X_{t-p}, \cdots)$ *et* $\varepsilon_t = X_t - X_t^*$ *, alors*

— $(\varepsilon_t)_{t\in\mathbb{Z}}$*est un bruit blanc*

— $\text{cov}(\varepsilon_t, \varepsilon_{t-k}) = 0, \forall k > 0$

Définition 1.2.10. *Avec les notations du théorème ci-dessus :*

i) $(\varepsilon_t)_{t\in\mathbb{Z}}$*est le processus des innovations de* $(X_t)_{t\in\mathbb{Z}}$

ii) ε_t *est l'innovation de* X_t

iii) X_t^* *est la prévision optimale de* X_t *à l'instant* $(t-1)$

Remarque 1.4. $\varepsilon_t = X_t - X_t^* = X_t - EL\left(X_t|\underline{X_{t-1}}\right)$, *donc* $\varepsilon_t \perp 1$ *et* $\varepsilon_t \perp X_{t-k}, \forall k > 0$, *ce qui s'interpréter par :*

$$E(\varepsilon_t) = 0 \text{ et } \forall k > 0,\ E(\varepsilon_t X_{t-k}) = \text{cov}(\varepsilon_t, X_{t-k}) = 0$$

Théorème 1.2.5 (H. Wold)**.** *Soient* $(X_t)_{t\in\mathbb{Z}}$ *un processus stationnaire et* $(\varepsilon_t)_{t\in\mathbb{Z}}$ *le processus des innovations correspondant. Alors*

$$\exists (a_k)_{k\in\mathbb{Z}} / \sum_{k=0}^{+\infty} |a_k| < +\infty \text{ et } X_t = m + \sum_{k=0}^{+\infty} a_k \varepsilon_{t-k}$$

Proposition 1.2.4 (Densité spectrale)**.** *Soit* $(X_t)_{t\in\mathbb{Z}}$ *un processus stationnaire de la forme :*

$$X_t = m + \sum_{j=0}^{+\infty} a_j \varepsilon_{t-j} \text{ où } (\varepsilon_t)_{t\in\mathbb{Z}} \rightsquigarrow \mathcal{BB} \text{ et } \sum_{j=0}^{+\infty} |a_j| < +\infty$$

Alors

i) $\sum\limits_{h\in\mathbb{Z}} |\gamma_X(h)| < +\infty$

ii) $\forall \omega \in [-\pi, \pi]$, $f_X(\omega) = \frac{1}{2\pi} \sum\limits_{h\in\mathbb{Z}} \gamma_X(h) e^{i\omega h}$, f_X *est la densité spectral de* $(X_t)_{t\in\mathbb{Z}}$

Démonstration.

$$\sum_{h\in\mathbb{Z}} |\gamma_X(h)| = \sum_{h\in\mathbb{Z}} \left| \sum_{j,k} a_j a_k \gamma_\varepsilon(h + j - k) \right|,$$

et

$$\gamma(h + j - k) = \begin{cases} 0 & \text{si } h + j - k \neq 0 \\ \sigma_\epsilon^2 & \text{si } h + j - k = 0 \end{cases}$$

Danc

$$\sum_{h\in\mathbb{Z}} |\gamma_X(h)| = \sum_{h\in\mathbb{Z}} \left| \sigma_\varepsilon^2 \sum_j a_j a_{h+j} \right| \leq \sigma_\varepsilon^2 \sum_{h,j} |a_j|\,|a_{h+j}| = \sigma_\varepsilon^2 \left(\sum_j a_j \right)^2 < +\infty$$

□

Proposition 1.2.5. *Sous les hypothèses précédentes,*

$$f_X(\omega) = \frac{1}{2\pi} \sum_{h\in\mathbb{Z}} \gamma_X(h) \cos(\omega h)$$

Démonstration.

$$\begin{aligned} f_X(\omega) &= \frac{1}{2\pi}[\gamma_X(0) + \sum_{h>0} \gamma_X(h) e^{i\omega h} + \sum_{h<0} \gamma_X(h) e^{i\omega h}] \\ &= \frac{1}{2\pi}[\gamma_X(0) + \sum_{h>0} \gamma_X(h) (\underbrace{e^{i\omega h} + e^{-i\omega h}}_{2cos(\omega h)})] \\ &= \frac{1}{2\pi}[\gamma_X(0) + \sum_{h\neq 0} \gamma_X(h) cos(\omega h)] \\ &= \frac{1}{2\pi}[\sum_{h\in\mathbb{Z}} \gamma_X(h) cos(\omega h)] \end{aligned}$$

□

Théorème 1.2.6 (Injectivité)**.** *Avec les notations précédentes,*

$$\forall h \in \mathbb{Z},\ \gamma_X(h) = \int_{[-\pi,\pi]} f_X(\omega) e^{-i\omega h} d\omega = \int_{[-\pi,\pi]} f_X(\omega) cos(\omega h) d\omega$$

Démonstration.

$$\begin{aligned} \int_{[-\pi,\pi]} f_X(\omega) e^{-i\omega h} d\omega &= \frac{1}{2\pi} \int_{[-\pi,\pi]} \left(\sum_{k\in\mathbb{Z}} \gamma_X(k) e^{i\omega} \right) e^{-i\omega h} d\omega \\ &= \frac{1}{2\pi} \sum_{k\in\mathbb{Z}} \gamma_X(k) \underbrace{\left(\int_{[-\pi,\pi]} e^{-i\omega(k-h)} d\omega \right)}_{=\begin{cases} 0 & \text{si } k \neq h \\ 2\pi & \text{si } k = h \end{cases}} \qquad = \gamma_X(h) \end{aligned}$$

□

Conséquence 1.2.1. *La fonction $f_X \mapsto \gamma_X$ est une bijection donc $(X_t)_t$ est caractérisé complètement.*

Définition 1.2.11 (Auto-corrélations inverses)**.** *Soit $(X_t)_{t\in\mathbb{Z}}$ tel que $X_t = \sum_{j\in\mathbb{Z}} a_j \varepsilon_{t-j}$ où $\varepsilon_t \rightsquigarrow \mathcal{BB}$ et $\sum_j |a_j| < +\infty$. On suppose que $\omega \mapsto \frac{1}{f_X(\omega)} e^{-i\omega h}$ est integrable sur $[-\pi, \pi]$. On appelle auto-covariance inverse d'ordre h de $(X_t)_{t\in\mathbb{Z}}$*

$$\gamma_X^i(h) = \int_{[-\pi,\pi]} f_X(\omega) e^{-i\omega h} d\omega$$

L'auto-corrélation inverse d'ordre h est alors définie comme

$$\rho_X^i(h) = \frac{\gamma_X^i(h)}{\gamma_X^i(0)}$$

1.2.4 Une estimation paramétrique de la tendance (trend)

Nous supposons que la série temporelle étudiée soit la réalisation d'un processus stochastique composé d'une tendance déterministe m_t et d'une partie aléatoire ϵ_t (supposée de moyenne nulle) :

$$X_t = m_t + \epsilon_t.$$

Une méthode simple consiste à supposer que cette tendance est linéaire :

$$m_t = a + bt,$$

et d'estimer les paramètres a et b par *moindres carrés.*
Ainsi, si on observe la série $x_1, \ldots, x_n$, il faut trouver a et$b$ qui minimisent la quantité :

$$\sum_{t=1}^{n} (x_t - a - bt)^2$$

Les solutions de ce problème sont :

$$\hat{a} = \frac{6}{n(n-1)} \left(-\sum_{t=1}^{n} t x_t + \frac{2n+1}{3} n\bar{x} \right),$$

$$\hat{b} = \frac{12}{n(n^2-1)} \left(\sum_{t=1}^{n} t x_t + \frac{n+1}{2} n\bar{x} \right).$$

L'hypothèse de linéarité de la tendance convient très bien à certaines séries temporelles *(voir la Figure(1.1))*. Mais ce n'est pas le cas de toutes les séries *(voir la Figure(1.2))*. Il est alors possible de supposer que la tendance soit de forme polynomiale :

$$m_t = a + bt + ct^2$$

et d'estimer les paramètres a, b et c par moindres carrés. Mais il est parfois difficile d'estimer le degré du polynôme, et lorsque le degré est trop important, le nombre de paramètres à estimer devient grand et les calculs fastidieux. Dans cette situation, on a recourt à une méthode d'estimation non paramétrique [Jacques (2013)].

1.2.5 Estimation non paramétrique : moyenne mobile

Tendance : Supposons que la tendance m_t soit linéaire dans un petit intervalle $[t-q, t+q]$ autour de t. Dans ce cas, un bon estimateur de la tendance est *la moyenne* sur cet intervalle :

$$\hat{m}_t = \frac{1}{2q+1} \sum_{k=-q}^{q} x_{t+k}\,.$$

On peut donc estimer la tendance à chaque temps t en calculant la moyenne sur les observations étant dans une fenêtre de largeur $2q+1$ autour de t : c'est ce que l'on appelle une estimation par *moyenne mobile* :

Pour éviter les problèmes de bord, on suppose que $x_t = x_1 \; si \; t < 1 \; et \, x_t = x_n \; si \, t > n$.

Tendance et saisonnalité. Supposons désormais que le processus ne comporte pas uniquement une tendance, mais également une saisonnalité :

$$X_t = m_t + s_t + \epsilon_t,$$

avec s_t une fonction T *-périodique*. Le principe d'estimation est (en simplifiant légèrement) le suivant : On estime la tendance moyenne sur une période, puis on estime la composante saisonnière en moyennant sur toutes les périodes les écarts à la tendance moyenne de la période.

Application : Nous faisons une décomposition du trafic mensuel en tendance, saisonnalité et erreur . La Figure(1.3) donne de haut en bas : le trafic mensuel brut, la tendance, la composante saisonnière et l'erreur. Septembre 2001 est marqué par un trait vertical. Examinons ces chronogrammes . Le premier montre la croissance du trafic et sa saisonnalité prononcée qui rend peu lisible l'effet du 9/11. Par contre, le graphique de la tendance met en évidence la chute de trafic due au 9/11, puis la reprise progressive de la croissance par la suite [Yves (2011)].

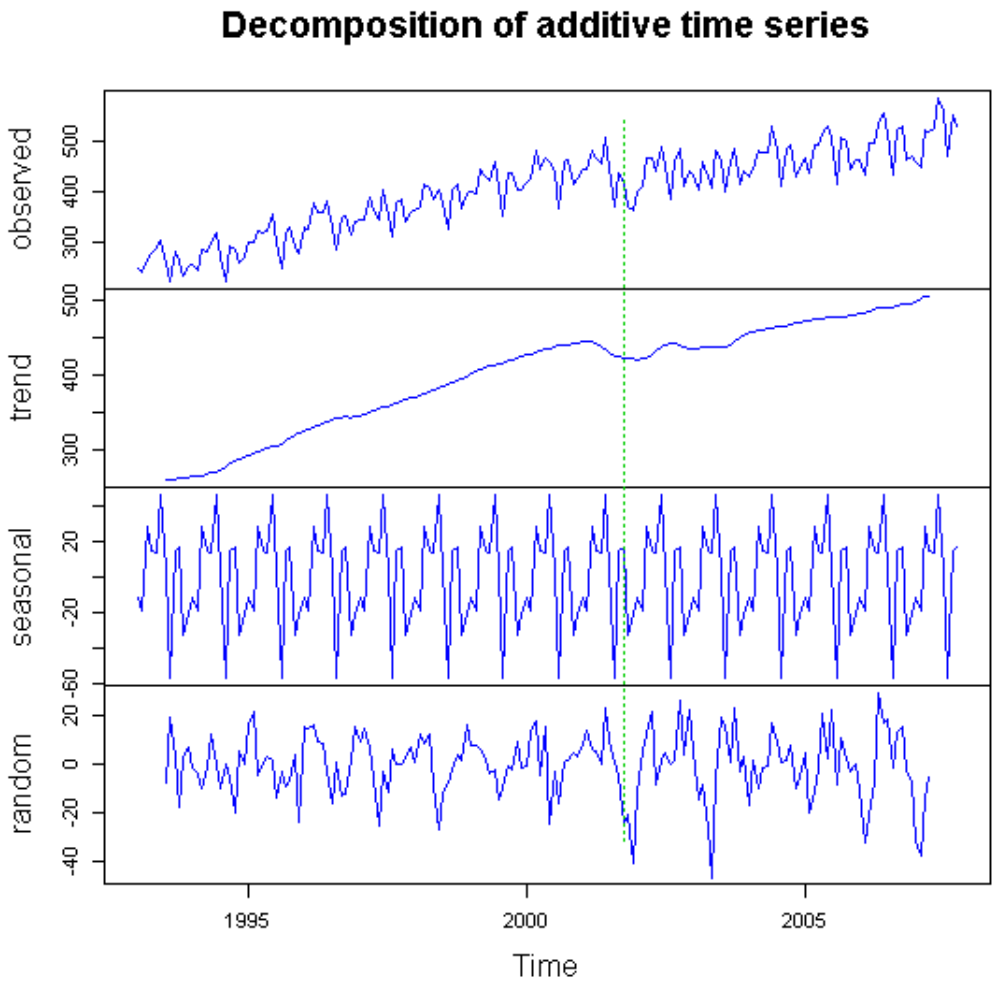

fig. 1.3 – Trafic passager mensuel et sa décomposition additive, unité 10^3 passagers.

1.2.6 Elimination de la tendance et de la saisonnalité par la méthode des différences

Cette méthode permet de supprimer les tendances et saisonnalité d'une série temporelle sans les estimer.
Soit Δ_T l'opérateur qui associe $(X_t - X_{t-T})$ à (Xt) :

$$\Delta_T = X_t - X_{t-T}.$$

On note Δ l'opérateur Δ_1 , et Δ_T^k l'opérateur $\underbrace{\Delta_T \circ \cdots \circ \Delta_T}_{k\ fois}$.

Proposition 1.2.6. *Soit un processus admettant une tendance polynomiale d'ordre k :*

$$X_t = \underbrace{\sum_{j=0}^{k} a_j t^j}_{m_t} + \varepsilon_t \,.$$

Le processus ΔX_t admet une tendance polynomiale d'ordre k-1.

Ainsi, en appliquant k fois Δ, on élimine la tendance.

1.2.7 Test sur la série résiduelle

L'objectif des techniques présentées est d'obtenir une série stationnaire (ou tout au moins le plus stationnaire possible), et en particulier sans tendance ou saisonnalité. L'étape suivante consiste à modéliser la série résiduelle obtenue. La première chose à faire est donc de tester s'il y a dépendance entre les termes de cette série. Si ce n'est pas le cas, on dit que la série résiduelle (stationnaire) est un bruit blanc (définition1.3).

Comment tester si on est en présence d'un bruit blanc ?

Par l'étude de la fonction d'auto-corrélation empirique. Lorsque n est assez grand, les *auto-corrélations d'un bruit blanc* sont *approximativement indépendantes* et de loi $\mathcal{N}\left(0\,;\frac{1}{n}\right)$. Ainsi, 95% des *auto-corrélations* devraient se trouver dans l'intervalle $\left[\frac{-1.96}{n}, \frac{1.96}{n}\right]$, et en traçant les 40 premières auto-corrélations il ne devrait pas y en avoir plus de 3 en dehors de ces limites.

A l'aide du test du portemanteau Plutôt que de regarder si chaque *auto-corrélation* est dans les bornes de l'intervalle précédent, nous considérons la statistique définie par la somme des h premières *auto-corrélations au carré*

$$Q = n \sum_{j=1}^{h} \hat{\rho}^2\,(j) \ .$$

La statistique Q suit une loi du χ^2 à h degrés de liberté. Il est donc possible de construire un test qui consistera à rejeter l'hypothèse nulle (la série est un bruit blanc) si Q est supérieur au quantile $\chi^2_{h,1-\alpha}$.

Ljung et Box (1978) ont amélioré ce test en considérant la statistique [2]

$$Q_{LB} = n(n+2)\sum_{j=0}^{h}\frac{\hat{\rho}^2(j)}{n-j} \ ,$$

dont la distribution est mieux approximée que la précédente par une loi du χ^2 à h degrés de liberté

1.3 Modélisation des séries stationnaires

Après avoir examiné des outils d'exploration, nous abordons la modélisation des séries temporelles. On considère qu'une série temporelle observée $\{y_t$, t =1,..., T$\}$ est la réalisation de v.a. (variables aléatoires)$\{Yt$, t =1,..., T$\}$ qui forment elles-mêmes une portion d'un processus aléatoire $\{Y_t$, t =..., 0,1,2,...$\}$, c'est-à -dire *d'une série infinie de v.a.* Pour pouvoir faire de l'estimation (de moyenne, de variance,...), il faut que le phénomène ait une certaine stabilité. La notion de stationnarité, est la clef de l'analyse des séries temporelles [Argon (2009)].

1.3.1 Corrélation, fonction d'auto corrélation et fonction d'auto corrélation partielle

Fonction d'auto corrélation (ACF). Considérons une série (faiblement) stationnaire X_t. On est souvent intéressé par décrire la dépendance de X_t par rapport à son passé, notamment pour expliquer le niveau actuel de la série par le niveau à une date précédente. Si une dépendance est linéaire, elle est bien décrite par *le coefficient d'auto corrélation.* Par définition, le coefficient d'auto corrélation d'ordre l est :

$$\rho_l = \frac{Cov(X_t, X_{t-l})}{\sqrt{V(X_t)\,V(X_{t-l})}}$$

Mais, $V(X_{t-l}) = V(X_t) = \gamma_0$ et $E(X_t) = \mu$ donc :

$$\rho_l = \frac{Cov(X_t, X_{t-l})}{V(X_t)} = \frac{E[(X_t-\mu)(X_{t-l}-\mu)]}{E(X_t-\mu)^2} = \frac{\gamma_l}{\gamma_0}$$

Fonction d'auto-corrélation partielle(ACPF) de X_t, est définie par :

$$\begin{cases} \alpha(1) = corr(X_2, X_1) = \rho(1) \\ \alpha(k) = corr\left(X_{k+1} - P_{\overline{sp}\{1,X_2,\ldots,X_k\}}X_{k+1}, X_1 - P_{\overline{sp}\{1,X_2,\ldots,X_k\}}X_1\right) \end{cases}$$

où $P_{\overline{sp}\{1,X_2,\ldots,X_k\}}$ est la projection sur le sous-espace engendré par $1, X_2, \ldots, X_k$.
$\alpha(k) = \phi_{kk}$, $k \geq 1$ avec ϕ_{kk} tel que :

$$\begin{pmatrix} \rho_0 & \rho_1 & \cdots & \rho_{k-1} \\ \rho_1 & \rho_2 & \cdots & \rho_{k-2} \\ \vdots & \cdots & \ddots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \cdots & \rho_0 \end{pmatrix} \begin{pmatrix} \phi_{k1} \\ \phi_{k2} \\ \vdots \\ \phi_{kk} \end{pmatrix} = \begin{pmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_k \end{pmatrix}$$

2. C'est ce test qui est implémenté dans la fonction Box.test de R.

1.3.2 Les processus auto-régressifs $AR(p)$

Les processus auto-régressifs, construits à partir de l'idée que l'observation au temps t s'explique linéairement par les observations précédentes.

Définition 1.3.1. *On dit que (X_t) est un processus auto-régressif d'ordre p (centré) s'il s'écrit*

$$X_t = \varepsilon_t + \sum_{j=1}^{p} a_j X_{t-j}\,, \tag{1.1}$$

où ε_t est un bruit blanc centré de variance σ_2.

ε_t est l'**innovation** contenue dans le processus au temps t
Les coefficients a_j doivent vérifier la contrainte suivante pour assurer la stationnarité du processus :
le polynôme caractéristique du processus (1.1), $A(z) = 1 - a_1 z - ... - a_p z^p$, ne doit avoir que des racines (réelles ou complexes)de module strictement supérieur à 1.

Proposition 1.3.1. *La variance du processus $AR(p)$ défini en (1.1) est :*

$$\sigma(0) = \sigma^2 + \sum_{j=1}^{p} \phi_j \sigma(j)$$

et l'auto-covariance, pour tout $h > 0$:

$$\sigma(h) = \sum_{j=1}^{p} a_j \sigma(h-j)\,.$$

On note aussi $\Phi(L)X_t = \epsilon_t$

Proposition 1.3.2. *Si $X_t \rightsquigarrow AR(p)$ tel que , $\phi(L) X_t = \mu + \varepsilon_t$ alors*

$$E(X_t) = \frac{\mu}{\phi(1)} = \frac{\mu}{1 - (\varphi_1 + \cdots + \varphi_p)}$$

Démonstration.

$$X_t = \mu + \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + \varepsilon_t$$
$$E(X_t) = \mu + \varphi_1 E(X_{t-1}) + \cdots + \varphi_p E(X_{t-p}) + E(\varepsilon_t)$$

$$\begin{aligned} m &= \mu + \varphi_1 m + \cdots + \varphi_p m \\ &= \frac{\mu}{1 - (\varphi_1 + \cdots + \varphi_p)} = \frac{\mu}{\phi(1)} \end{aligned}$$

□

Proposition 1.3.3. *$X_t \rightsquigarrow AR(p)$ est tel que $\phi(L) X_t = \mu + \varepsilon_t$ et si l'on pose $Y_t = X_t - m$ où $m = E(X_t)$, on a alors $\phi(L) Y_t = \varepsilon_t$ et $E(Y_t) = 0$.*

Démonstration. $m = E(X_t) = \frac{\mu}{\phi(1)}$ et $\phi(L)(X_t - m) = \phi(L) X_t - \phi(L) m$ or $Lm = m$ et par conséquent :

$$\phi(L) m = (1 - (\varphi_1 + \cdots + \varphi_p)) m = \phi(1) m$$

Finalement :

$$\phi(L)(X_t - m) = \phi(L) X_t - \mu = \varepsilon_t$$

□

Ecriture $MA(\infty)$quand les racines de ϕ sont de module strictement supérieur à 1

On suppose que $\phi(L) X_t = \mu + \varepsilon_t$ où $\phi(L) = 1 - (\varphi_1 L + \cdots + \varphi_p L^p)$ et aussi que $|z| \leq 1 \Rightarrow \phi(z) \neq 0$. On suppose que $\phi(z) = \prod_1^p (1 - \lambda_i z)$ où $|\lambda_i| = \dfrac{1}{|z_i|} < 1$, alors $\phi(L)$ est inversible et $\phi(L)^{-1} = \sum\limits_{k=0}^{\infty} a_k L^k = A(L)$ tel que $\sum |a_k| < \infty$ et $a_0 = 1$.

On en déduit

$$\begin{aligned} X_t &= A(L)\mu + A(L)\epsilon_t \\ &= A(1)\mu + (\sum_0^\infty a_k L^k)\epsilon_t \\ &= m + \sum_0^\infty a_k L^k \epsilon_{t-k} \end{aligned}$$

Car $\phi(1)^{-1}\mu = m$.

Proposition 1.3.4. *sous les hypothèses précédentes, $(X_t)_{t\in\mathbb{Z}}$ admet une représentation $MA(\infty)$ i.e.*

$$X_t = m + \sum_{k=0}^{\infty} a_k \varepsilon_{t-k},\ \text{où}\ a_0 \in \mathbb{R},\ \sum_{k=0}^{\infty} |a_k| < +\infty$$

Proposition 1.3.5. *Si $X_t \rightsquigarrow AR(p)$ alors les $|\rho(h)|$ et les $\gamma(h)$ décroissent vers 0 exponentiellement avec h.*

Démonstration.

$$\forall h > 0,\ \rho(h) - \varphi_1 \rho(h-1) - \cdots - \varphi_p \rho(h-p) = 0.$$

Le polynôme caractéristique de cette relation de récurrences est :

$$z^p - \varphi_1 z^{p-1} - \cdots - \varphi_{p-1} z - \varphi_p = z^p \left(1 - \frac{\varphi_1}{z} - \cdots - \frac{\varphi_{p-1}}{z^{p-1}} - \frac{\varphi_p}{z^p}\right) = z^p \phi\left(\frac{1}{z}\right)$$

Avec $\phi(L) X_t = \varepsilon_t$ et $\phi(L) = 1 - \varphi_1 L - \cdots - \varphi_p L^p$. Les racines du polynôme caractéristique sont les $\lambda_i = \frac{1}{z_i}$(les z_i sont les racines de ϕ) avec $|\lambda_i| < 1$.

La forme générale de la solution est, si $z_1, \cdots, z_n$ sont les racines distinctes de ϕ de multiplicité respective $m_1, \cdots, m_n$:

$$\rho(h) = \sum_{i=1}^{n} \sum_{k=0}^{m_i - 1} \alpha_{ik} \lambda_i^k h^k$$

$|\lambda_i| < 1$ donc $\rho(h)$ décroît vers 0 exponentiellement avec h. □

Proposition 1.3.6. *Si $X_t \rightsquigarrow AR(p)$et si $\phi(L) X_l = \mu + \varepsilon_t$ est sa représentation canonique, alors : (auto-corrélation partielle d'ordre p)*

$$\tau(h) = \begin{cases} 0 & si\ p > h \\ \neq 0 & si\ non \end{cases}$$

Démonstration. r(h) est le coefficient de X_{t-h} dans $EL\left(X_t|X_{t-1},\cdots,X_{t-h}\right)$ et

$$X_t = \mu + \underbrace{\varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p}}_{\in \mathcal{L}(1,X_t,\cdots,X_{t-p}) \subset \mathcal{L}(1,X_t,\cdots,X_{t-h})} + \varepsilon_t$$

donc

$$\begin{aligned} EL\left(X_t|X_{t-1},\cdots,X_{t-h}\right) &= \mu + \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + EL\left(\varepsilon_t|X_{t-1},\cdots,X_{t-h}\right) \\ &= \mu + \varphi_1 X_{t-1} + \cdots + \varphi_p X_{t-p} + 0 \end{aligned}$$

— Si $h > p$, le coefficient de X_{t-h} est 0.
— Si $h = p$, le coefficient de X_{t-p} est $\varphi_p \neq 0$.

□

Proposition 1.3.7. *Si $X_t \rightsquigarrow AR(p)$, alors :*

$$\gamma(h) = \begin{cases} 0 & si\ |h| > p \\ \neq 0 & si\ |h| = p \end{cases} \quad et \quad \rho(h) = \begin{cases} 0 & si\ |h| > p \\ \neq 0 & si\ |h| = p \end{cases}$$

Démonstration. $\rho_i(h) = \frac{\gamma_i(h)}{\gamma_i(0)}$ où :

$$\gamma_i(h) = \int_{-\pi}^{\pi} \frac{1}{f_X(\omega)} e^{i\omega h} d\omega$$

en remplaçant X_t par $X_t - m$;

$$f_X(\omega)\left|\phi\left(e^{i\omega}\right)\right|^2 = f_\varepsilon(\omega) = \frac{\sigma_\varepsilon^2}{2\pi} \Rightarrow f_X(\omega) = \frac{\sigma_\varepsilon^2}{2\pi} \frac{1}{\left|\phi\left(e^{i\omega}\right)\right|^2}$$

Et par conséquent :

$$\frac{1}{f_X(\omega)} = \frac{2\pi}{\sigma_\varepsilon^2}\left|\phi\left(e^{i\omega}\right)\right|^2$$

$\phi(z) = 1 - \varphi_1 z - \cdots - \varphi_p z^p = \sum_{k=0}^{p} \psi_k z^k$, avec $\psi_0 = 1$ et $\psi_k = -\varphi_k$, $k > 0$,

$$\frac{1}{f_X(\omega)} = \frac{2\pi}{\sigma_\varepsilon^2}\left(\sum_{k=0}^{p} \psi_k e^{i\omega k}\right)^2 = \frac{2\pi}{\sigma_\varepsilon^2} \sum_{0 \leq k,l \leq p} \psi_k \psi_l e^{i\omega(k-l)}$$

Ainsi

$$\gamma_i(h) = \frac{2\pi}{\sigma_\varepsilon^2} \sum_{0 \leq k,l \leq p} \psi_k \psi_l \underbrace{\int_{-\pi}^{\pi} e^{i\omega(k-l+h)} d\omega}_{=0,\, sauf\ si\ k-l+h=0}$$

Or $(k-l) \in [-p,p]$ donc si $h > p$, $\gamma_i(h) = 0$. En revanche si $h = p$:

$$\int_{-\pi}^{\pi} e^{i\omega(k-l+h)} d\omega \Leftrightarrow p = l - k \Leftrightarrow \begin{cases} l = p \\ k = 0 \end{cases}$$

Donc

$$\gamma_i(p) = \frac{4\pi^2}{\sigma_\varepsilon^2} \psi_0 \psi_p = -\frac{4\pi^2}{\sigma_\varepsilon^2} \varphi_p \neq 0$$

□

1.3.3 Les processus en moyenne mobile $MA(q)$

Définition 1.3.2. *On appel* ***le moyenne mobile****(Moving Average) d'ordre q un processus de la forme*

$$X_t = m + \epsilon_t - \theta_1\epsilon_{t-1} - \cdots - \theta_q\epsilon_{t-q},$$

Où les ϵ_j pour $t-q \leq j \leq t$ sont des bruits blancs centrés de variance σ^2.

où $X_t = \theta(L)\epsilon_t$ On suppose bien évidement que q est inférieur au nombre d'observations.

Proposition 1.3.8.

$$E(X_t) = m$$

Remarque 1.5.

— $(X_t)_{t\in\mathbb{Z}}$ *est nécessairement stationnaire.*
— *On note $X_t = m + \theta(L)\varepsilon_t$ où $\theta(L) = 1 - \theta_1 L - \cdots - \theta_q L^q$.*
— *Comme $X_t - m = \theta(L)\varepsilon_t$, on peut centrer X_t (.i.e. $E(X_t) = 0$).*

Ecriture $AR(\infty)$ quand les racines de θ sont de module > 1

Sous ces hypothèses $\theta(L)$ est inversible et $\theta(L)^{-1} = \sum_{k=0}^{+\infty} a_k L^k$ avec $a_0 = 1$ et $\sum |a_k| < +\infty$. Il s'en suit que

$$X_t - m = \theta(L)\varepsilon_t \Leftrightarrow \theta(L)^{-1}(X_t - m) = \varepsilon_t \Leftrightarrow \theta(L)^{-1}X_t - \frac{m}{\theta(1)} = \varepsilon_t$$

Soit encore $\sum_{k=0}^{+\infty} a_k X_{t-k} - \mu = \varepsilon_t$ où $\mu = \dfrac{m}{\theta(1)}$. D'où la représentation canonique $AR(\infty)$

$$X_t = \sum_{k=1}^{+\infty} a_k X_{t-k} + \frac{m}{\theta(1)} + \varepsilon_t$$

Cas où des racines de θ sont de module < 1

On suppose qu'on s'est ramené à $X_t = \theta(L)\varepsilon_t$ par centrage :

$$X_t = \left[\prod_{i/|\lambda_i|<1}(1-\lambda_i L)\right]\left[\prod_{i/|\lambda_i|<1}(1-\lambda_i L)\right]\varepsilon_t$$

On définit :

$$\theta*(L) = \left[\prod_{i/|\lambda_i|<1}(1-\lambda_i L)\right]\left[\prod_{i/|\lambda_i|>1}\left(1-\frac{1}{\lambda_i}L\right)\right]$$

On définit aussi (η_t) par $X_t = \theta*(L)\eta_t$, d'où $\eta_t = \theta*(L)^{-1}X_t$. On montre que $f_\eta(\omega) = cte \Rightarrow (\eta_t) \rightsquigarrow \mathcal{BB}$. On a donc

$$\begin{cases} X_t = \theta^*(L)\eta_t \\ \text{toutes les racines de } \theta^* \text{ sont de module} > 1 \\ \eta_t \rightsquigarrow \mathcal{BB} \end{cases}$$

C'est la représentation canonique de (X_t) et (η_t) est le processus des innovations.

Propriétés des processus *MA(q)*

On suppose que la représentation étudiée est la représentation canonique

$$\begin{cases} X_t = m + \theta(L)\epsilon_t \\ \text{toutes les racines de } \theta \text{ sont de module} > 1 \\ \theta(L) = 1 - \theta_1 L - \cdots - \theta_q L^q \\ \epsilon_t \rightsquigarrow \mathcal{BB} \end{cases}$$

Proposition 1.3.9. *(Auto-covariance) Sous les hypothéses precedents*

$$\gamma(h) = \begin{cases} 0 & si\ \ |h| > q \\ -\theta_q \sigma_\epsilon^2 \neq 0 & si\ \ |h| = q \\ \sigma_\epsilon^2(-\theta_h + \sum_{i=h+1}^{q} \theta_i \theta_{i-h}) & si\ i \leq |h| < q \\ \sigma_\epsilon^2(1 + \sum_{i=1}^{q} \theta_i^2) & si\ h = 0 \end{cases}$$

On en déduit $\rho(h) = 0$ *si* $|h| > q$ *et* $\rho(q) \neq 0$.

Démonstration. $X_t = \varepsilon_t - \sum_{k=1}^{q} \theta_k \varepsilon_{t-k}$ après centrage.

— Si $h = 0$

$$\gamma(0) = Var(X_t) = \sigma_\varepsilon^2 \left(1 + \sum_{k=1}^{q} \theta_k^2\right) \neq 0$$

Car $\text{cov}(\varepsilon_{t-j}, \varepsilon_{t-k}) = 0$ *si* $j \neq k$.

— Si $h > q$

$$\gamma(h) = cov(X_t, X_{t-h})$$
$$= cov\left(\underbrace{(\varepsilon_t - \theta_1\varepsilon_{t-1} - \cdots - \theta_q\varepsilon_{t-q})}_{\varepsilon_{t-j},\, j\in[0,q]}, \underbrace{(\varepsilon_{t-h} - \theta_1\varepsilon_{t-h-1} - \cdots - \theta_q\varepsilon_{t-h-q})}_{\varepsilon_{t-k},\, k\in[h,q+h]}\right)$$

$\Rightarrow\ t - j \neq t - k \Rightarrow \gamma(h) = 0$

— Si $h = q$

$$\gamma(q) = \text{cov}\left[(\varepsilon_t - \theta_1\varepsilon_{t-1} - \cdots - \theta_q\varepsilon_{t-q}), (\varepsilon_{t-h} - \theta_1\varepsilon_{t-q-1} - \cdots - \theta_q\varepsilon_{t-2q})\right] = -\theta_q\sigma_\varepsilon^2$$

— Si $1 \leq |h| < q$

$$\gamma(h) = \text{cov}(\varepsilon_t - \theta_1\varepsilon_{t-1} - \cdots - \theta_q\varepsilon_{t-q}, \varepsilon_{t-h} - \theta_1\varepsilon_{t-q-1} - \cdots - \theta_q\varepsilon_{t-h-q})$$
$$= -\sum_{i=1}^{q} \theta_i \,\text{cov}\left[\varepsilon_{t-i}, \varepsilon_{t-h} - \sum_{k=1}^{q} \theta_k \varepsilon_{t-h-k}\right] \text{ car } \text{cov}(\varepsilon_t, \varepsilon_{t-i}) = 0\ \forall i > 0$$
$$= -\theta_h\sigma_\varepsilon^2 + \sum_{i=1}^{q}\sum_{k=1}^{q} \theta_i\theta_k \underbrace{\text{cov}(\varepsilon_{t-i}, \varepsilon_{t-h-k})}_{=0\, si\, i\neq h+k}$$
$$= -\theta_h\sigma_\varepsilon^2 + \left(\sum_{i=h+1}^{q} \theta_i\theta_{i-h}\right)\sigma_\varepsilon^2$$

□

Proposition 1.3.10. *$\rho_i(h)$ décroît exponentiellement avec h.*

Démonstration. $\rho_i(h) = \frac{\gamma_i(h)}{\gamma_i(0)}$ avec et $\gamma^i(h) = \int_{-\pi}^{\pi} \frac{1}{f_X(\omega)} e^{i\omega h} d\omega$ et

$$X_t = \theta(L)\varepsilon_t \Rightarrow f_X(\omega) = \frac{\sigma_\varepsilon^2}{2\pi}\left|\theta\left(e^{i\omega}\right)\right|^2.$$

D'où

$$\frac{1}{f_X(\omega)} = \frac{2\pi}{\sigma_\varepsilon^2 |\theta(e^{i\omega})|^2}$$

Soit $(Y_t)_{t\in\mathbb{Z}}$ un processus tel que $\theta(L) Y_t = \eta_t$ et $Y_t \rightsquigarrow AR(q)$:

$$\frac{\sigma_\eta^2}{2\pi} = f_Y(\omega)\left|\theta\left(e^{i\omega}\right)\right|^2 \Rightarrow f_Y(\omega) = \frac{\sigma_\eta^2}{2\pi}\frac{1}{|\theta(e^{i\omega})|^2}.$$

On a donc :

$$f_Y(\omega) = \frac{1}{f_Y(\omega)} \Leftrightarrow \frac{2\pi}{\sigma_\varepsilon^2} = \frac{\sigma_\eta^2}{2\pi} \Leftrightarrow \sigma_\eta^2 = \frac{4\pi^2}{\sigma_\varepsilon^2}$$

□

1.3.4 Les processus mixtes $ARMA(p, q)$

Définition 1.3.3. *Un processus stationnaire $(X_t)_{t\in\mathbb{Z}}$ admet une représentation ARMA(p, q) canonique minimale s'il vérifie une équation :*

$$\phi(L) X_t = \mu + \theta(L)\varepsilon_t \tag{1.2}$$

Où

1. $\varepsilon_t \rightsquigarrow BB\left(0, \sigma^2\right)$
2. $\phi(L) = 1 - \sum_{k=1}^{p} \varphi_k L^k$, *avec* $\varphi_p \neq 0$
3. $\theta(L) = 1 - \sum_{k=1}^{q} \theta_k L^k$, *avec* $\theta_q \neq 0$
4. *ϕ et θ ont toutes leurs racines de module strictement supérieur à 1 (représentation canonique).*
5. *ϕ et θ n'ont pas de racines communes (représentation minimale).*

Remarque 1.6.

a. *Il existe des solutions non stationnaires : soit (X_t)un processus stationnaire et (Y_t) déterministe tel que $\phi(L) Y = 0$. On définit $Z_t = Y_t - X_t$ qui vérifie l'équation(1.2).*

b. *Si ϕ et θ ont des racines de module strictement supérieur à 1 mais admettent une racine commune, alors*

$$\phi(L) = (1 - \lambda L)\varphi_0(L) \text{ et } \theta(L) = (1 - \lambda L)\theta_0(L)$$

D'où

$$\varphi_0(L) X_t = \frac{\mu}{1-\lambda} + \theta_0(L)\varepsilon_t \Rightarrow X_t \to ARMA(p-1, q-1)$$

Proposition 1.3.11.

1. $E(X_t) = \dfrac{\mu}{\phi(1)} = m$
2. $\phi(L)(X_t - m) = \theta(L)\varepsilon_t$

Démonstration.

1. On a

$$E(X_t - \varphi_1 X_{t-1} - \cdots - \varphi_p X_{p-1}) = E(\mu + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \cdots - \theta_q \varepsilon_{t-q})$$

   Et puisque (X_t) est stationnaire

$$m(1 - \varphi_1 - \cdots - \varphi_p) = \mu + 0 \Rightarrow m = \frac{\mu}{\phi(1)}$$

2. $\phi(L) X_t = \phi(1) m + \theta(L)\varepsilon_t = \phi(L) m + \theta(L)\varepsilon_t$, donc

$$\phi(L)(X_t - m) = \theta(L)\varepsilon_t$$

□

Proposition 1.3.12.

1. *Pour $h > q$, les $\gamma(h)$ et les $\rho(h)$ vérifient les équations de récurrence d'ordre :*

$$\gamma(h) = \sum_{k=1}^{p} \varphi_k \gamma(h-k) \text{ et} \rho(h) = \sum_{k=1}^{p} \varphi_k \rho(h-k)$$

2. *Elles décroissent donc vers 0 exponentiellement avec h, pour $h > q$.*

Démonstration.

1. $X_t = \sum_{k=1}^{p} \varphi_k X_{t-k} - \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$, d'où :

$$\begin{aligned}\gamma(h) = E(X_t, X_{t-h}) &= \sum_{k=1}^{p} \varphi_k E(X_{t-k}, X_{t-h}) - \sum_{i=1}^{q} \theta_i \underbrace{E(\varepsilon_{t-i}, X_{t-h})}_{=0} \\ &= \sum_{k=1}^{p} \varphi_k \gamma(h-k)\end{aligned}$$

   Par suit :

$$\rho(h) = \sum_{k=1}^{p} \varphi_k \rho(h-k)$$

2. Les $\gamma(h)$ et les $\rho(h)$ vérifient une équation de récurrence dont le polynôme caractéristique est $z^{p+1}\phi\left(\dfrac{1}{z}\right)$, les conditions initiales sont

$$\begin{cases}\gamma(q-k) \\ \rho(q-k),\end{cases} \quad 0 \leq k \leq p-1$$

□

Equations de Yule-Walker

L'équation précédente pour $q+1 \leq k \leq q+p$ donne :

$$\begin{pmatrix} \rho(q) & \cdots & \rho(q+p-1) \\ \vdots & \ddots & \vdots \\ \rho(q+p-1) & \cdots & \rho(q) \end{pmatrix} \begin{pmatrix} \varphi_1 \\ \vdots \\ \varphi_p \end{pmatrix} = \begin{pmatrix} \rho(p+1) \\ \vdots \\ \rho(p+q) \end{pmatrix}$$

Quand ρ est connu ou estimé, on peut alors calculer les ϕ_i (et inversement).

1.3.5 Les processus $ARIMA(p,d,q)$

Lorsque l'on a une série (X_t) à non stationnarité stochastique, il convient de la modéliser à l'aide d'un processus $ARIMA(p,d,q)$ où *d désigne l'ordre de différenciation (ou d'intégration).*

Définition 1.3.4. *Un processus $ARIMA(p,d,q)$ ou "**A**uto**r**egressive **I**ntegrated **M**oving **A**verage" d'ordre $p,d,$ et q pour la série (X_t) est un processus de la forme suivante :*

$$(1-\phi_1 B-\cdots-\phi_p B^p)\nabla^d x_t = (1-\theta_1 B-\cdots-\theta_q B^q)\varepsilon_t$$

ou encore

$$(1-\phi_1 B-\cdots-\phi_p B^p)(1-B)^d x_t = (1-\theta_1 B-\cdots-\theta_q B^q)\varepsilon_t$$

où ε_t est un bruit blanc centré de variance σ^2,B est l'opérateur de retard tel que $Bx_t = x_{t-1}$ et $B^p x_t = x_{t-p}$,∇^d est l'opérateur de différence de degré d ($d \geq 0$ est un entier positif), $(\phi_1,\cdots,\phi_p)$ et $(\theta_1,\cdots,\theta_q)$ sont des coefficients à estimer.

La série (X_t) est une série *non stationnaire* alors que la série $w_t = \nabla^d x_t$ est une série stationnaire. Estimer les paramètres du processus $ARIMA(p,d,q)$ pour la série (X_t) *non stationnaire* revient à estimer les coefficients du processus $ARMA(p,q)$ pour la série (W_t) *stationnaire.*

Détermination des paramètres p et q

Ayant transformé correctement ces données, le problème revient à trouver un modèle ARMA satisfaisant et particulièrement aussi à déterminer p et q à trouver les fonctions d'autocorrélation et les fonctions d'autocorrélation partielle.
L'identification se base principalement sur l'analyse des ACF et des $PACF$ des séries considérés . On peut distinguer les cas d'espèces suivants : Si on trouve que les ACF et les $PACF$ ne sont pas significatifs pour tous les retards, *alors les résidus obéissent à un processus de bruit blanc*[voir Lagnoux (1996)].

table 1.1 – Les propriétés des fonctions d'autocorrélation et d'autocorrélation partielle

Processus	ACF	**PACF**
$AR(1)$	Décroissance exponentielle si $\phi_1 > 0$ ou Décroissance Sinu soïdale amortie si $\phi_1 < 0$.	Pic significatif pour le premier retard : positif si $\phi_1 > 0$ et négatif si $\phi_1 < 0$. Les autres coeficients sont nuls pour des retards supérieurs à 1.
$AR(2)$	Décroissance exponentielle si $\phi_1 < 0$ et $\phi_2 > 0$ Décroissance Sinusoïdale amortie si $\phi_1 > 0$ et $\phi_2 . < 0$	Pic significatif pour le premier et le second retards , les autres coefficients sont nuls pour des retards supérieurs à 2 .
$AR(p)$	Décroissance exponentielle ou Décroissance sinusoïdale amortie selon le signe des ϕ_i	Pic significatif pour les " p " premiers retards , les autres coefficients sont nuls pour des retards supérieurs à p .
$MA(1)$	Pic significatif pour le premier retard : positif si $\theta_1 < 0$ et négatif si $\theta_1 > 0$ Les autres coeficients sont nuls pour des retards supérieurs à 1 .	Décroissance exponentielle si $\theta_1 > 0$ Décroissance Sinusoïdale amortie si $\theta_1 < 0$
$MA(2)$	Pic significatif pour le premier et le second retards , les autres coeficients sont nuls pour des retards supérieurs à 2.	Décroissance exponentielle si $\theta_1 > 0$ et $\theta_2 > 0$ Décroissance Sinusoïdale amortie si $\theta_1 > 0$ et $\theta_2 < 0$.
$ARMA(1,1)$	Décroissance géométrique à partir du premier retard, le signe est déterminé par $\phi_1 - \theta_1$.	Décroissance exponentielle si $\theta_1 > 0$ ou Décroissance Sinusoïdale amortie si $\theta_1 < 0$.
$ARMA(p,q)$	Décroissance exponentielle ou Décroissance sinusoïdale a amortie tronquée après $(q-p)$ retards.	Décroissance exponentielle ou Décroissance sinusoïdale amortie tronquée après $(q-p)$ retards.

1.4 La méthode de BOX-JENKINS

La méthode de prévision de **Box et Jenkins (1976)** est particulièrement bien adaptée au traitement de séries chronologiques complexes et à d'autres situations dans lesquelles la loi de base n'est pas immédiatement apparente. La puissance et l'attrait véritables de cette solution aux problèmes de prévision sont dus à son aptitude à manier des lois complexes. Cependant, comme elle traite des situations beaucoup plus compliquées, il est bien difficile de saisir les principes de cette technique, ainsi que les limites de son application. En outre, le coût de la méthode de Box-Jenkins, dans une situation donnée, est généralement bien supérieur à celui de toutes les autres méthodes quantitatives. Il faut dire tout de même que, pour cette dépense plus élevée, on obtient une précision bien meilleure . Elle se décompose en plusieurs étapes [Bourbonnais and Usunier (2013)] :

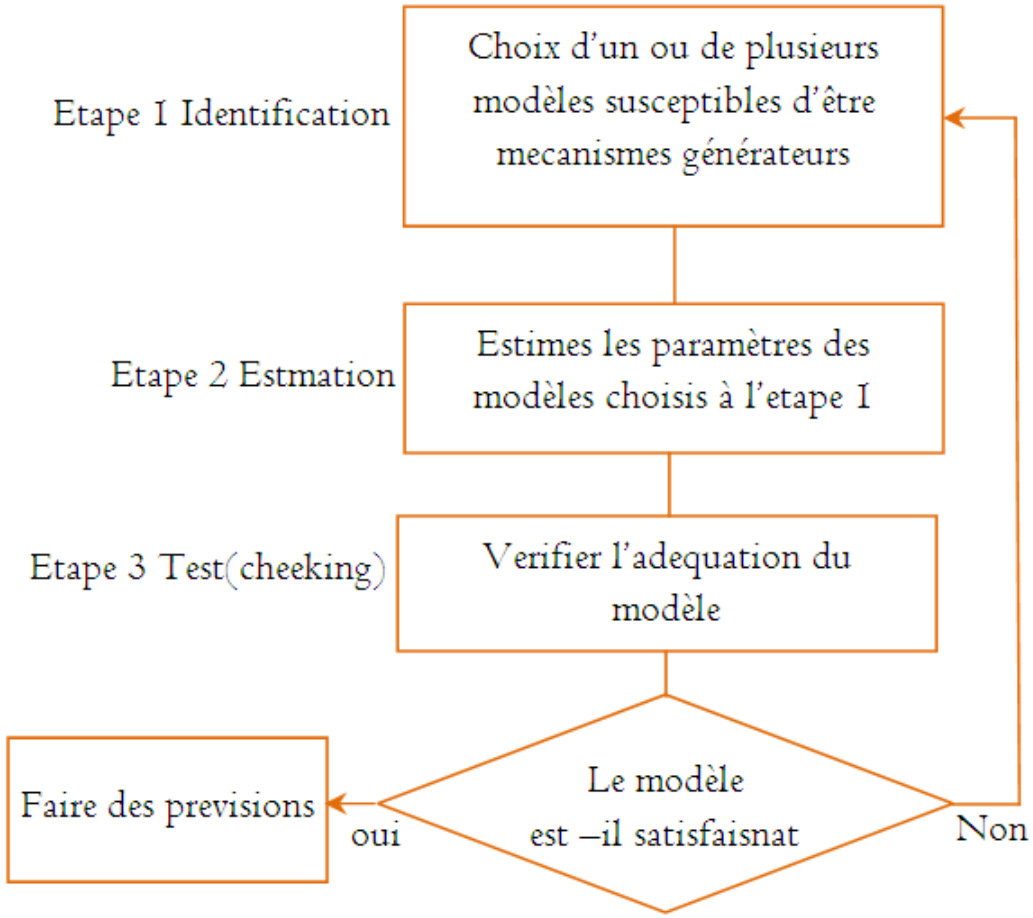

fig. 1.4 – Organigramme de fonctionnement de méthode de BOX et JENKINS

1.4.1 Estimation des paramètres du processus $ARMA(p, q)$

L'estimation des coefficients du processus $ARMA(p, q)$ s'effectue principalement à l'aide de *la méthode du maximum de vraisemblance*. On suppose pour cela que

$$\epsilon_t \rightsquigarrow \mathcal{N}(0, \sigma_\epsilon^2)$$

Méthode d'estimation du maximum de vraisemblance : La méthode du maximum de vraisemblance est couramment utilisée pour estimer les coefficients des modèles des séries temporelles car c'est une méthode simple à mettre en place pour estimer des modèles . Soit le modèle :

$$y_t = a_0 + a_1 x_t + \epsilon_t.$$

La fonction de densité de la loi normale de la variable y_t s'écrit :

$$f(y_t) = \frac{1}{\sigma_\varepsilon \sqrt{2\pi}} \exp\left[\frac{-(y_t - E(y_t))^2}{2{\sigma_\varepsilon}^2}\right] = \frac{1}{\sigma_\varepsilon \sqrt{2\pi}} \exp\left[\frac{-(y_t - a_0 - a_1 x_t)^2}{2{\sigma_\varepsilon}^2}\right]$$

La fonction de vraisemblance est donnée par :

$$f\left(y_1, \ldots, y_n; a_0, a_1, {\sigma_\varepsilon}^2\right) = \prod_{t=1}^{n} f\left(y_t; \alpha, \beta, {\sigma_\varepsilon}^2\right)$$

où $f(y_t;\alpha,\beta,{\sigma_\varepsilon}^2)$ est la fonction de densité de y_t.
On obtient alors :

$$f\left(y_1,\ldots,y_n;a_0,a_1,{\sigma_\varepsilon}^2\right)=\prod_{t=1}^{n}\frac{1}{\sigma_\varepsilon\sqrt{2\pi}}\exp\left[\frac{-(y_t-a_0-a_1x_t)^2}{2{\sigma_\varepsilon}^2}\right]$$

$$=\left(\frac{1}{\sigma_\varepsilon\sqrt{2\pi}}\right)^n\exp\left[\sum_{t=1}^{n}\frac{-(y_t-a_0-a_1x_t)^2}{2{\sigma_\varepsilon}^2}\right].$$

Il vient alors :

$$\ln f\left(y_1,\ldots,y_n;a_0,a_1,{\sigma_\varepsilon}^2\right)=n\ln\left(\frac{1}{\sigma_\varepsilon\sqrt{2\pi}}\right)-\left[\sum_{t=1}^{n}\frac{(y_t-a_0-a_1x_t)^2}{2{\sigma_\varepsilon}^2}\right]$$

Cette fonction est à *maximiser*. Les valeurs des coefficients qui permettent de maximiser la fonction sont issues des conditions du premier ordre suivantes :

$$\frac{\partial Lnf(\cdots)}{\partial a_0}=0\ ,\quad \frac{\partial Lnf(\cdots)}{\partial a_1}=0\ ,\quad \frac{\partial Lnf(\cdots)}{\partial \sigma_\varepsilon}=0\ .$$

1.4.2 Validation du processus $ARMA(p,q)$

Lors de la détermination des ordres *petq* du processus $ARMA(p,q)$ à l'aide des corrélogrammes simple et partiel, on peut être amené à sélectionner plusieurs ordres possibles *petq* pour le processus $ARMA(p,q)$. Après avoir estimé les différents processus $ARMA(p,q)$ possibles, il reste à les valider et à les départager. La validation des processus passe par un examen des coefficients estimés et par un examen des résidus.

Tests sur les coefficients

Parmi les processus $ARMA$ estimés, on ne retiendra que ceux dont tous les coefficients ont un de $t_{Student}>1,96$ (pour un risque de 5% et pour une taille d'échantillon suffisammentgrande : $T>30$).

1.5 Tests sur les résidus

1.5.1 Tests d'autocorrélation

Il existe un grand nombre de tests d'autocorrélation, les plus connus sont ceux de Box and Pierce (1970) et Ljung and Box (1978) .
Soit une autocorrélation des erreurs d'ordre $K(K>1)$:

$$\varepsilon_t=\sum_{i=1}^{K}\rho_i\varepsilon_{t-i}+\upsilon_t \text{ où } \upsilon_t\rightsquigarrow\mathcal{N}(0,{\sigma_\upsilon}^2)$$

Les hypothèses du test de Box-Pierce sont les suivantes :

$$\begin{cases} H_0: & \rho_1=\rho_2=\ldots=\rho_K=0\\ H_1: & \textit{il existe au moins un } \rho_i \textit{ significativement diffrent de } 0.\end{cases}$$

Pour effectuer ce test, on a recours à la statistique Q qui est donnée par :

$$Q=n\sum_{i=1}^{K}\hat{\rho}_i^2$$

Sous l'hypothèse H_0 vraie, Q suit l*a loi du Khi-Deux* avec K degrés de liberté :
$Q \rightsquigarrow \chi^2(K)$ si $Q > k^*$ où k^* est la valeur donnée par *la table du Khi-Deux* pour un risque fixé et un nombre K de degrés de liberté. Alors , on rejette H_0 et on accepte H_1 *(autocorrélation des erreurs).*

1.5.2 Tests de normalité

Pour calculer des intervalles de confiance prévisionnels et aussi pour effectuer les tests de Student sur les paramètres, il convient de vérifier la normalité des erreurs. Le test de Jarque et Bara [Bera et al. (1984)], fondé sur la notoin de Skewness(asymétrie) et de Kurtosis(aplatissement), permet de vérifier la normalité d'une distribution statistique.

Les tests du Skewness et du Kurtoris

Soit $\mu_k = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^k$ le moment centré d'ordre k, le coefficient de Skewness $\left(\beta_1^{1/2}\right)$est égal à : $\left({\beta_1}^{1/2}\right) = \frac{\mu_3}{{\mu_2}^{3/2}}$ et le coefficient de Kurtoris :$\beta_2 = \frac{\mu_4}{\mu_2^2}$. Si la distribution est normale et le nombre d'observation grand(n>30) : $\beta_1^{1/2} \rightarrow N\left(0; \sqrt{\frac{6}{n}}\right)$ *et* $\beta_2 \rightarrow N\left(3; \sqrt{\frac{24}{n}}\right)$ On construit alors les statistiques :

$$\nu_1 = \frac{\left|\beta_1^{1/2} - 0\right|}{\sqrt{\frac{6}{n}}} \quad et \quad \nu_1 = \frac{|\beta_2 - 3|}{\sqrt{\frac{24}{n}}}$$

que l'on compare à 1,96(valeur de la loi normale au seuil de 5%). Si les hypothèses $H_0 : \nu_1 = 0\, et\, \nu_2 = 0$ sont vérifiées, alors $\nu_1 \leq 1,96\ et\ \nu_2 \leq 1,96$; dans le cas contraire, l'hypothèse de normalité est rejetée.

Le test de Jarque et Bara

Il s'agit d'un test qui synthétise les résultats précédents ; si $\beta_1^{1/2}\, et\, \beta_2$ obéissent à des lois normales alors la quantité s : $s = \frac{n}{2}\beta_1 + \frac{n}{24}(\beta_2 - 3)^2$ suit un χ^2 à deux degrés de liberté.
Donc si $s > \chi^2_{1-\alpha}(2)$, on rejettel'hypothèse H_0 de normalité des résidus au seuil α.

Le Test de Kolmogorov-Smirnov

Le test de Kolmogorov-Smirnov est un test non paramétrique de conformité de deux populations régies par des variables quantitatives continues. Il est plus puissant que le test de χ^2 et ne nécessite pas de regroupement en classes ce qui le rend efficace dans le cas de petits échantillons. Le principe est de comparer les fréquences relatives cumulées (FRC) de l'échantillon E et celui, théorique, de la population P. La fonction descriminante est l'écart maximal observé (qui est aussi une distance entre deux probabilités) :

$$D_c = \max_{1 \leq i \leq n} |FRC_{obs}(x_i) - FRC_{th}(x_i)|$$

Les hypothèses relatives à un test bilatéral sont les suivantes :
$\begin{cases} H_0 : FRC_{obs} = FRC_{th} & pour\, tout\, x\, réel; \\ H_1 : FRC_{obs} \neq FRC_{th} & pour\, au\, moins\, une\, valeur\, de\, x\, réel. \end{cases}$

La distribution de la variable D est, sous l'hypothèse H_0 connue et tabulée.
Dès que $n > 40$, on peut considérer les valeurs seuils suivantes :

$$D_{5\%} = \frac{0.886}{\sqrt{n}} \quad D_{1\%} = \frac{1.031}{\sqrt{n}}$$

On rejetera H_0 dès que $D_c \leq D_s$.

1.5.3 Tests de stationnarité

Tests de Dickey-Fuller augmenté
Soit Y_t notre série initiale centrée. Le test de Dickey-Fuller augmenté(ADF,1981)consiste à effectuer la régression 1.3 et tester la signifficativité du coefficient $\rho - 1$ à l'aide des seuils de MacKinnon (1994) . $D_t = \mu + \alpha t$ est la partie déterministe de la série, $\alpha\, et\, \mu$ pouvant être nuls. Selon les valeurs de $\alpha\, et\, \mu$, on obtient les différentes versions du test ADF du tableau1.2

$$\Delta Y_t = D_t + (\rho - 1)Y_{t-1} + \sum_{i=1}^{p-1} \psi_i \Delta Y_{t-i} + \varepsilon_t \tag{1.3}$$

DICKEY-FULLER augmenté : test de racine unitaire
H_0 : $\rho - 1 = 0$ dans la regression1.3 . Racine unitaire, non stastionnarité.
H_1 : $\rho - 1 < 0$, stationnarité de la série.
statistique de test : statistique de STUDENT du coefficient $(\rho - 1)$.
rejet de H_0 : seuils de Mac KINNON, tableau1.2.

table 1.2 – Différentes versions du test DFA.

version du test		valeur critique à 5%
Tendance+ constante	$\alpha \neq 0, \mu \neq 0$	-3.454
Constante	$\alpha = 0, \mu \neq 0$	-2.89
ϕ	$\alpha = 0, \mu = 0$	-1.944

Remarque 1.7. *Il faut aussi tester la normalité du résidu (au moins en dessinant un histogramme). Sans cette normalité , les statistiques de test sur les coefficients n'ont pas les lois asymptotiques habituelles et les intervalles de confiance sur les coefficients, appuyés sur le théorème de la limite centrale, sont très approximatifs.*

Cependant, on montre qu'un modèle AR(p) choisi par le critère AIC permet une prédicition avec une plus petite erreur $\hat{o}_\varepsilon^2\,(p)$ (on dit que ce critèreest asymptotiquement éfficace : le rapport entre l'erreur de prédiction évaluée au minimum théorique et l'erreur se basant sur la valeur choisie par le critère AIC tend vers 1 si T tend vers l'infini),voir von Sachs and Van Bellegem (2005) et Venables and Smith (2009).

1.5.4 Critères de choix des modèles

Après examen des coefficients et des résidus, certains modèles sont écartés. Pour départager les modèles restants, on fait appel aux critères standards et aux critères d'information.
Critères standards :

- L'erreur absolue moyenne (Mean Absolute Error) :
$$MAE = \frac{1}{n}\sum_{t}|e_t|$$
où e_t est le résidu du modèle $ARMA$ étudié et n le nombre d'observation.
- l'erreur quadratique moyenne (Mean Squared Error) :
$$MSE = \frac{1}{n}\sum_{1}^{n} e_t^2.$$
- la racine carrée de l'erreur quadratique moyenne(Root Mean Square Error) :
$$RMSE = \sqrt{\frac{1}{n}\sum_{1}^{n} e_t^2}.$$
- Ecart absolu moyen en pourcentage (Mean Absolute Percent Error) :
$$MAPE = 100\frac{1}{n}\sum_{1}^{n}|\frac{e_t}{X_t}| \ .$$

Plus la valeur de ces critères est faible, plus le modèle estimé est proche des observations.

Critères d'information :

- Le critère d'Akaike : $AIC = \ln\sigma_\epsilon^2 + \frac{2(p+q)}{n}$
- Le critère de Schwarz : $SIC = \ln\sigma_\varepsilon^2 + (p+q)\frac{\ln(n)}{n}$
- Le critère d'information de Hannan - Quinn : $HQ = \ln\sigma_\varepsilon^2 + \alpha(p+q)\ln[\frac{\ln(n)}{n}]$

où $\alpha(>2)$ est une constante [Besse (2009)].

Conclusion

Rappelons que la méthode de BOX et JENKINS s'applique à la prévision à court terme. La raison en est que la supposition faite précédemment conduit, pour une série stationnaire, à une convergence des prévisions vers la moyenne arithmétique de la série car, si la valeur la plus probable du terme tenant compte de l'erreur résiduelle est nulle, on remarquera, en regardant l'expression générale des séries, qu'à la limite nous n'aurons que des termes autorégréssifs qui, tendent vers zéro, à mesure que l'ordre du processus augmente.

Sa facilité d'abord fait, de jour en jour, grossir le nombre de ses adeptes. Mais il faudrait peut être dire qu'aucune méthode statistique ne peut fournir des résultats de prévision plus précis que les données elles-mêmes. C'est pourquoi une attention soutenue devrait être accordée aux méthodes d'acquisitions des données en vue de leur traitement statistique.

Le prochain chapitre, aborde une nouvelle méthode pour estimer les paramètres du modèle. Cette méthode repose sur **les Réseaux de Neurones**.

CHAPITRE 2

Les réseaux de neurones artificiels

Introduction

DAns ce chapitre nous nous pencherons sur les méthodes les plus connues conçues pour la prévision des séries temporelles. Cette étude comportera des modèles et des algorithmes d'apprentissage provenant de différentes familles. On s'intéressa plus particulièrement aux réseaux de neurones artificiels. Nous verrons qu'il existe de nombreuses optimisations et adaptations de ces méthodes.

2.1 L'apprentissage artificiel

L'apprentissage artificiel(*Machine Learning : ML*), est une branche de l'intelligence artificielle qui peut être mise à contribution pour appréhender les problèmes de prédictions [Bishop (2006), Mitchell (1997)]. On peut classer les méthodes d'apprentissage artificiel en deux grandes familles : l'apprentissage non supervisé et l'apprentissage supervisé.

Dans l'apprentissage non supervisé, l'algorithme va lui-même, sans information supplémentaire, catégoriser les variables d'entrées. Ce type d'apprentissage permet l'élaboration d'une représentation interne de l'espace des données d'entrée en identifiant une structure statistique sous-jacente des variables sous une forme plus ou moins simple.
De façon générale, l'apprentissage artificiel supervisé consiste à créer un modèle de prédiction (*classification ou prévision*) à partir d'une base d'apprentissage comprenant les exemples d'entrée ainsi que les sorties désirés associées. Les paramètres du modèle vont ainsi s'adapter en comparant à chaque fois les sorties obtenues et les sorties désirées, d'où l'appellation supervisé [Mitchell (1997)].
Une fois le modèle obtenu par une base d'apprentissage, l'utilisation d'une base de test, comprenant des nouveaux exemples non utilisés pendant l'apprentissage, permet de mesurer les performances de la méthode. Une des possibilités est de calculer l'Erreur Quadratique Moyenne, en anglais Mean Square Error (*MSE*) (3.1).
L'algorithme d'apprentissage permet de "prédire" une valeur cible étant donnée une ou des valeurs d'entrées. Dans le cas où cette valeur cible est discrète (dans un ensemble

fini), la tâche réalisée par l'algorithme est appelée classification supervisée puisqu'il s'agit de trouver la classe correspondant à un exemple donné en entrée.
Soit $E = \left\{(X(t), y(t)) \in \mathbb{R}^d \times \mathbb{R} / 0 \leq t \leq N\right\}$ l'ensemble de données qui peut être une base d'apprentissage ou une base de test.$X(t)$ est le vecteur d'entrée à l'instant t, $X(t) \in \mathbb{R}^d$, $y(t)$ représente la valeur cible correspondant à $X(t)$ et N le nombre d'exemples dans la base. Le but de l'algorithme d'apprentissage est de trouver une fonction f, la représentation mathématique du modèle obtenu, qui soit le plus proche possible de la fonction cible $\tilde{f}$. Le système pourra être évalué à la fin par l'erreur MSE :

$$MSE = \frac{1}{N} \sum_{t=0}^{N} (\hat{y}(t) - y(t))^2 \tag{2.1}$$

Où $f(X(t)) = \hat{y}(t)$ l'approximation de la valeur $y(t)$. Pour répondre à ce problème, plusieurs méthodes ont été développées. On s'intéressera dans cette thèse à deux types de méthodes : les réseaux de neurones à propagation avant, les réseaux de neurones récurrents.

Les Réseaux de Neurones à Propagation Avant sont des réseaux à deux ou plusieurs couches. La caractéristique principale de ces réseaux est que l'information traverse dans un sens unique des entrées vers la sortie. Les Réseaux de Neurones Récurrents [Rumelheat (1986), Haykin (1999a) et Boné (2000)] sont, dans le cas général, des architectures sans couches (mais on peut tout à fait imaginer des architectures récurrentes assez proches des architectures à couches Boné (2000)) ; leur caractéristique principale est la possibilité d'avoir des boucles dans le graphe d'interconnexions.

2.2 Les Réseaux de Neurones Artificiels

Les réseaux de neurones artificiels, ANN (*Artificial Neural Networks*), constituent la première classe de méthodes qui nous intéressera dans cette thèse. Leur nom est dû aux neurones du cerveau humain. En effet, c'est la description des processus mentaux faite par les neurobiologistes qui est à l'origine des modèles théoriques des ANN. Ils sont aussi appelés réseaux connexionnistes ou réseaux neuromimétiques.

Deux éléments caractérisent ces méthodes, une organisation appelée réseau comprenant un certain nombre d'automates aux fonctionnalités relativement simples appelés *neurones*. L'information se propage dans les réseaux sur des connexions pondérées par des paramètres souvent appelés poids. Le deuxième élément est l'algorithme d'apprentissage dont l'objectif est de faire évoluer les poids du réseau de neurones de manière à obtenir un comportement global intéressant.
Dans cette section, nous passerons rapidement en revue les principes élémentaires des réseaux de neurones. Nous établirons aussi un lien entre les méthodes statistiques rencontrées (2.2) et ces méthodes connexionnistes.

$$x(t) = \sum_{i=1}^{p} w_i x(t-i) + \varepsilon(t) \tag{2.2}$$

2.2.1 Principes des Réseaux de Neurones Artificiels

Ce paragraphe présente le modèle de neurones le plus souvent utilisé et qui sera considéré dans le reste de ce travail. Mathématiquement un neurone peut être repré-

senté par une fonction à plusieurs variables avec une sortie unique. La **figure 2.1** représente le schéma classique d'un neurone formel. Les valeurs $x_1, \cdots, x_n$ représentent les entrées du neurone i. Les valeurs w_{ij} représentent les poids associés à chaque entrée x_j, $(1 \leq j \leq n)$.

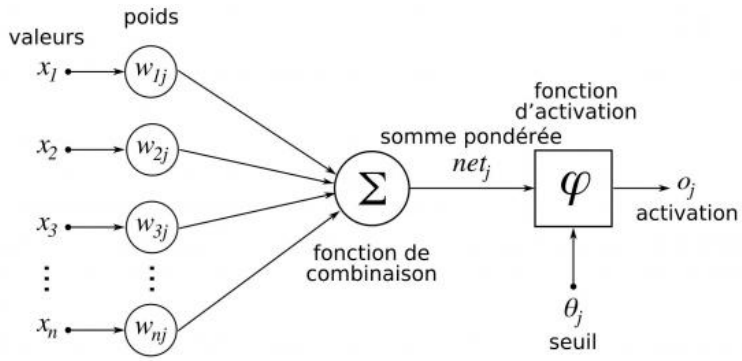

fig. 2.1 – architecture d'un neurone formel

Un neurone peut être également caractérisé par quatre aspects :

— La nature des entrées qui peut être soit binaire soit réelle ; dans notre cas ce sont toujours des réels.

— La fonction de sommation des entrées qui peut être linéaire ou non linéaire ; En pratique, ainsi que dans nos expérimentations, c'est une combinaison pondérée par les poids w_{ij} des entrées qui est utilisée ;

— La nature de la fonction de transfert utilisée (linéaire ou non-linéaire). En pratique, on trouve des fonctions à seuil (équation 4.2) ou des fonctions sigmoïdes (équation 4.3) ou encore des fonctions gaussiennes (équation 4.4) ;

— La nature de la sortie (binaire ou réelle). Dans notre cas, elle sera toujours réelle.

Fonction à seuil :

$$f(x) = \begin{cases} 1 & six \geq a \\ 0 & sinon \end{cases} \tag{2.3}$$

Fonction sigmoïde :

$$f(x) = \frac{1}{1 + e^{-ax}} \tag{2.4}$$

Fonction gaussienne :

$$f(x) = e^{\left(\frac{-ax^2}{2}\right)} \tag{2.5}$$

Nous pouvons remarquer que la fonction de transfert sigmoïde possède un régime quasi linéaire au voisinage de 0 et non linéaire ailleurs. Cette particularité offre au réseau la possibilité de s'adapter aussi bien aux problèmes linéaires qu'aux problèmes non linéaires.

2.2.2 Les équivalents connexionnistes des modèles statistiques

Il est possible de traduire les modèles statistiques dans leurs équivalents connexionnistes [Kouam (1993) et Connor et al. (1994)]. Les architectures ainsi obtenues sont appelées Réseaux à Propagation Avant(*Feed Forward Neural Networks* ; *FFNN*). Pour les modèles AR, les réseaux équivalents ressemblent à un réseau à propagation avant à deux couches, tandis que pour d'autres modèles comme le $ARMA$, les réseaux

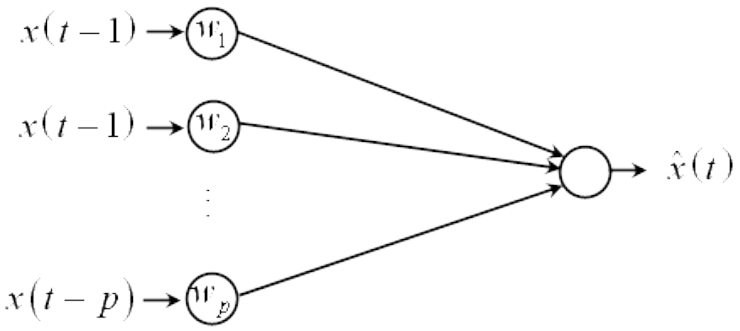

fig. 2.2 – architecture neuronale du le modèle $AR(p)$

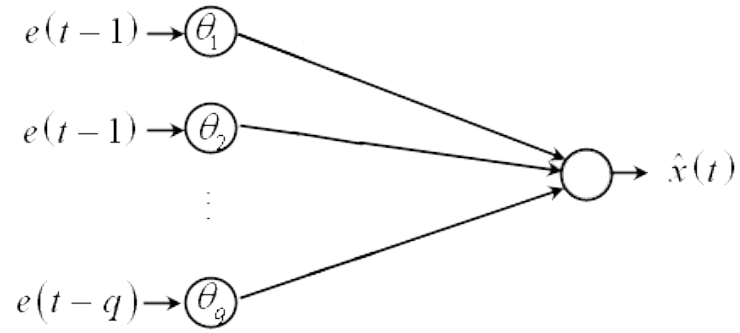

fig. 2.3 – architecture neuronale du le modèle $MA(q)$

équivalents sont dans la famille des réseaux récurrents. La figure (2.2) représente l'architecture neuronale équivalente au modèle $AR(p)$. La valeur prédite à l'instant $t-1$ est notée $\hat{x}(t)$ et elle est donnée par l'équation (4.5).

$$\hat{x}(t) = \sum_{i=1}^{p} w_i x(t-i) \tag{2.6}$$

L'erreur est définie par

$$e(t) = \hat{x}(t) - x(t) \tag{2.7}$$

Le modèle à moyenne mobile est défini par (voir la definition 1.3.2). En pratique, bien que $e(t) \neq \epsilon(t)$ puisque$e(t)$ est le reflet de la partie imprévisible de la série et de la limitation du modèle, les valeurs $\epsilon(t-i)$ sont remplacées par les résidus $e(t-1)$ pour estimer $\hat{x}(t+1)$ (équation 4.7).

$$\hat{x}(t+h) = \sum_{i=1}^{q} \theta(i) e(t-i) \tag{2.8}$$

Le RNA équivalent est présenté dans la figure (2.3). L'architecture connexionniste équivalente au modèle $ARMA$ ressemble aussi à une réunion des deux modèles précédents (voir figure 2.4), avec des entrées $e(t-i)$ placées par l'utilisateur et calculées avec $e(t) = \hat{x}(t) - x(t)$. D'après la figure 2.4, le réseau reçoit en entrée une fenêtre temporelle glissante qui contient les valeurs $\{x(t-1), ..., x(t-p), e(t-1), ..., e(q)\}$,tandis que le neurone de sortie fournit $\hat{x}(t)$. Un autre modèle connexionniste a été proposé dans [Back and Tsoi (1991)] et dans [Crucianu (1997)]. Ces réseaux reposent sur deux neurones utilisant des connexions *FIR* (*Finite Impulse Response*) ou *IIR* (*Infinite Impulse Response*). qui vont être décrites ci-après.

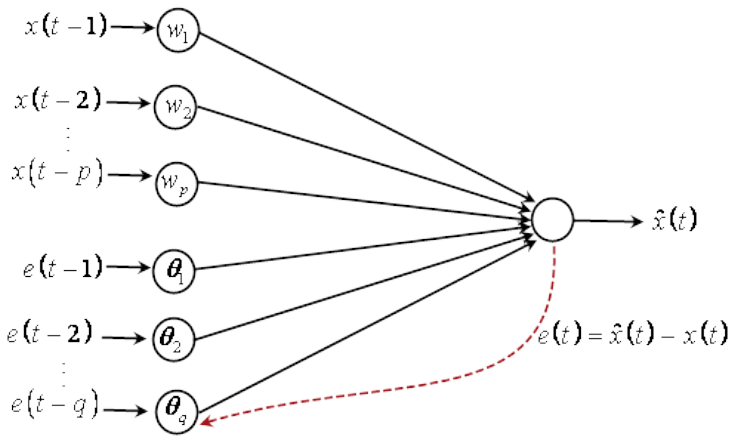

fig. 2.4 – architecture neuronale pour le modèle $ARMA(p, q)$

Les connexions Finite Impulse Response (*FIR*)

Une connexion *FIR* est une sorte de méta-connexion contenant des connexions associées intégrant des retards consécutifs. Il s'agit d'utiliser des blocs de retard ; un bloc de retard permet d'introduire un retard d'un pas de temps dans la propagation de l'information. Autrement dit, l'information ne se transmet pas aussitôt et met un pas de temps pour être transmise.
Ainsi une connexion *FIR* permet d'obtenir p retards(voir figure 2.5), à chaque retard correspond un poids $w_{ij}^{(t)}$, où r $(1 \leq r \leq p)$ est le degré du retard c'est-à-dire le nombre de blocs utilisés. Une connexion *FIR* permet de réaliser un modèle *AR* [Crucianu

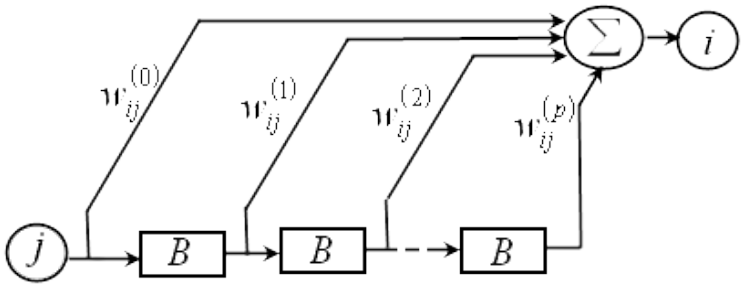

fig. 2.5 – Connexion *FIR* entre deux neurones

(1997)].

Les connexions Infinite Impulse Response (*IIR*)

Comme le montre la figure 2.6, Une connexion IIR est une connexion FIR à laquelle on ajoute à la sortie de la connexion une autre connexion *FIR* bouclée, elle permet de mettre en œuvre le modèle *ARMA* [Back and Tsoi (1991)].

2.3 Les Réseaux de Neurones à Propagation Avant

Dans ce paragraphe, nous détaillerons principalement deux types de réseaux à propagation avant : le Perceptron Multi Couche (*Multi-Layer Perceptron*), et les Réseaux à Fonction à Base Radiale(*Radial Basis Function Networks ; RBFN*).

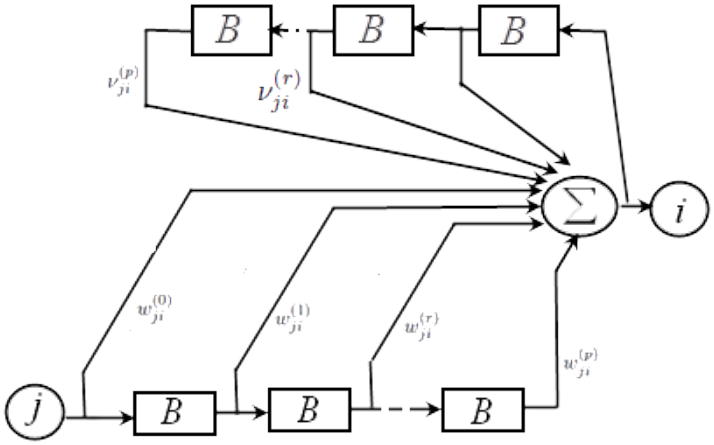

fig. 2.6 – Connexion IIR entre deux neurones

2.3.1 Le perceptron multi couche

Architecture de base

Dans ce type de réseaux (*MLP*), les neurones sont organisés de manière à constituer plusieurs couches. En pratique, le nombre de couches varie de 3 à 4 dont une couche d'entrée, une couche de sortie et une ou deux couches cachées.(figure 2.7).

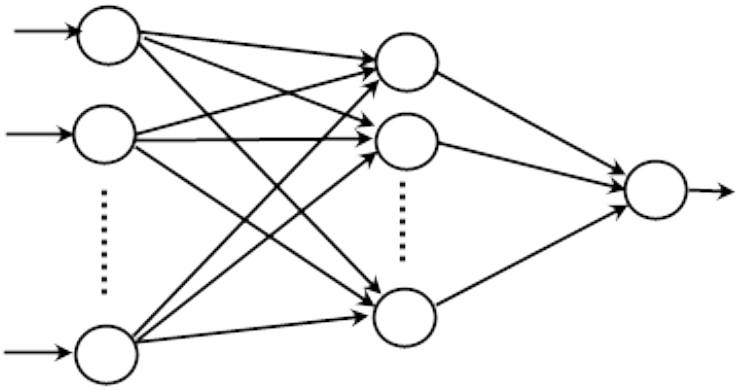

fig. 2.7 – Architecture générique d'un MLP.

La couche d'entrée contient autant de neurones que la dimension de l'espace d'entrée. La couche de sortie contient un neurone unique pour la valeur de la prévision. Enfin, une ou plusieurs couches cachées contiennent un nombre variable de neurones.

Caractéristiques de base

Du point de vue fonctionnel, tout comme les modèles linéaires, les *MLP* effectuent un certain nombre d'opérations algébriques. Grâce aux fonctions de transfert des neurones du réseau, les *MLP* réalisent une composition de fonctions sur les entrées. Ainsi, la sortie du réseau dépendra des entrées en cours et de la structure des connexions. La seule mémoire que possède un *MLP* est celle des exemples passés en entrée.
Si on note $(w_{ij})_{1\leq i\leq c, 1\leq j\leq m+1}$ les poids reliant les entrées $(x_j)_{1\leq i\leq m}$ et les unités

cachées $(u)_{1\leq i\leq c}$ et $(a_{ij})_{1\leq i\leq c,\leq j\leq c+1}$ les poids reliant les unités cachées et les sorties $(y_i)_{1\leq i\leq s}$, et si les unités cachées ont une fonction d'activation ϕ, le MLP représentera la fonction :

$$y_i = \sum_{j=1}^{c} a_{ij}\left(\phi\left(\sum_{k=1}^{m} w_{jk}x_k + w_{j0}\right)\right) + a_{i0},\ 1 \leq i \leq s \tag{2.9}$$

La possibilité d'approcher des fonctions par des MLP a déjà suscité une abondante littérature. Cybenko (1989), Hornik et al. (1989), ont étudié par exemple, l'approximation des fonctions continues à support compact par des MLP à une couche cachée, munis de fonction d'activation sigmoïdes. Ainsi on a le théorème de Hornik et al. (1989).

Théorème 1. *Soit un perceptron multicouches définis par l'équation (2.9), où ϕ est une fonction strictement croissante et bornée. Soit K un compact de $\mathbb{R}^m$. Alors , pour toute fonction f continue à support compact $(f \in C(K))$, $f : \mathbb{R}^m \to \mathbb{R}^s$ et pour $\varepsilon > 0$, il existe un entier C et un vecteur de paramètre*

$$w = \left((w_{ij})_{1\leq i\leq C,\,1\leq j\leq m+1},\,(a_{ij})_{1\leq i\leq s,\,1\leq j\leq C+1}\right) \in \mathbb{R}^{C\times(m+1)+s\times(C+1)}$$

tels que $\forall\ (x_1, \cdots, x_m) \in \mathbb{R}^m$,

$$\left\| f(x_1, \cdots, x_m) - \left(\sum_{j=1}^{C} a_{i(j+1)}\left(\phi\left(\sum_{k=1}^{m} w_{j(k+1)}x_k + w_{j1}\right)\right) + a_{i1}\right)_{1\leq i\leq s}\right\| < \varepsilon$$

Où $\|.\|$ est une norme de $\mathbb{R}^s$.

Ce théorème ne fournit pas de vitesse d'approximation. C'est pourquoi différents travaux ont suivi pour préciser la vitesse de convergence. Citons par exemple Barron (1993).

Théorème 2. *Soit $f : \mathbb{R}^m \to \mathbb{R}$, une fonction et $\tilde{f}$ sa transformation de Fourier. Posons $|w|_1 = \sum_{j=1}^{m} |w_j|$ la norme 1 de w sur $\mathbb{R}^m$. Soit un perceptron multicouches défini par l'équation(2.9), où ϕ est une fonction sigmoïde, et une seule sortie. Alors, si f vérifie*

$$\int_{\mathbb{R}^m} |w|_1 |f\ (w)| dw < \infty$$

l'erreur d'approximation de la fonction par un perceptron multicouches est bornée par $O\left(\frac{1}{\sqrt{C}}\right)$.

autrement dite, pour des réseaux de neurones à fonction d'activation sigmoïdale, l'erreur commise dans l'approximation varie comme l'inverse du nombre de neurones cachés, et elle est indépendante du nombre de variables de la fonction à approcher [Connor et al. (1994)].

Par conséquent, pour une précision donnée, donc pour un nombre de neurones cachés donné, la quantité de paramètres du réseau est proportionnelle au nombre de variables de cette fonction. Ce résultat s'applique aux réseaux de neurones à fonction d'activation sigmoïdale puisque la sortie de ces neurones n'est pas linéaire par rapports aux poids synaptiques.

Variantes du *MLP*

La première application d'un *MLP* pour la prévision de séries temporelles remonte à Lapedes and Farber (1987). Dans Weigend and Rumelhart (1992) des expérimentations sont réalisées sur la série des taches solaires et sur des données financières. Les *MLP* de base ont été appliqués pour diverses problématiques de prévision. Citons la consommation électrique Mori and Urano (1996), certains phénomènes naturels (prédiction de débit d'eau dans des rivières, décomposition journalière de sédiments) Atiya et al. (1999), Cigizoglu (2004).

On retrouve aussi des variantes des *MLP* combinant la puissance de prédiction des réseaux et la clarté de modélisation des méthodes *Box-Jenkins*. On trouve, par exemple, dans Zhang (2003) un *MLP* combiné avec le modèle statistique *ARIMA* Box et al. (2015). Le même principe est appliqué dans Aburto and Weber (2007) pour la prédiction des stocks des chaînes de production.

D'autres méthodes portent sur l'amélioration de l'algorithme d'apprentissage. Par exemple, avec des méthodes d'optimisation comme dans Wang and Lu (2006) par les *Particle Swarm Optimization* (*PSO*). Dans Cottrell et al. (1995), un algorithme est proposé pour les *MLP* qui consiste à détecter et à éliminer les poids peu significatifs.

2.3.2 Les Réseaux de Fonctions à Base Radiale

Les Réseaux de Fonctions à Base Radiale, en anglais *Radial Basis Function Network* (*RBFN*), sont des réseaux similaires aux *MLP* en termes d'architectures et d'algorithmes d'apprentissage. La différence réside dans l'utilisation des fonctions de transfert à base radiale (*gaussienne*).

Architecture de base

La figure 2.8 montre l'architecture générale d'un *RBFN*(*Radial Basis Function Network*). Dans les neurones de la couche cachée sont généralement utilisées des fonctions de transfert gaussiennes. Dans les neurones de la couche d'entrée et la couche de sortie, le plus fréquent est d'utiliser les fonctions de transfert linéaires.

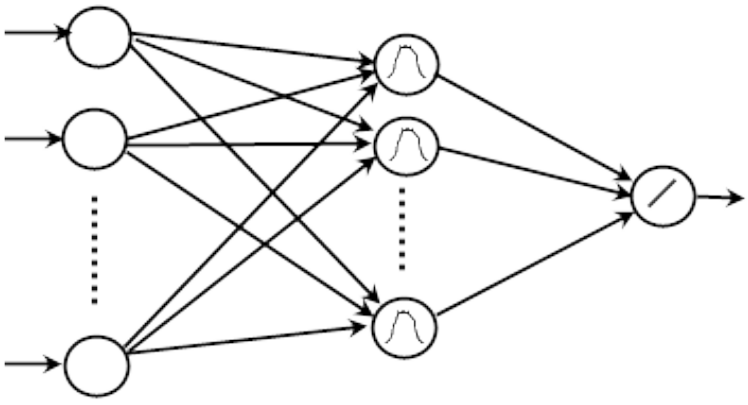

fig. 2.8 – Structure d'un réseau RBFN.

Caractéristiques de base

Tout comme les *MLP*, les *RBFN* possèdent eux aussi la propriété d'approximation universelle Haykin (1998). Les entrées sont propagées aux noyaux gaussiens ϕ_j avec

$j \in \{1, \cdots, m\}$, m étant le nombre de neurones cachés. Les valeurs du noyau sont ensuite pondérées par des coefficients λ_j afin de produire la valeur de sortie $\hat{y}(t)$, approximation de la sortie désirée $y(t)$:

$$\hat{y}(y) = \sum_{j=1}^{m} \lambda_j \phi_j \left(X(t), c_j, \mu_j\right) \tag{2.10}$$

Les noyaux gaussiens sont définis par :

$$\phi_j \left(X(t), c_j, \mu_j\right) = exp(-\frac{\|X(t) - c_j\|^2}{2\sigma_j}) \tag{2.11}$$

Les paramètres c_j et σ_j désignent respectivement le centre et la largeur de la fonction gaussienne. L'intérêt principal provient du fait que les paramètres peuvent être appris indépendamment, avec un apprentissage relativement rapide pour les λ_j comme solution d'un système d'équations linéaires.

On peut trouver les détails de l'apprentissage du *RBFN* dans Orr (1998) et Benoudjit and Verleysen (2003). En pratique, les paramètres c_j influencent de façon relativement limitée le résultat final, alors que les paramètres σ_j améliorent sensiblement l'approximation de la relation entrées-sortie. Théoriquement, il faut que les noyaux gaussiens soient assez larges et se recouvrent suffisamment pour que les parties de la distribution où aucun noyau n'est venu se placer soient prises en compte par le *RBFN*. En même temps, ils ne doivent pas être trop larges pour qu'ils ne recouvrent pas l'ensemble de la distribution. Les paramètres λ_j sont uniques, une fois que c_j et σ_j sont fixés. Leur influence sur la sortie est directe, il s'agit du poids utilisé dans l'équation 2.10.

Variantes du *RBFN*

On trouve dans la littérature de nombreuses applications directes des *RBFN* pour la prévision dans différents domaines Kassam and Cha (1993). Par ailleurs, il est tout à fait possible de combiner de tels réseaux avec les modèles statistiques de séries temporelles comme dans le cas des *MLP*. Dans Wedding II and Cios (1996), le *RBFN* donne en sortie la valeur de prédiction mais aussi une valeur de confiance. Les valeurs à faible confiance sont alors ignorées. Pour le reste, elles sont combinées avec les entrées dans un modèle de *Box et Jenkins* classique. L'algorithme d'apprentissage est parfois amélioré par des algorithmes d'optimisation. Dans Sheta and De Jong (2001), un algorithme génétique est utilisé.

2.3.3 Algorithme d'apprentissage : cas du *MLP*

L'algorithme d'apprentissage le plus utilisé pour les *FFNN* ainsi que leurs variantes est l'algorithme de *la Rétro-Propagation du gradient*(anglais *Back-Propagation BP*)Rumelhart et al. (1985). C'est un algorithme qui est souvent utilisé dans le cadre de nos expériences dont le principe est de rétro-propager le gradient de l'erreur de la couche de sortie vers l'entrée.

L'algorithme d'apprentissage peut s'appliquer de deux manières différentes : soit en considérant l'erreur instantanée $e(t)$, soit en considérant l'erreur globale $E(t_1, t_l)$

comprise entre les instants t_1 et t_l, avec :

$$E(t_1, t_l) = \sum_{t=t_1}^{t_l} e(t) \tag{2.12}$$

$e(t)$ est donnée généralement par la différence entre la sortie désirée et la sortie obtenue. La première méthode nous donne un apprentissage temps réel dans lequel la modification de poids se fait à chaque exemple présenté

$$\Delta w_{ij}(t) = -\eta \frac{\partial e(t)}{\partial w_{ij}} \tag{2.13}$$

Dans la deuxième méthode, l'algorithme fonctionne en mode différé, c'est-à-dire que les poids sont modifiés à la fin de chaque cycle (quand tous les exemples de la base d'apprentissage ont été présentés)

$$\Delta w_{ij}(t_1, t_l) = -\eta \frac{\partial E(t_1, t_l)}{\partial w_{ij}} \tag{2.14}$$

Prenons le cas de l'apprentissage d'un *MLP* avec des connexions à délais (*FIR*) d'ordre k avec $0 \leq k \leq p_i$. On suppose que l'apprentissage se fait avec l'algorithme *BP* en mode différé. On note $s_i(t)$ la sortie du neurone i,

$$s_i(t) = f_i(net_i(t)) \tag{2.15}$$

où f_i est la fonction de transfert du neurone i et net_i est la valeur de sommation des entrées du neurone i donnée par :

$$net_{ij}(t) = \sum_{k=0}^{p_i} w_{ij}^{(k)} s_j(t-k) \tag{2.16}$$

où k représente le degré du retard de la connexion entre deux neurones i et j. Nous appliquons la descente du gradient sur la fonction d'erreur définie par l'équation (2.12). La variation totale des poids w_{ij} est obtenue en appliquant une chaine de dérivation sur l'équation (2.14),

$$\frac{\partial E(t_1, t_i)}{\partial w_{ij}^{(k)}} = \sum_{t=t_1}^{t_l} \frac{\partial E(t_1, t_i)}{\partial s_i(t)} \frac{\partial s_i(t)}{\partial net_i(t)} \frac{\partial net_i(t)}{\partial w_{ij}^{(k)}} \tag{2.17}$$

On trouve facilement les deux derniers membres :

$$\frac{\partial s_i(t)}{\partial net_i(t)} = f'(net_i(t)) \tag{2.18}$$

$$\frac{\partial net_i(t)}{\partial w_{ij}^{(k)}} = s_j(t-k) \tag{2.19}$$

Le terme $\frac{\partial E(t_1, t_l)}{\partial s_i(t)}$ se calcule selon la position du neurone. Pour un neurone de la couche de sortie, il est obtenu directement en dérivant par rapport à $s_i(t)$,

$$\frac{\partial E(t_1, t_l)}{\partial s_i(t)} = -\left[s_i(t) - y(t)\right] \tag{2.20}$$

Pour les neurones de la couche cachée, on introduit le terme $\partial net_r(t')$ pour tous les neurones successeurs.

$$\frac{\partial E(t_1,t_l)}{\partial s_i(t)} = \sum_{r\in succ(i)} \sum_{t'=t_1}^{t_l} \frac{\partial E(t_1,t_l)}{\partial net_r(t')} \frac{\partial net_r(t')}{\partial s_i(t)} \tag{2.21}$$

où $succ(i)$ représente l'ensemble du neurones successeurs au neurone i. Or, $net_r(t)$ peut s'écrire

$$net_r(t) = \sum_{r\in succ(i)} \sum_{t'=t_1}^{t_l} w_{ri}^{(k)} s_i(t-k) \tag{2.22}$$

et par conséquent,

$$\frac{\partial net_r(t')}{\partial s_i(t)} = \begin{cases} w_{ri}^{(t'-t)} & si\, 0 \leqslant t'-t) \leq p_r \\ 0 & sinon \end{cases} \tag{2.23}$$

L'équation générale de la modification des poids est décrite alors par les équations (2.24) et (2.26)

$$\Delta w_{ij}^k = -\eta \sum_{t=t_1}^{t_l} \frac{\partial E(t_1,t_l)}{\partial s_i(t)} f'(net_i(t)) s_i(t-k) \tag{2.24}$$

$$\frac{\partial E(t_1,t_l)}{\partial s_i(t)} = \begin{cases} -(y_i(t)-s_i(t)) & si\ i\ est\ un\ neurone\ de\ sortie \\ \sum_{r\in succ(i)} \sum_{t'}^{t+p_r} \frac{\partial E(t_1,t_l)}{\partial net_r(t')} w_{ri}^{(t-t')} & si\ i\ est\ un\ neurone\ caché \end{cases} \tag{2.25}$$

2.3.4 Diverses problématiques liées aux *FFNN*

Dans Hoptroff (1993) et Moody (1995), des études générales ont dressé un bilan sur l'utilisation des réseaux à propagation avant pour des applications financières. D'autres travaux se sont attachés à caractériser les capacités de ces réseaux et les conditions qui peuvent influencer les performances. La nature des données est un des facteurs déterminants pour la performance finale du réseau de neurones.

La saisonnalité et le changement de régime sont d'autres aspects des données qui rendent la tâche de prédiction plus difficile. Le changement de régime a été étudié par González-Romera et al. (2008), alors que Zhang and Qi (2005) étudie l'efficacité des prétraitements des séries saisonnières sur la performance des *MLP*. Il conclut que même avec les prétraitements, les performances des *MLP* en prédiction restent limitées.

Le choix de l'architecture est un facteur tout aussi important. Il touche notamment le nombre de neurones d'entrée Mori and Urano (1996) ainsi que le nombre de neurones cachés Kaastra and Boyd (1996). dans Zhang et al. (2001) il est conclu que le nombre d'entrées est un facteur qui s'est montré plus déterminant sur les performances du réseau que le nombre de neurones cachés, et ce en testant sur un grand nombre de séries temporelles. Ce résultat n'est pas forcément en contradiction avec le théorème d'approximation universelle Cybenko (1989).

On remarque aussi que les performances des réseaux chutent lorsque l'on augmente l'horizon de prévision. Ce phénomène a été testé dans de nombreux travaux dont Atiya et al. (1999) et Karunasinghe and Liong (2006) pour la prévision des séries naturelles (*débit de rivières*).

La limite la plus spécifique aux réseaux à propagation avant pour la prévision est la fenêtre temporelle fixe qu'ils possèdent en entrée. Les réseaux à propagation avant restent les plus vulnérables à ces effets de dépendance à long terme face aux réseaux de neurones récurrents qui possèdent une mémoire interne théoriquement plus large.

2.4 Les Réseaux de Neurones Récurrents

La principale caractéristique des Réseaux de Neurones Récurrents (*Recurrent Neural Networks*), est l'existence de cycles dans le graphe orienté d'interconnexions des neurones qui représente l'architecture du réseau. L'avantage de ce type d'architectures est que le réseau réalise une mémoire interne qui peut remplacer la fenêtre temporelle utilisée dans les *FFNN* [voir Haykin (2010), Boné (2000)].

2.4.1 Architectures de réseaux récurrents

Architecture de Jordan et de Elman

Cette architecture particulière a été proposée par Jordan (1986) comme tout premier réseau récurrent. Sa principale caractéristique est l'existence d'une mémoire (appelée *couche de contexte*) liée à la couche de sortie et ayant le même nombre de neurones (figure 2.9(a)). Cette couche de contexte joue le rôle d'une mémoire qui garde l'état précédent du réseau. Inspiré du réseau de Jordan, le réseau de Elman (1989) est très similaire dans son architecture (figure 2.9(b)). La différence réside dans le bouclage qui ne se fait plus entre la couche de sortie et la couche de contexte, mais entre cette dernière et la couche cachée. A chaque instant t, la couche de contexte contient les valeurs de l'instant $t-1$ de la couche cachée, permettant ainsi d'injecter les anciennes valeurs de la couche cachée sans traitement. L'apprentissage se fait de la même manière.

Une amélioration de Jordan est proposée dans Song (2011), l'objectif est de trouver une initialisation robuste du réseau par le biais d'un algorithme d'apprentissage à contraintes.

Parmi les références dans ce type d'architecture, par exemple Kelo and Dudul (2012), Zhou et al. (2011) pour l'application à la prévision des séries temporelles de consommation d'électricité et de gaz, Liou et al. (2008) pour des applications à la robotique. Ce type d'architecture est plus fréquent dans le traitement de données que dans la prévision de séries temporelles. Un algorithme d'apprentissage qui offre aux réseaux une mémoire interne plus grande a été proposé, qui s'applique également aux architectures d'*Elman* et de *Jordanie*, comme les réseaux totalement récurrents(les architectures récurrentes génériques dans lesquelles les cycles peuvent exister entre n'importe quels neurones). Cette architecture possède un neurone sur la couche d'entrée et un neurone sur la couche de sortie. Dans Tenti (1996) l'architecture totalement récurrente est utilisé pour prédire les taux de change sur certaines devises.dans la suite, nous traitons les algorithmes d'apprentissage des réseaux globalement récurrents.

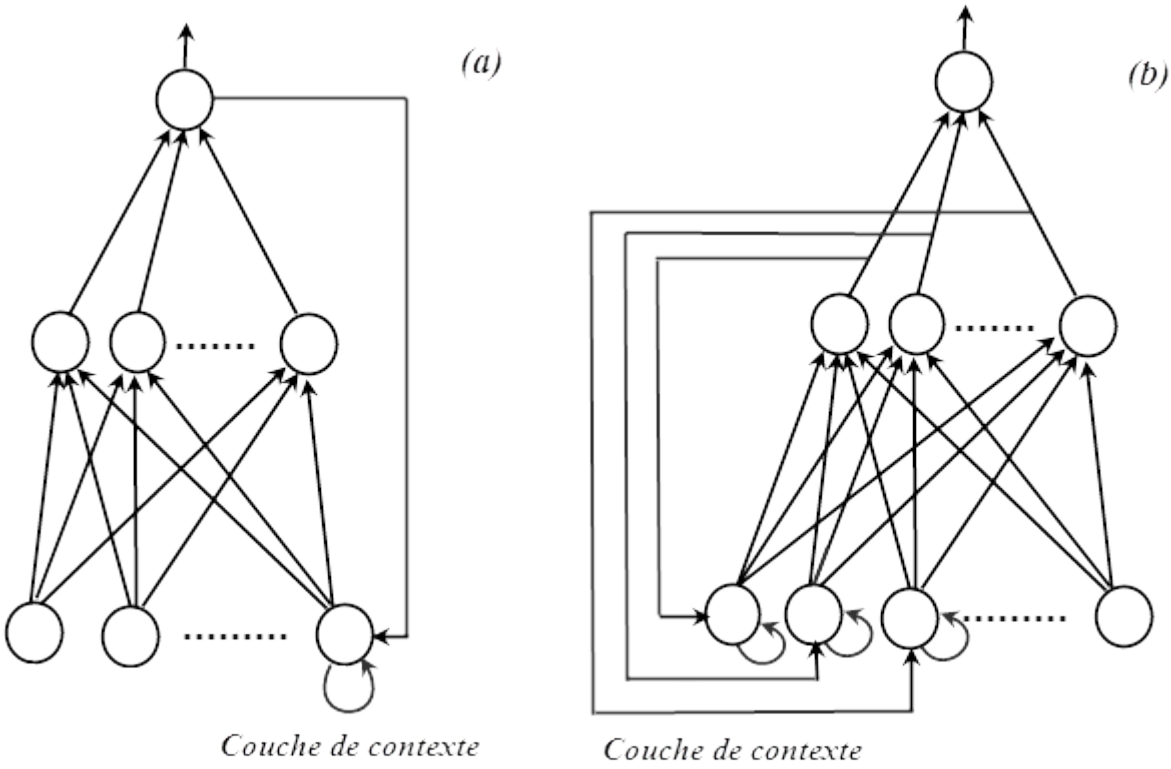

fig. 2.9 – réseau récurrent de Jordan (a) et Elman (b)

2.4.2 Algorithmes d'apprentissage des *RNN* : cas *BPTT*

On s'intéressera plus spécifiquement dans cette partie à l'algorithme de la rétropropagation dans le temps (*Back Propagation Through Time (BPTT)*). L'idée principale de cet algorithme est de déplier dans le temps le *RNN* de façon à obtenir un réseau à l couches (voir figure 2.10). Dans le cas d'un réseau récurrent, un neurone i peut être décrit par :

$$s_i(t) = f_i(net_i(t-1)) + x_i(t) \tag{2.26}$$

avec

$$net_i(t-1) = \sum_{j \in pred(i)} w_{ij}(t-1)s_j(t-1) \tag{2.27}$$

où $w_{ij}(t-1)$ est la duplication du poids w_{ij} correspondant au réseau récurrent d'origine à l'instant $(t-1)$. La variation d'un poids Δw_{ij} se calcule sous forme de la somme de toutes les variations. En appliquant la descente du gradient sur l'erreur totale, on obtient l'équation (2.28),

$$\Delta w_{ij}\,(t_1,\, t_1 - 1) = -\eta \frac{\partial E\,(t_1,\, t_l)}{\partial w_{ij}} = -\eta \sum_{\tau=t_1}^{t_l-1} \frac{\partial E\,(t_1,\, t_l)}{\partial w_{ij}\,(\tau)} \tag{2.28}$$

Nous trouvons, par la chaîne de dérivation :

$$\frac{\partial E\,(t_1,\, t_l)}{\partial w_{ij}\,(\tau)} = \frac{\partial E\,(t_1,\, t_l)}{\partial net_i\,(\tau)} \frac{\partial net_i\,(\tau)}{\partial w_{ij}\,(\tau)} \tag{2.29}$$

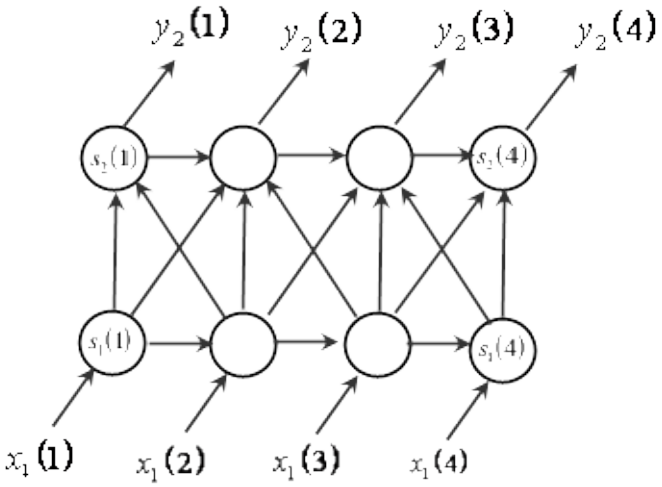

fig. 2.10 – dépliement d'un *RNN*

pour $\tau = t_l - 1$, en peut considérer le neurone i comme un neurone de la couche de sortie, donc

$$\frac{\partial E(t_1, t_l)}{\partial s_i(\tau+1)} = \frac{\partial e(t_l)}{\partial s_i(t_l)} = \begin{cases} s_i(t_l) - d_i(t_l) & si\ i \in O(\tau+1) \\ 0 & sinon \end{cases} \tag{2.30}$$

où $O(\tau+1)$ est l'ensemble de neurones de sortie à l'instant $t+1$.
Pour le cas $t_1 \leq \tau \leq t_l$, le neurone i se positionne dans une couche cachée. Nous obtenons

$$\frac{\partial E(t_1, t_l)}{\partial s_i(\tau+1)} = \frac{\partial e(\tau+1)}{\partial s_i(\tau+1)} + \sum_{j \in succi(i)} \frac{\partial E(t_1, t_l)}{\partial net_j(\tau+1)} \frac{\partial net_j(\tau+1)}{\partial s_i(\tau+1)} \tag{2.31}$$

D'après l'équation (2.27), on a

$$\frac{\partial net_j(\tau+1)}{\partial s_i(\tau+1)} = w_{ij}(\tau+1) \tag{2.32}$$

d'où

$$\frac{\partial E(t_1, t_l)}{\partial s_i(\tau+1)} = \begin{cases} s_i(\tau+1) - d_i(\tau+1) + \sum\limits_{j \in succ(i)} \frac{\partial E(t_1, t_l)}{\partial net(\tau+1)} w_{ij}(\tau+1) & si\ i \in O(\tau+1) \\ \sum\limits_{j \in succ(i)} \frac{\partial E(t_1, t_l)}{\partial net(\tau+1)} & sinon \end{cases} \tag{2.33}$$

Nous pouvons ainsi conclure à la forme finale de modification des poids pour l'algorithme de rétro-propagation dans le temps($BPTT$)

$$\Delta w_{ij}(t_1, t_l - 1) = -\eta \sum_{\tau=t_1}^{t_l-1} \frac{\partial E(t_1, t_l)}{\partial net_i(\tau)} s_j(\tau) \tag{2.34}$$

— pour $\tau = t_l - 1$

$$\frac{\partial E(t_1, t_l)}{\partial net_i(\tau)} = \begin{cases} (si(\tau) - d_i(\tau)) f_i'(net_i(\tau)) & \text{si } i \in O(\tau) \\ 0 & \text{sinon} \end{cases} \tag{2.35}$$

— pour $t_1 \leq \tau \leq t_l$

$$\frac{\partial E(t_1,t_l)}{\partial net_i(\tau)} = \begin{cases} [s_i(\tau) - d_i(\tau) + \sum\limits_{j \in succ(i)} \frac{\partial E(t_1,t_l)}{\partial net_j(\tau+1)} w_{ij}(\tau+1)] f_i'(net_i(\tau)) & si\ i \in O(\tau) \\ \sum\limits_{j \in succ(i)} \frac{\partial E(t_1,t_l)}{\partial net_j(\tau+1)} w_{ij}(\tau+1) f_i'(net_i(\tau)) & sinon \end{cases} \tag{2.36}$$

où $d_i(\tau)$ est la sortie désirée à l'instant τ. Si N est le nombre total neurones ,l'algorithme d'un réseau totalement récurrent a une complexité de $O(N^2)$. Pour éviter cela, il suffit de limiter la profondeur de la rétro-propagation.

2.4.3 Algorithmes d'apprentissage des *RNN* : cas *RTRL*

L'apprentissage récurrent temps réel(*Real Time Recurrent Learning*). Dans cet algorithme, à chaque pas de temps les poids sont modifiés. Le principe de l'algorithme est le calcul des variations des poids par la descente du gradient en appliquant ses variations à chaque nouvel exemple présenté (voir l'équation 2.37).

$$\Delta w_{ij} = \sum_{t=t_1}^{t_l} \Delta w_{ij}(t) \tag{2.37}$$

tel que

$$\Delta w_{ij}(t) = -\nu \frac{\partial e(t)}{\partial w_{ij}} = v \sum_{t \in T(t)} (d_r(t) - s_r(t)) \frac{\partial s_r(t)}{\partial w_{ij}} \tag{2.38}$$

Par le même raisonnement appliqué avec $BPTT$, on retrouve la formule de modification des poids à l'instant t en fonction de l'instant $(t-1)$.

$$\frac{\partial s_r(t)}{\partial w_{ij}} = f'\left(net_r(r-1)\right) \left[\partial_{ir} s_j(t-1) + \sum_{q \in pred(r)} \frac{\partial s_q(t-1)}{\partial w_{ij}} \right] \tag{2.39}$$

avec la condition initiale

$$\frac{\partial s_r(t)}{\partial w_{ij}} = 0 \tag{2.40}$$

Cet algorithme se caractérise par une plus faible occupation mémoire par rapport à $BPTT$ mais une complexité en temps plus élevée ($O(N^4)$ *comparée à* $O(N^3)$).

Conclusion

Comme les *RNA* se caractérisent selon le domaine d'utilisation et des problèmes qu'ils peuvent résoudre, nous avons cité leurs domaines reliant à leurs capacités de résoudre un tel problème. Ils se caractérisent aussi par leurs différentes architectures et topologies, traitées dans ce chapitre.

Le type des réseaux de neurones choisi dans ce chapitre est le perceptron multi couche *MLP*. C'est de loin le plus utilisé, aussi nous nous focaliserons sur ce type d'outil, sans tenir compte des nombreux autres réseaux que l'on a énumérés durant ce chapitre.

Le prochain chapitre va détailler la méthodologie des algorithmes génétiques, ainsi le fondement théoriques de cette méthode.

CHAPITRE 3

Algorithmes génétiques

Les algorithmes génétiques sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et de l'évolution naturelle (qui s'inspirent de la science génétique développée au $XIXe$ siècle par Darwin, [Dréo et al. (2003))] : croisements, mutations, sélection, etc. Les algorithmes génétiques ont déjà une histoire relativement ancienne puisque les premiers travaux de *John Holland* sur les systèmes adaptatifs remontent à 1962 [Holland (1962)]. L'ouvrage de *David Goldberg* [Goldberg (1989a)] a largement contribué à les vulgariser.

Le but principal de ces algorithmes est de trouver une solution pour des problèmes difficiles -des problèmes où on ne connaît pas de méthodes exactes pour les résoudre en temps raisonnable. Ces approches permettent l'évolution d'une génération (solution) à une autre par un ensemble d'opérations (mutation et croisement) tout en cherchant à améliorer une fonction objectif. Le codage des éléments de la population est l'élément le plus important, puisqu'il permet de modéliser un problème sous forme de données informatiques manipulables [Alliot and Durand (2005), Chouchani (2010)].

Leurs champs d'application sont très vastes. Outre leurs applications plus spécifiques en économie discutées dans cet article, les AG sont utilisés en programmation génétique [Koza (1992)] , en théorie du contrôle optimal [Krishnakumar and Goldberg (1992), Marco et al. (1996) et plus récemment Jamshidi et al. (2002)], ou encore en théorie des jeux répétés [Axelrod et al. (1987)], ainsi que pour l'optimisation de fonctions [De Jong (1980)].

Pan et al. (1995) utilisent les AG afin de trouver les paramètres optimaux de régressions non-linéaires. Dans un autre ordre d'idée, Boné et al. (1998) utilisent les AG pour trouver la forme économétrique théorique la mieux adaptée à la modélisation de séries temporelles. Chez eux, la population de chromosomes définit le type de modélisation (AR, MA ou $ARMA$) et la valeur des coefficients associés. Chikr El Mezouar and Gasmi (2015) les chromosomes représentes les coefficients du modèle *auto-régressive* .

3.1 Principe général

Un algorithme génétique réalise une optimisation dans un espace de données. Ils utilisent un vocabulaire similaire à celui de la génétique naturelle. Cependant, les

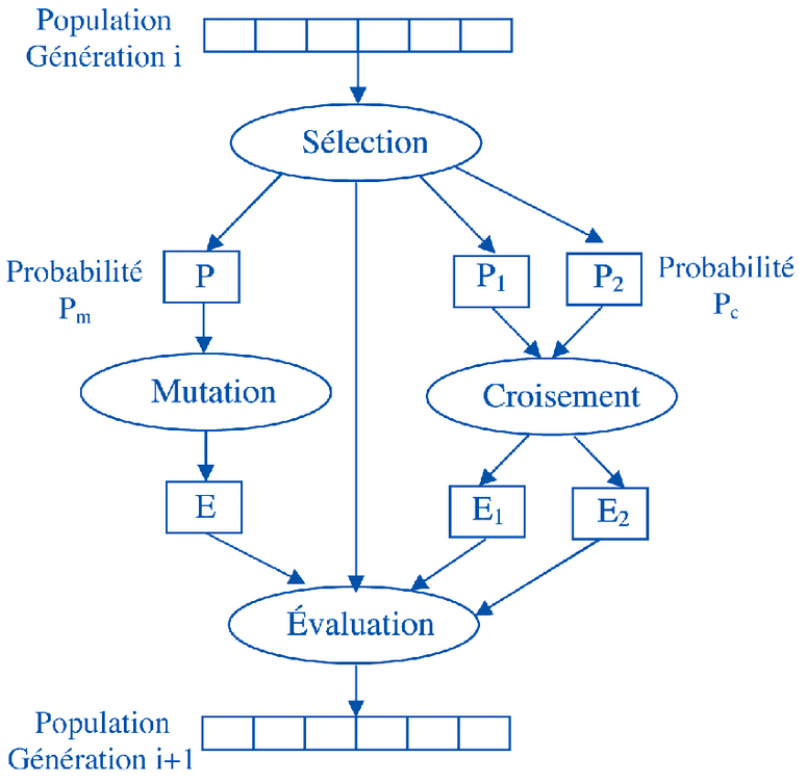

fig. 3.1 – Principe général d'un algorithme génétique

processus naturels au quels ils font référence sont beaucoup plus complexes. On parlera ainsi d'individu dans une population. L'*individu* est composé de *chromosomes*. Les chromosomes sont eux-mêmes constitués de *gènes* qui contiennent les caractères héréditaires de l'individu. Les principes de *sélection*, de *croisement*, de *mutation* s'appuient sur les processus naturels du même nom [Durand (1996)]. Le principe général des algorithmes génétiques est décrit par la figure (3.1).

Nous commençons par générer une population d'individus aléatoirement. Pour passer d'une génération k à la génération $(k+1)$, les trois opérations suivantes sont répétées pour tous les éléments de la population k. Des couples de parents P_1 et P_2 sont sélectionnés en fonction de leurs adaptations. L'opérateur de croisement leur est appliqué avec une probabilité P_c et génère des couples d'enfants C_1 et C_2. D'autres éléments P sont sélectionnés en fonction de leur adaptation. L'opérateur de mutation leur est appliqué avec la probabilité P_m (P_m est généralement très inférieur à P_c) et génère des individus mutés P'. Le niveau d'adaptation des enfants (C_1, C_2) et des individus mutés P' sont ensuite évalués avant insertion dans la nouvelle population.

Les critères d'arrêt de l'algorithme peuvent être choisis par :

— Nombre de générations que nous aimerions exécuter peut être fixé a priori (pour trouver une solution dans un temps déterminé).

— L'algorithme peut être arrêté lorsque la population n'évolue plus assez rapidement.

Pour pouvoir utiliser les techniques génétiques il faut disposer des éléments suivants :

1. génération aléatoire d'un certain nombre de séquences de bits pour composer la

« population » initiale ;

2. mesure de l'adaptation de chacune des séquences présentes ;
3. reproduction de chaque séquence en fonction de son adaptation. Les séquences les mieux adaptées se reproduisent mieux que les séquences inadaptées. La nouvelle population est composée des séquences après reproduction ;
4. on remplace un certain nombre de paires de séquences tirées aléatoirement par le croisement de ces paires (le lieu de croisement dans la séquence de bits est également choisi de façon aléatoire.). Chaque nouvelle paire est constituée de la façon suivante :
   — une séquence est composée de la première partie de la première séquence et de la seconde partie de la seconde séquence ;
   — l'autre séquence est composée de la première partie de la seconde séquence et de la seconde partie de la première séquence.
5. mutation d'un bit choisi aléatoirement dans une ou plusieurs séquences tirées au sort ;
6. Retour à l'étape 2.

Avant de détailler chacun de ces points, rappelons les fondements mathématique des AGs.

3.2 Convergence théorique

On trouve néanmoins plusieurs démonstrations de convergence théorique des algorithmes évolutionnaires dont :

— Au sens de la convergence faible des mesures de probabilités,un résultat très général de convergence globale proposé par Zilinskas and Zhigljavsky (1991) ;
— Des résultats partiels basés sur un modèle des algorithmes évolutionnaires en tant que chaînes de Markov dans l'espace des populations [voir Davis (1991) ; Nix and Vose (1992) ; Fogel et al. (1966)] ;
— Un résultat de convergence fort pour les *AGs* basé sur la théorie de *Friedlin Weinzel* des perturbations stochastiques des systèmes dynamiques [Cerf (1994)]. Cette dernière approche est la plus satisfaisante tant sur le plan mathématique, que sur celui de la modélisation, les différents opérateurs étant présentés comme «perturbant» un *processus Markovien* représentant la population à chaque étape. Ici encore il est démontré l'importance de l'opérateur de mutation, le croisement pouvant être totalement absent. L'investissement nécessaire pour la compréhension des démonstrations de *Cerf* est assez important. De plus, de nombreuses interrogations demeurent cependant concernant les relations réelles entre les différents paramètres caractérisant l'algorithme génétique et les choix pratiques de ceux-ci. Dans ce domaine, la pratique devance encore la théorie, même si les mécanismes commencent à être plus clairs. Les raffinements utilisés en pratique pour faire converger l'*AG* (*sharing, scaling, clustering*) n'ont pas encore été étudiés sur le plan théorique ;
— Des résultats de convergence globaux avec taux de convergence pour certains algorithmes de stratégies d'évolution, et sur des fonctions convexes [Bäck and Schwefel (1993)].Le fait que ces résultats soit obtenus sur des fonctions convexes appauvrit largement leur intérêt, sachant que pour de telles fonctions, des algorithmes classiques de type gradient sont généralement largement plus performants.

Les algorithmes génétiques binaires sont historiquement plus anciens , donc été les plus étudiés. Nous allons les étudier en premier, avant de présenter la modélisation basée sur les chaînes Markov.

3.2.1 Définitions fondamentales

Nous donnons d'abord quelques définitions

Définition 3.2.1. *Séquence/Chromosome/Individu.*
On appelle séquence A de longueur $l(A)$ une suite $A = a_1 a_2 \cdots a_l$, avec $i \in [1, l]$, $a_i \in V = \{0, 1\}$.

En codage binaire, les chromosomes sont des séquences. Dans le cas d'un codage non binaire, tel que le codage réel (voir partie application), la séquence ne contient qu'un point, nous avons $A = \{a\}$ avec $a \in \mathbb{R}$.

Nous avons besoin de la notion de schéma et de quelques définitions qui lui sont associées :

Définition 3.2.2. *Schéma.*
On appelle schéma H de longueur l une suite $H = a_1 a_2 \cdots a_i$ avec $\forall i \in [1, l]$, $a_i \in V^+ = \{0, 1, *\}$.

une $*$ en position i signifie que a_i peut être indifféremment un 0 ou un 1.

Définition 3.2.3. *Instance.*
On dit qu'une séquence $A = a_1 a_2 \cdots a_l$ est une instance d'un schéma $H = b_1 b_2 \cdots b_l$ si pour tout i tel que $b_i \neq *$ on a $a_i = b_i$.

exemple : $H = 10*0$ est un schéma et les séquences 1000 et 1010 sont des instances de H.

Définition 3.2.4. *Position fixe, position libre.*
Soit un schéma H. On dit que i est une position fixe de H si $a_i = 0$ ou $a_i = 1$. On dira que i est une position libre de H si $a_i = *$.

Définition 3.2.5. *Ordre d'un schéma.*
On appelle ordre du schéma H le nombre de positions fixes de H, noté $o(H)$.

Définition 3.2.6. *Longueur fondamentale*
On appelle longueur fondamentale du schéma H la distance séparant la première position fixe de H de la dernière position fixe de H. On note cette longueur $\delta(H)$.

exemple : $H = \underset{1}{*}\underset{2}{0}\underset{3}{1}\underset{4}{*}\underset{5}{0}$ a pour longueur fondamentale $\delta(H) = 5 - 2 = 3$.

Définition 3.2.7. *Adaptation d'une séquence.*
On appelle adaptation d'une séquence A, une valeur positive que nous noterons $f(A)$ (fitness).

Définition 3.2.8. *Adaptation d'un schéma.*
On appelle adaptation d'un schéma H la valeur :

$$f(H) = \frac{\sum_{i=1}^{2^{(l(H)-o(H))}} f(A_i)}{2^{(l(H)-o(H))}}$$

où les A_i décrivent l'ensemble des instances de H.

3.2.2 Effets de la reproduction

Soit un ensemble $S = \{A_1, \cdots, A_n\}$ de n séquences de bits tirées aléatoirement. Durant la reproduction, chaque séquence A_i se reproduira avec une probabilité :

$$p_i = p(A_i) = \frac{f(A_i)}{\sum_{i=1}^{n} f(A_i)}$$

Si nous ayons un nombre $m(H,t)$ à l'instant t de séquences représentant le schéma H dans la population S. Nous devons a l'instant $t+1$ statistiquement avoir :

$$m(H,t+1) = m(H,t).n.\frac{f(H)}{\sum_{i=1}^{n} f(A_i)} = m(H,t)\frac{f(H)}{\bar{f}_i}$$

$\overline{f}_t$ représente la moyenne de l'adaptation des séquences à l'instant t.
Si on posent $c_t(H) = \frac{f(H)}{\overline{f}_t} - 1$, on obtient : $m(H,t+1) = (1 + c_t(H))m(H,t)$, avec $c_t(H)$ l'approximation que est constant dans le temps. Nous avons alors :

$$m(H,t) = (1 + c(H))^t \cdot m(H,0)$$

Donc : si la reproduction seule était en jeu, les schémas forts élimineraient très rapidement les schémas faibles voir [Bourazza (2006)].

3.2.3 Effets des croisements

Intéressons nous à la probabilité de survie $p_s(H)$ d'un schéma H lors d'une opération de croisement. Considérons cela sur un exemple. Soit le schéma $H = **10*1**$. Supposons qu'une séquence qui est une instance de ce schéma soit croisée avec une autre séquence. Quelle est la probabilité pour que la séquence résultante soit encore une instance de H ? Il est impossible de répondre exactement à la question, tout au plus peut-on donner une borne inférieure de cette valeur. Il est clair que H ne sera pas détruit si le site de croisement qui est tiré au sort est inférieur à 3 (avant le premier 1) ou s'il est supérieur à 5 (après le dernier 1). On voit donc immédiatement qu'une borne inférieure de la probabilité de détruire un schéma H est $\frac{\delta(H)}{(l-1)}$. Donc la probabilité de survie dans un croisement est $1 - \frac{\delta(H)}{(l-1)}$. Si d'autre part on ne croise qu'une fraction p_c de séquences dans une population donnée, la probabilité de survie est donnée par :

$$p_s \geq 1 - p_c\frac{\delta(H)}{(l-1)}$$

De ce résultat et des résultats précédents découle une loi d'évolution d'une population :

$$m(H,t+1) \geq m(H,t)(1 + c_t(H))\left(1 - p_c \cdot \frac{\delta(H)}{l-1}\right)$$

3.2.4 Effets des mutations

Supposons que la probabilité de mutation d'un bit dans une séquence soit p_m . Dans un schéma H, seules les positions fixes peuvent être détruites. Comme la probabilité de survie d'un bit est $1 - p_m$, la probabilité de survie d'un schéma H contenant $o(H)$ positions fixes est $(1 - p_m)^{o(H)}$. La probabilité de mutation étant

toujours petite devant 1, on peut faire un développement limité au premier ordre, ce qui nous donne une probabilité de survie de $1-o(H)p_m$. l'équation finale s'écrit donc :

$$m\left(H,\, t+1\right) \simeq m\left(H,\, t\right)\left(1+c_t\left(H\right)\right)\left(1-p_c\frac{\delta\left(H\right)}{l-1}-o\left(H\right)p_m\right)$$

La formule ci-dessus nous apprend deux choses :

— les schémas qui ont une longueur fondamentale petite sont plus favorisés que les autres, lors de la génération d'une nouvelle population ;

— les schémas qui ont un ordre petit sont plus favorisés que les autres, lors de la génération d'une nouvelle population.

Ceci enseigne une chose fondamentale pour le codage de données : les schémas qui codent les données « intéressantes » pour le problème doivent avoir un ordre et une longueur fondamentale faibles, alors que les données « sans intérêt » doivent être codées par des schémas qui ont un ordre et une longueur fondamentale élevés.

Les résultats ci-dessus ne sont qu'une introduction à la théorie des schémas.Il existe beaucoup de détails dans les références suivantes : Goldberg (1989b) ; Vose (1991) ; Bridges and Goldberg (1987) ; Smith et al. (1991) ; Bui and Moon (1994).

3.3 Modélisation par chaîne de Markov

Nous présentons ici une version simplifiée de cette théorie et les principaux résultats de convergence. Les principaux résultats asymptotiques directement liés aux algorithmes génétiques ont été développé par Cerf (1994) basé sur les travaux de Catoni (1990) et Trouve (1993). En particulier, la théorie de Freidlin and Wentzell (1998) est la pierre angulaire de ces études. Comme l'avons nous vu, l'algorithme génétique est un algorithme stochastique itératif qui opère sur des ensembles de Points, et qui est bâti à l'aide de : mutation, croisement et sélection, que nous présentons plus formellement à présent.

sur la base d'un codage binaire, P représentant le nombre de bits utilisés pour le codage. La fonction d'évaluation, f est définie sur l'espace $E = \{0,1\}^P$ à valeurs dans $\mathbb{R}^+$. Problème est donc de localiser l'ensemble des maximums globaux de f, ou, à défaut, de trouver rapidement et efficacement des régions de l'espace où se situent ces maximums.

3.3.1 Description formelle des AGs

Soit N la taille (fixe) de la population, notons X_k la population de la génération k : il s'agit d'une matrice $X_k = (X_k^1, \cdots, X_k^N) \in E^N$, avec $(X_k^i)_{1\leq i\leq N}$ sont des chromosomes de taille P (chaînes de bits). Le mécanisme de transition comporte trois étapes :

$$X_k \xrightarrow{mutation} Y_k \xrightarrow{croisement} Z_k \xrightarrow{selection} X_{k+1}$$

Chacune de ces étapes peut être modélisée formellement.

Mutation :

L'opérateur considéré ici est le suivant : pour chaque composante de chaque élément X_k^i, une variable de *Bernouilli* de paramètre p_m (probabilité préalablement choisie) est tirée indépendamment et suivant le résultat l'élément binaire examiné est changé ou non. (1 en 0 et 0 en 1).

Croisement :

L'opérateur étudié ici est l'opérateur à un point (*slicing crossover*). Ici encore, la probabilité de croisement p_c est fixée initialement.

Pour construire la population Z_k, $\frac{N}{2}$ couples sont formés à partir de la population Y_k (par exemple en appariant les individus consécutifs de Y_k, ou bien en choisissant au hasard et uniformément des individus dans Y_k). Pour chaque couple, une variable de *Bernoulli* de paramètre p_c est tirée pour décider si le croisement a lieu. Si c'est le cas, un site de coupure est tiré au hasard, et les segments finaux des deux chromosomes sont échangés.

Sélection

Les N individus de la population X_{k+1} sont obtenus après sélection des individus de Z_k. On conserve ainsi les « meilleurs » individus de Z_k, indépendamment à l'aide d'une distribution de probabilité qui favorise les individus de Z_k les mieux adaptés. La probabilité de sélection de l'individu Z_k^i est :

$$p_i = p(Z_k^i) = \frac{f(Z_k^i)}{\sum_{j=1}^{N} f(Z_k^j)}$$

3.3.2 Modélisation

La présentation des opérateurs permet de modéliser la suite des $(X_k)_{k\in\mathbb{N}}$ en une chaîne de Markov, d'espace d'états $E = (\{0, 1\}^p)^N$. La propriété principale de cette formalisation est que la loi de X_k est uniquement déterminée par :

- la loi de la génération initiale X_0 ;
- le mécanisme de transition de X_k à X_{k+1} [voir Cerf (1994)] ;
- Il est homogène (indépendant de la génération, k) ;
- Il est irréductible, (la probabilité de joindre deux points quelconques de l'espace d'état, en un nombre fini de générations est non nulle) soit :

$$\forall x,\, y \in E,\, \exists\, r \in \mathbb{N},\;\; P\left[X_{k+r} = y | X_k = x\right] > 0$$

  (Le mécanisme permet d'explorer tout point de l'espace d'état, avec une probabilité non nulle) ;
- Il est irrégulier (cette hypothèse n'est pas fondamentale).

Théorème 3.3.1. *Une chaîne de Markov homogène irréductible apériodique d'espace d'états fini est ergodique et possède une unique mesure de probabilité stationnaire ou invariante.*

Cette mesure stationnaire correspond à la loi régissant l'équilibre du processus, elle est définie , pour tout y, comme :

$$\mu(y) = \lim_{k\to+\infty} P[X_k = y | X_0 = x]$$

Nous savons également que tout élément de l'espace d'états est de probabilité non nulle pour cette mesure. Toutefois, si ce résultat nous permet de savoir qu'il existe une dynamique de fond de l'algorithme génétique, il nous reste à en déterminer les propriétés, l'influence des opérateurs (et des paramètres associés) qui jouent un grand rôle dans le processus [voir Durand (1996)]. Pour cela nous introduisons les notations

suivantes :
Notations : Si $x = (x_1, \ldots, x_N)$ est un élément de E^N et i un point de E, nous noterons :

$$\hat{f}(x) = \hat{f}(x_1, \cdots, x_N) = max\{f(x_i) : 1 \leq i \leq N\}$$
$$\hat{x} = \{x_k \in \arg\max f(x)\}$$
$$[x] = \{x_k : 1 \leq k \leq N\}$$

Processus de fond (X_k^∞) :

C'est à partir de ce processus de fond que l'algorithme génétique est reconstitué en étudiant ses perturbations aléatoires par les différents opérateurs. Il est défini comme un processus limite, lorsque les perturbations ont disparu. C'est une chaîne de Markov sur E^N dont le mécanisme de transition est très simple :
Les N composantes de X_{k+1}^∞sont choisies indépendamment et suivant la loi uniforme sur l'ensemble $\hat{X}_k^\infty$:

— Les individus dont l'adaptation n'est pas maximale en k, sont éliminés et n'apparaissent pas dans la génération $k+1$;
— Les individus dont l'adaptation est maximale, ont des chances de survies égales.

Cette chaîne est instantanément piégée dans l'ensemble des populations d'équi-adaptation

$$S = \{\, x \in\ E^N :\ f(x_1) = \cdots = f(x_N)\}$$

puis, au bout d'un certain temps, elle est absorbée par une population uniforme :

$$\forall x \in E^N,\ P[\exists x_i \in \hat{x},\ \exists K\, \forall k \geq K \quad X_k^\infty = x_i \mid X_0^\infty = x_{ini}] = 1$$

En effet, le mécanisme de sélection ne préserve pas la diversité et certains individus disparaissent par hasard.

Lorsque la population est devenue uniforme et en l'absence ici de perturbations, il ne se passe plus rien. Ceci peut également se traduire en définissant les populations uniformes comme les états absorbants de la chaîne X_k^∞. Nous allons maintenant étudier la situation où ce processus est perturbé.

Processus perturbé (X_k^l)

La modélisation proposée par *Cerf* [Cerf (1994)], part du processus de fond (X_k^∞),décrit ci-dessus, qui est perturbé aléatoirement, les perturbations sont indicées par le paramètre l. La chaîne de Markov (X_k^∞) devient donc une suite de chaînes de Markov (X_k^l). dont le mécanisme de transition est donné par la succession des transformations générées par les opérateurs.

$$X_k^l \xrightarrow{mutation} U_k^l \xrightarrow{croisement} V_k^l \xrightarrow{selection} X_{k+1}^l$$

Nous devons modéliser avec précision les opérateurs.

Mutation

Il est assez naturel d'introduire la probabilité $p_l(i,j)$ de transition de mutation entre les points i et j de E, comme un noyau Markovien p_l. Sur la chaîne X_k^l, la probabilité de transition entre les points x et u de E^N est :

$$P[U_k^l = u \mid X_k^l = x] = p_l(x_1, u_1) \times \cdots \times p_l(x_N, u_N)$$

Plus précisément , et pour analyser la dynamique de X_k^l lorsque l tend vers l'infini, nous reportons ici les hypothèses sur le mode et la vitesse de convergence des probabilités de transition. Supposons l'existence d'un noyau, α (irréductible) , sur E : (il existe un chemin de E)

$$\forall\, i,\, j \,\in E\, \exists\, i_0,\, i_1,\, \cdots,\, i_r \;\; i_0 = i\,,\;\; i_r = j \;\; \text{tel que} \prod_{0 \le k \le r-1} \alpha\,(i_k,\, i_{k+1}) > 0$$

L'irréductibilité du noyau α assure que tout point de l'espace est potentiellement visitable.
La vitesse de convergence du noyau p_l, est caractérisée par le réel positif a, tel que p_l admette le développement suivant :

$$\forall i, j \in E,\ \forall s > 0\,,\ p_l(i,j) = \begin{cases} \alpha(i,j) l^{-a} + o(l^{-s}) & \text{si } i \neq j \\ 1 - \alpha(i,j) l^{-a} + o(l^{-s}) & \text{si } i = j \end{cases} \tag{3.1}$$

La condition de positivité de a nous permet de faire disparaître les perturbations lorsque l tend vers l'infini.

$$\forall i, j \in E \lim_{l \to \infty} p_l(i,j) = \delta(i,j) = \begin{cases} 0 & \text{si } i \neq j \\ 1 & \text{si } i = j \end{cases} \tag{3.2}$$

Croisement

Les croisements sont modélisés comme des petites perturbations aléatoires des couples formés par les individus consécutifs de la population U_k^l. Ces couples sont ici formés par les éléments successifs de la population, les transitions sont gérées par le noyau Markovien q_l sur $E \times E$, cette fois, de sorte que :

$$P(V_k^l = v \,|\, U_k^l = u) = q_l((u_1, u_2), (v_1, v_2)) \cdot q_l((u_3, u_4), (v_3, v_4)) \cdots$$

Pour ce noyau q_l nous supposerons l'existence d'un noyau irréductible β sur $E \times E$, la vitesse de convergence est alors paramétrée par le réel positif b de sorte que :

$$\forall (i_1, j_1) \in E \times E,\ \forall (i_2, j_2) \in E \times E,\ \forall s > 0$$

$$q_l((i_1,j_1),(i_2,j_2)) = \begin{cases} \beta((i_1,j_1),(i_2,j_2)) l^{-b} + o(l^{-s}) & \text{si}(i_1,j_1) \neq (i_2,j_2) \\ 1 - \beta((i_1,j_1),(i_2,j_2)) l^{-b} + o(l^{-s}) & \text{si}(i_1,j_1) = (i_2,j_2) \end{cases} \tag{3.3}$$

L'évanouissement asymptotique des croisements est également imposée par la positivité de b :

$$\forall i_1, i_2, j_1, j_2 \in E \quad \lim_{i \to \infty} q_l((i_1,j_1),(i_2,j_2)) = \delta(i_1,i_2)\delta(j_1,j_2) \tag{3.4}$$

Sélection

C'est l'opérateur le plus compliqué et également le plus important puisqu'il permet la convergence vers les optima de f. Il est modélisé à l'aide d'une fonction de sélection F_l dont Cerf (1994) nous donne une définition précise, pouvant être résumée par :

$$\begin{array}{rcl} F_i : \{1, \cdots N\} \times (\mathbb{R}^+)^N & \to & [0,1] \\ (i, f_1, \cdots, f_N) & \mapsto & F_l(i, f_1, \cdots, f_N) \end{array}$$

telle que :

— $F_l(i, f_1, \cdots, f_N)$ est une probabilité sur $\{1, \cdots, N\}$;
— Cette probabilité est indépendante de l'indexation des f_1, f_2, $\cdots$, f_N (on peut permuter les f_i) ;
— La probabilité favorise les éléments i associés à des valeurs élevées (i.e.) Si $f_1 \geq f_2 \geq \cdots \geq f_N$, alors $F_l(1, f_1, \cdots, f_N) \geq F_l(2, f_1, \cdots, f_N) \geq \cdots \geq F_l(N, f_1, \cdots, f_N)$

Cet outil nous permet d'écrire la probabilité de transition correspondant à la dernière étape.

$$P[X_{k+1}^l = x \,|\, V_k^l = v] = \prod_{r=1}^{N} Y_l(x_r, v_r)$$

Ceci signifie que la probabilité de transition est le produit des probabilités sur chacune des N composantes de E.
La probabilité Y_l entre deux composantes (x_r, y_r) est donnée par :

$$Y_l(x_r, v_r) = \sum_{k:v_k = x_k} F_l(k, f(v_1), \cdots, f(v_N))$$

Comme pour les autres opérateurs, la fonction de sélection doit être sélectionnée et sa vitesse de convergence caractérisée ; pour la fonction de sélection F_l, nous choisissons :

$$F_l(k, f(v_1), \cdots, f(v_N)) = \frac{\exp(c \cdot f_k \ln(l))}{\sum_{r=1}^{N} \exp(c \cdot f_r \ln(l))} \tag{3.5}$$

Ce choix correspond à une probabilité de sélection favorisant de fortes adaptations au détriment du faible, le réel positif c indexant cette fonction.
Le mécanisme de sélection opérant sur le processus de fond (X_l^∞), correspond à la fonction de sélection F_∞ définie par (la loi uniforme sur $\hat{x}$) :

$$F_\infty(k, f(x_1), \cdots, f(x_N)) = \frac{1_{\hat{x}}(x_k)}{card(\hat{x})}$$

La suite $(F_l)_{l \in \mathbb{N}}$ des fonctions de sélection tend vers cette loi uniforme

$$\forall x \in E^N,\ \forall k, \lim_{l \to \infty} F_l(k, f(x_1), \cdots, f(x_N)) = F_\infty(k, f(x_1), \cdots, f(x_N)) \tag{3.6}$$

La formulation proposée en (3.1), (3.3) et (3.5), permet un ajustement équitable de ces vitesses (elles sont logarithmiquement du même ordre) de sorte qu'aucun opérateur ne domine les autres dans la dynamique. Les conditions (3.2), (3.4) et (3.6) conduire à la convergence, lorsque l tend vers l'infini, du mécanisme de transition de la chaîne(X_k^l) vers le mécanisme de transition de la chaîne (X_k^∞)

$$\forall x, y \in E^N,\ \forall k, \lim_{l \to \infty} P(X_{k+1}^l = z \,|\, X_k^l = y) = P(X_{k+1}^\infty = z \,|\, X_k^\infty = y)$$

et en ce sens, la chaîne (X_k^l) est une perturbation de la chaîne (X_k^∞).
La chaîne (X_k^l) se comporte alors comme le ferait (X_k^∞).

Conclusion

D'autres résultats existent, tant dans le travail de *Cerf*, que dans la littérature citée dans cette section. Ils demandent cependant un investissement supplémentaire dans la compréhension des outils développés. Le but de cette section était de montrer

que l'étude théorique des algorithmes génétiques donne de substantiels résultats. De nombreuses interrogations demeurent cependant concernant les relations réelles entre les différents paramètres caractérisant l'algorithme génétique et les choix pratiques de ceux-ci. Dans ce domaine, la pratique devance encore la théorie, même si les mécanismes commencent à être plus Clairs [Alliot and Durand (2005)]. En d'autres termes, pour chaque problème pour lequel on trouve un codage, des opérateurs de croisement et de mutation efficaces on peut fabriquer un problème qui rendra inefficace ce codage [Chikr El Mezouar and Gasmi (2015), Cerf (1996a), Cerf (1996b)] et ces opérateurs [Durand (2004)].

3.4 Description détaillée

Le codage des paramètres et les opérateurs de sélection, de croisement et de mutation sont présentés en détails.

3.4.1 Codage

Il y a plusieurs types de codage : binaire, réel, codage de Gray et codage dynamique des paramètres. Chacun ayant ses propres avantages et inconvénients.

Historiquement le codage utilisé par les algorithmes génétiques était représenté sous forme de chaînes de bits contenant toute l'information nécessaire à la description d'un point dans l'espace d'états. Ce type de codage a pour intérêt de permettre la création d'opérateurs de croisement et de mutation simples. C'est également en utilisant ce type de codage que les premiers résultats de convergence théorique ont été obtenus. Cependant, ce type de codage n'est pas toujours bon :
par exemple, les chaînes « 01111 » et « 10000 » correspondent à deux configurations réelles voisines alors qu'elles diffèrent de cinq bits. Cette caractéristique peut s'avérer pénalisant pour la recherche locale par croisement.

L'utilisation du code de *Gray* (*binaire réfléchi* ; permet de coder une valeur numérique en cours d'évolution en une suite de configurations binaires se différenciant l'une de l'autre par le changement d'état d'un seul bit à la fois.) a été recommandée pour répondre à ce problème [voir Caruana and Schaffer (1988),et Back (1993)].

Pour des problèmes d'optimisation dans des espaces de grande dimension, le codage binaire peut rapidement devenir mauvais. Généralement, chaque variable est représentée par une partie de la chaîne de bits et la structure du problème n'est pas bien reflétée, l'ordre des variables ayant une importance dans la structure du chromosome, alors qu'il n'en a pas forcément dans la structure du problème.

Une des améliorations majeures consiste alors à se servir directement de nombres réels [Wright (1991),et Goldberg (1991)]. Les résultats donnés par Michalewicz (1994) montrent que la représentation réelle aboutit souvent à une meilleure précision et qu'en règle générale le gain en termes de temps de calcul (CPU) est important. Ils en concluent qu'une représentation plus naturelle du problème offre des solutions plus efficaces (pour la partie application , nous avons utilisé le codage réel).

3.4.2 population initiale

Le choix de la population initiale d'individus peut conditionner fortement la rapidité de l'algorithme. Elle peut être obtenue en générant aléatoirement les individus

dans l'espace de recherche suivant une loi de probabilité uniforme, si la position de l'optimum dans l'espace d'états est totalement inconnue [Michalewicz and Janikow (1991)]. Si par contre, des informations a priori sur le problème sont disponibles, il paraît naturel d'engendrer les individus dans un sous-domaine particulier afin d'accélérer la convergence.

Disposant d'une population d'individus non homogène, la diversité de la population doit être entretenue au cours des générations, afin de parcourir le plus largement possible l'espace d'état. C'est le rôle des opérateurs de croisement et de mutation.

3.4.3 Opérateur de croisement

Le premier opérateur du module de reproduction est le croisement, il consiste à recombiner des gènes pour simuler la période de reproduction entre deux individus. À chaque génération, cet opérateur réorganise des combinaisons de gènes, nommées schémas, dans le but d'augmenter progressivement le mérite des chromosomes. Ce type de croisement à découpage de chromosomes est très efficace pour les problèmes discrets [voir Chikr El Mezouar and Gasmi (2015), Spears and De Jong (1991)].

Pour un codage réel, plusieurs types de croisement sont possibles. Nous citerons, en particulier, le croisement arithmétique entier. La particularité de ce dernier réside d'un nombre aléatoire r [Alliot (1998)], tel que les enfants $C_1^{'}$ et $C_2^{'}$ obtenus après croisement, soient le résultat d'une combinaison linéaire des deux parents C_1 et C_2 :

$$\begin{cases} C_1^{'} = & rC_1 + (1-r)C_2 \\ C_2^{'} = & rC_2 + (1-r)C_1 \end{cases} \tag{3.7}$$

Nous pouvons imaginer et tester des opérateurs de croisement plus ou moins complexes sur un problème donné mais l'efficacité de ceux-ci est souvent intrinsèquement liée au problème.

3.4.4 Opérateur de mutation

L'opérateur de mutation donne aux algorithmes génétiques la propriété de l'ergodicité de parcours d'espace. Cette propriété indique que l'algorithme génétique sera susceptible d'atteindre tous les points de l'espace d'état, sans pour autant les parcourir tous dans le processus de résolution. Ainsi en toute rigueur, l'algorithme génétique peut converger sans croisement, et certaines implémentations fonctionnent de cette manière [Fogel et al. (1966)].

3.4.5 Principes de sélection

Par analogie au processus de sélection naturelle, un caractère aléatoire est conféré à la sélection des individus tout en exploitant les valeurs de la fonction d'adaptation.La sélection permet d'identifier statistiquement les meilleurs individus d'une population et d'éliminer les mauvais. On trouve dans la littérature un nombre important de principes de sélection plus ou moins adaptés aux problèmes qu'ils traitent [Michalewicz (1994), et Rahmat-Samii and Michielssen (1999)].
Les deux principes de sélection suivants sont les plus couramment utilisés :

- La sélection proportionnelle (*Proportional Selection* ou *Roulette Wheel Selection*) ;
- La sélection du tournoi (*tournament selection*).

Le principe de la *sélection proportionnelle* consiste à dupliquer chaque individu proportionnellement à son adaptation dans son milieu [Holland (1975)] : En pratique, on utilise une roue de loterie divisée en N secteurs (représentant les individus à sélectionner) dont la surface est proportionnelle à la probabilité de sélection correspondante. Elle est lancée autant de fois qu'il y a d'individus et à chaque coup, l'individu désigné par l'aiguille est copié dans la nouvelle population.

L'inconvénient majeur de cette méthode repose sur le fait qu'un individu qui n'est pas le meilleur peut tout de même dominer la sélection. Cette méthode peut aussi engendrer une perte de diversité par la domination d'un individu. Un troisième inconvénient est sa faible performance vers la fin du processus, quand les individus se ressemblent (car la population est alors dominée par des individus qui ont des performances relatives similaires), [voir Dawid (2011) , p.39].

Dans le cadre de notre travail, on a utiliser le principe de sélection par tournoi. Brièvement, il existe d'autres méthodes de sélection, la plus connue étant celle du tournoi (*tournament selection*) : on tire deux individus aléatoirement dans la population et on reproduit le meilleur des deux dans la nouvelle population. On réitère cette procédure jusqu'à ce que la nouvelle population soit complète. Cette méthode donne de bons résultats. Toutefois, aussi important que soit la phase de sélection, elle ne crée pas de nouveaux individus dans la population. Ceci est le rôle des opérateurs de croisement et de mutation [Bäck (1995)].

Conclusion

Dans ce chapitre nous avons établi les bases fondamentales de l'algorithme génétique. Nous avons aussi présenté certaines de ses limitations et souligné l'intérêt du codage réel. Le cadre théorique proposé dans ce chapitre a déjà permis d'obtenir des résultats substantiels de convergence.

De nombreuses interrogations demeurent, cependant, on concernant les relations réelles entre les différents paramètres caractérisant l'algorithme génétique et les choix pratiques de ceux-ci. Dans ce domaine, la pratique devance encore la théorie.

Enfin, différentes versions continues de ces modèles discrets sont envisageables ; il s'agit là d'un axe de travail important qui nécessitera de développer de nouveaux outils.

CHAPITRE 4

Application

4.1 Wind power forecasting using neural network and ARIMA models (field of "Kabertene", in southern Algeria)

Introduction

The use of renewable energies and preservation of the environment was and is still an urgent requirement. Environment policies of governments are in this direction. Several studies have used $ARIMA$ (autoregressive integrated moving average) of Box and Jenkins Box et al. (2015), forecasting methodology, but this method was successful in linear models, so many researchers have resorted to the use of other methods, such as supported by the genetic algorithms Biffignandi (1984) ,Santini and Tettamanzi (2001),Lin et al. (2005) ,Rahal and Abdelkader (2017) and neural networks, which gave good results in particular in non-linear models Pisoni et al. (2009).

Research efforts on ANNs for forecasting are considerable. The literature is vast and growing. Hill et al. (1994) review the literature comparing ANNs with statistical models in time series forecasting and regression-based forecasting McCulloch and Pitts (1943). This work had further motivated us to explore the feasibility of neural networks for predicting of wind power to produce electricity. Conceptually, Both approaches are nonparametric techniques and are very similar in that they attempt to discover the appropriate internal representation of the time series data. Few assumptions are needed and no a priori postulation of models is required. Furthermore, by iteratively adjusting the weights in the modeling process, the autocorrelation between the data can be explored and better estimates can be obtained.

In this study, we use the data used in the average daily production for the field of wind energy in "Kabertene" (in southern Algeria), from 01/01/2015 to 31/12/2015.

4.1.1 Methodology

Kabertene wind farm is the first wind-farm in southern Algeria owned by SKTM subsidiary Sonalgaz, covering 33 hectares ; the site of the wind farm is located in Kabertene, 73 km north of the city of Adrar, the geographical coordinates are $28° 27' 27.44 N$,

fig. 4.1 – Wind Farm of Kabertene

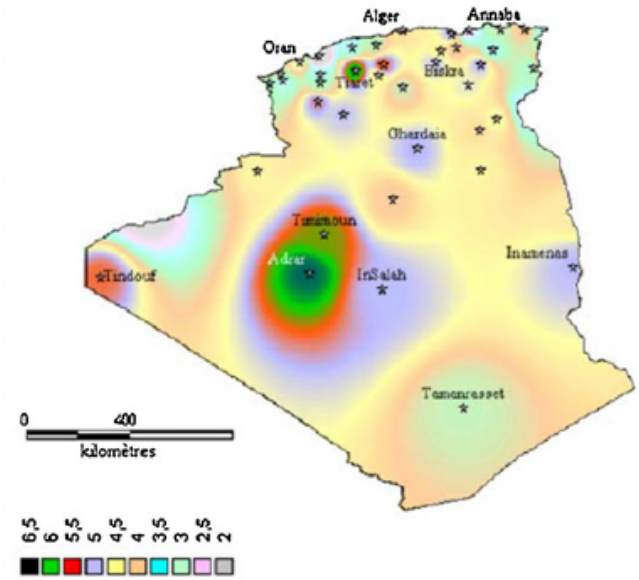

fig. 4.2 – Previous annual maps of wind speed in Algeria at 10 m height.

$0° \, 02' \, 59.08 \, W$. Cegelec, the French-Algerian consortium, made the best bid for construction with an offer of 7,257 DA per kilowatt-hour (roughly 0.07 euros).The 10,2MW wind farm comprises 12 Gamesa turbines, each of 850 kW with 12 m/s wind speed, the annual production target is about 33 GWh. The installation was completed in June 2014, see Mohamed et al. (2015). The decisive factors leading to the choice of site of the wind farm are : . Altitude ; . Barriers ; . Location relative to the wind direction ; . Location relative to the grid ; . Available area ; . Geography and geomorphology (soil quality, topography) ; . Access Opportunities ; . Road axis ; regulatory elements (sufficient distance from the extension of the city plan sufficient distance radar, Airport ...). This paper explores the potential application of the $ARIMA$ model and neural networks to this data. An overview of the essential features and some technical characteristics of the proposed methodologies are first discussed. This followed by a case study on the predicting of wind power to produce electricity and finally, the predictive of the proposed models are summarized.

4.1.2 ARIMA model

ARIMA models have been used in a great number of time series prediction problems because they are robust, as well as easy to understand and implement. However, difficulties exist with atypical values influencing the estimation of future values. A further disadvantage of stochastic models is generally their high order.

In the early 1970s, $ARIMA$ models were popularized by Box and Jenkins (1970), their names being associated with general ARIMA models applied to time series analysis and forecasting.

In an autoregressive integrated moving average model, the future value of a variable is assumed to be a linear function of several past observations and random errors. Denoted as $ARIMA(p, d, q)$ or expressed as :

$$y_t = \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{j=1}^{q} \theta_j e_{t-j} \tag{4.1}$$

or

$$\phi(B)(1 - B^d)y_t = \theta(B)e_t \tag{4.2}$$

where Y_t is the actual value ,$\{e_t\}$ is a white noise process with mean zero and variance σ^2, B is the backshift operator, $\phi(B)$ and $\theta(B)$ are polynomials of order p and q respectively. To ensure causality and invertibility, it is assumed that $\phi(B)$ and $\theta(B)$ have no roots for $|z| < 1$ [see Brockwell and Davis (2013)]. The seasonal $ARIMA(p, d, q)(P, D, Q)_s$ process is given by

$$\phi_p(B)\,\Phi_P(B^s)\,(1 - B)^d\,(1 - B^s)\,X_t = \theta_q(B)\,\Theta_Q(B^s)\,e_t \tag{4.3}$$

where $\Phi(B)$ and $\Theta(B)$ are polynomials of orders P and Q respectively, each containing no roots inside the unit circle. The most parsimonious model among all adequate ones is usually chosen as the final model for forecasting.

4.1.3 Neural network model

Artificial neural networks are forecasting methods that are based on simple mathematical models of the brain. They allow complex nonlinear relationships between the response variable and its predictors [Hyndman and Athanasopoulos (2014)]. Neural networks are mainly a nonlinear modeling approach that provides a fairly adequate approximation to any function [Ho et al. (2002)]. For time series forecasts, the model has the general form

$$y_t = \alpha_0 + \sum_{i=1}^{m} \alpha_i f\left(\sum_{j=1}^{n} \beta_{ij} y_{t-j} + \beta_{0j}\right) + \varepsilon_t \tag{4.4}$$

where n is the number of input nodes, m is the number of hidden nodes, f is a sigmoid transfer function such as the logistic : $f(x) = \frac{1}{(1+e^{-x})}$, $(\alpha_i)_{1\leq i\leq m}$ weights from the hidden to output nodes and $(\beta_{ij})_{1\leq i\leq m,\,1\leq j\leq n}$ are weights from the input to hidden nodes. $\alpha_0 = \beta_{i0} = 1$ from the bias terms and ε_t is the random error.

Often, neurons in the input layer have no activation function, their role being to transfer the inputs to the masked layer. A linear activation function is generally used for the output layer because the non-linear activation function can introduce a distortion of the predicted out-put [Khashei and Bijari (2010)].

The major research question we investigate is : What is the comparative advantage of neural networks to moving average model(MA) ?

To address this and other questions, we are conducting a simulation study with real data, the data used in the average daily production for the field of wind energy in " Kabertene "(in southern Algeria), to examine the problems and capabilities of the neural network model with respect to the above research questions.

4.1.4 Evaluation criteria for the uncertainty of WPF

Wind power forecasting has a character of the inherent uncertainty, which means that WPF cannot ever be exact. Therefore, it is essential that wind power forecasting is properly evaluated. It is very important to evaluate the error measures on data that have not been used to build the prediction model or to tune the method's parameters.

The evaluation methods on the uncertainty of wind power forecasting are listed as follows.

— The Mean Error(ME)

$$ME_k = \overline{e}_k = \frac{1}{N}\sum_{t=1}^{N} e_{t+k|t} \tag{4.5}$$

where :
$e_{t+k|t} = P_{t+k} - \widehat{P}_{t+k|t}$Is the error corresponding to time $t+k$ for the prediction made at time t ;
P_{t+k} is the measured power at time $t+k$;
$\widehat{P}_{t+k|t}$is the power forecast for time $t+k$ made at time t ;
N is the number of prediction errors used for method evaluation.

— The Mean Square Error(MSE)

$$MSE_k = \overline{e}_k^2 = \frac{1}{N}\sum_{t=1}^{N} {e^2}_{t+k|t} \tag{4.6}$$

— The Mean Absolute Error(MAE)

$$MAE_k = \frac{1}{N}\sum_{t=1}^{N} \left|e_{t+k|t}\right| \tag{4.7}$$

— The Mean Absolute Deviation(MAD)

$$MAD_k = \frac{1}{N}\sum_{t=1}^{N} \left|e_{t+k|t}\right| \tag{4.8}$$

4.1.5 Estimation models

The Wind-powered electricity production data (Figure 4.3) was modeled by the best models after residuals analysis for Box and Jenkins approach using $AICc$ and AIC criterions [see Hurvich and Tsai (1989),Cadenas et al. (2016)] is $ARIMA\,(0,0,1)$ with non-zero mean (Table 4.1) :

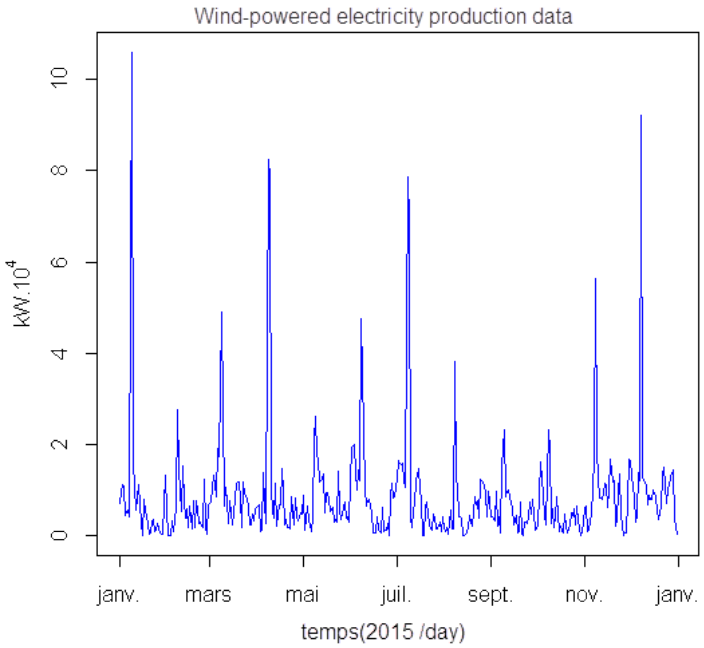

fig. 4.3 – electricity production

table 4.1 – Best ARIMA model

Model	*AIC Criterion*	*AICc Criterion*
$ARIMA(2,1,0)$ with non-zero mean	1118.722	1116.46
$ARIMA(0,0,0)$ with non-zero mean	1196.357	1195.77
$ARIMA(1,0,0)$ with non-zero mean	1123.064	1121.86
$ARIMA(0,0,1)$ **with non-zero mean**	**1114.419**	**1114.19**
$ARIMA(0,0,0)$ with zero mean	1342.742	1243.11
$ARIMA(1,0,1)$ with non-zero mean	1115.974	1114.76
$ARIMA(0,0,1)$ with zero mean	1199.316	1140.82

table 4.2 – Coefficient of the best model

Coefficient(s)	Estimate	s.e.
ma1	0.4797	0.0440
intercept	0.8764	0.0854

for the second method, the millstone network architecture, using the criterion AIC was $NNAR(2-2-1)$ (see Table 4.3 and Figure 4.4)

table 4.3 – Best architecture

ID	Architecture	Numbers of Weights	AIC	Correlation	R-Squared
1	[2-2-1]	**9**	**682,25003**	**0,51067**	**0,260648**
2	[2-6-1]	25	706,693569	0,555159	0,306628
3	[2-4-1]	17	695,021118	0,52665	0,275738
4	[2-5-1]	21	694,681547	0,558367	0,30614
5	[2-3-1]	13	686,048411	0,547912	0,300147

We used the following parameters : Input activation function=Logistic, Output error function=Sum-of-squares, Output activation function=Logistic.

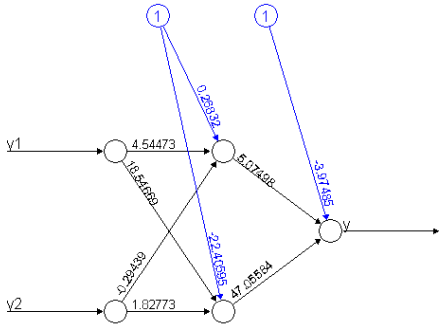

fig. 4.4 – $NNAR(2-2-1)$ architecture and weights

4.1.6 COMPARISONS

To compare the two models, we used last 85 values and the results are as in Figure (4.5).

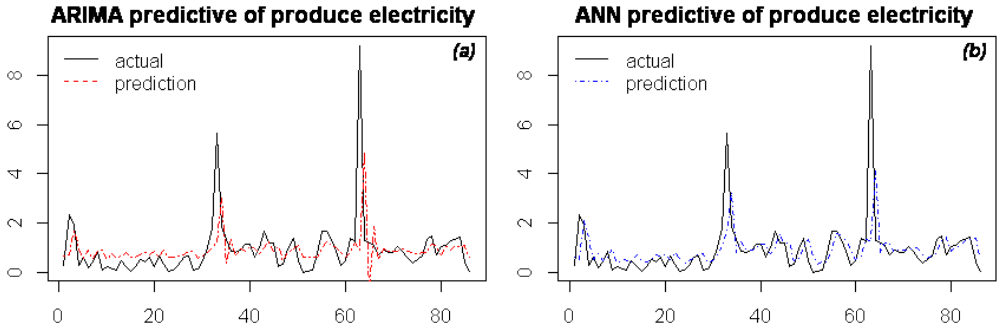

fig. 4.5 – (a) ARIMA prediction , (b) Neural network prediction.

We see that the $NNAR(2-2-1)$ (b) model is more performance than $ARIMA$ (a), and to confirm this result we use the estimation of differentiation criteria between the estimated models, the mean absolute error (MAE), root mean squared error ($RMSE$) and mean absolute deviation (MAD) [see Safi and White (2017), Khashei and Bijari (2010), Valenzuela et al. (2008), Qi and Zhang (2001), Zhang (2001), Chu and Zhang (2003)].

table 4.4 – Assessment criteria for differentiation between the two models.

MODEL	MAE	$RMSE$	MAD
$ARIMA(0,0,1)$	0.5808145	1.2025672	0.5544355
$NNAR(3-2-1)$	**0.5207443**	**1.1218845**	**0.4517646**

Considering this, we find that the performance of the neural network model is better than that of that the model of box and Jenkins for predicting the Wind-powered electricity. With a Mean Absolute Error (MAE) of 0.5808145, the Root Mean Square Error ($RMSE$) of 1.2025672, and median of the absolute deviations (MAD) of 0.5544355 on the validation data set, it is well-known that the performance of ANNs, with respect to forecasting, depends on a variety of factors [Dutta et al. (2006)]. Thus, the performance of ANN cannot be compared easily with the published findings of other studies.

Conclusion

Modeling some phenomenon despite all efforts by the statisticians, remains complicated and particularly to predict natural phenomena such as wind speed, that have a direct impact on the production of wind energy, in an attempt to estimate the model to predict the wind energy, we used 284 daily observations in a form of a time series beginning from January 1, 2015, until October 12, 2015, in the estimation stage and using 80 next daily observations from these data to test the model.

In this study, the neural network model is more appropriate than the model of box and Jenkins [Sim et al. (2014)] we showed differences between the criteria of comparison between the two models, and strengthened the capacity of networks of neurons in the modeling and prediction of nonlinear time series.

4.2 Application of GAs to determine the parameters of a time series model

Introduction

The study of time series or time series corresponds to the statistical analysis of observations equally spaced in time. For the majority of these phenomena, there is often a dependence time between observations, which has autoregressive modeling : the past is used to explain the present and predict the future. Model and predict a time series is assumed in the most cases, to make assumptions about its behavior using as it is non-deterministic series he'll have the random component "varies, but not too much." This condition will result in the "stationary " which implies a certain regularity of the process and allows to derive its asymptotic properties. When one has a series (X_i) in non-stationary stochastic,it should be modeled by a process $ARIMA(p, d, q)$, where d is the order of differentiation (or integration).

Having successfully transformed these data, the problem is to find a satisfactory $ARMA$ model and particularly as to determine p and q to find the autocorrelation functions and autocorrelation functions partial. The identification is mainly based on the analysis of the ACF (autocorrelation function) and $PACF$ (the *partial autocorrelation functions*) series considered.

The prediction method of *Box-Jenkins* [Box and Jenkins (1970),Vesin and Grüter (1999)] is particularly well suited to the treatment of series complex historical and other situations in which the Basic Law is not immediately apparent. However, as she deals with much more complicated situations, it is difficult to grasp the principles of this technique, as well as the limits of its application. In addition, the cost of *Box-Jenkins* method in a given situation is usually much higher than that of all other quantitative methods. We know that the applications of genetic algorithms are multiple : Medicine [Yang et al. (1998)], robotics[Zalzala and Fleming (1997)], analysis of time series[Mahfoud and Mani (1996),Packard (1990)], image processing[Chen et al. (1997)],$\cdots$.

Our application examples helped us realize that the coding of data for model a problem is complex. On the other hand, we also saw difficulties effectively choose good parameter for the various operators (mutation, crossover, selection, replace placement). Choices in relation to the operators themselves are also manageable, knowing that some are most appropriate to the problem and let correlation means that they d'optimiser.

We will study and model a real time series ; which represents the number of car accidents weekly between 1992 and 1995 (source : *Department of Statistics and O.R ,Faculty of science, Kuwait University*), using the method of Box and Jenkins, then we will hybridize the parameters obtained we use the genetic algorithms, and compare the results to make reliable forecasts.

4.2.1 Modeling and Prediction of the weekly car accidents data

Modeling using the Jenkiner method

The graphic (4.6) Series, which represents the number of car accidents weekly between 1992 and 1995.

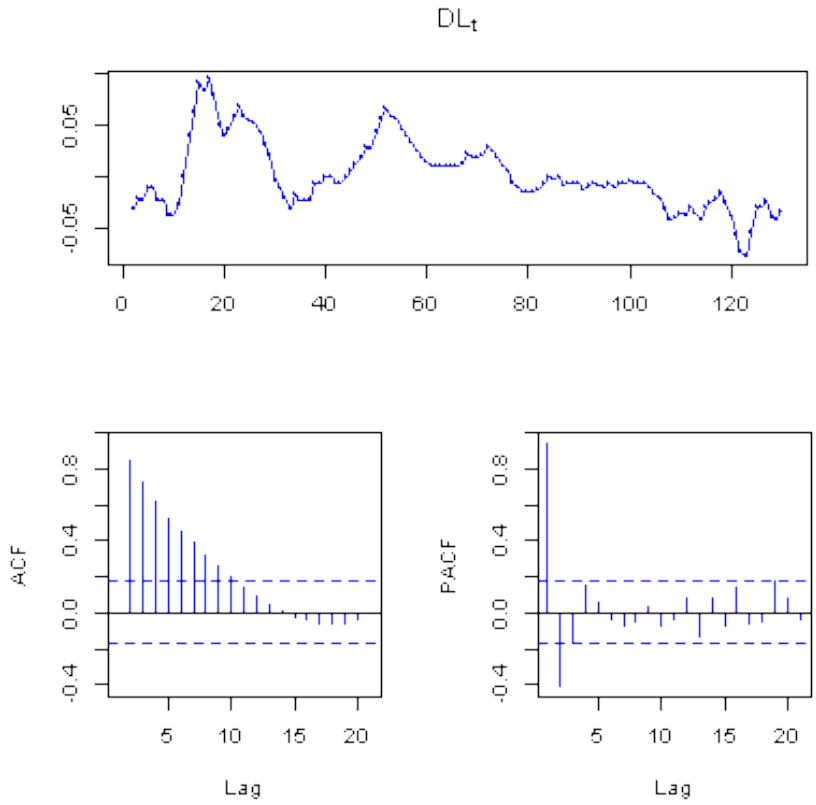

fig. 4.6 – Y_t , *ACF, PACF*

after the two transformations : $Lt = log(Y_t)$ and $DL = diff(L_t) = diff(log(Y_t))$ we obtain a stationary series $(DL) : p - value = 0 : 02066$ *test (adf.test (DL))*. Best suited to this model is $ARIM(2, 1, 0)$. Table (4.5) gives the coefficients of the model.

table 4.5 – Coefficient of the best model $ARIMA(2, 1, 0)$ of the series

Coefficient(s)	Estimate	Std. Error
ar1	1.4139	0.0767
ar2	−0.4827	0.0772

Move on to the adjustment of our series by this model. According to the graphic (4.7), there is no differentiation between the two series.
To validate the model we will study the residues.

table 4.6 – Tests residues

Test	Formula	$p - value$
Kolmogorov-Smirnovles	$lillie : test(residus)$	0.1994
Student t	$test(residus; mu = 0; conf : level = 0 : 95)$	0.8935
Ljung - Box	$Box.test(residus; lag = 50; type = "LjungBox")$	0.9477

This allows us to conclude that the residues form a white noise (Table4.6), where model validation. We can have confidence in our predictions.

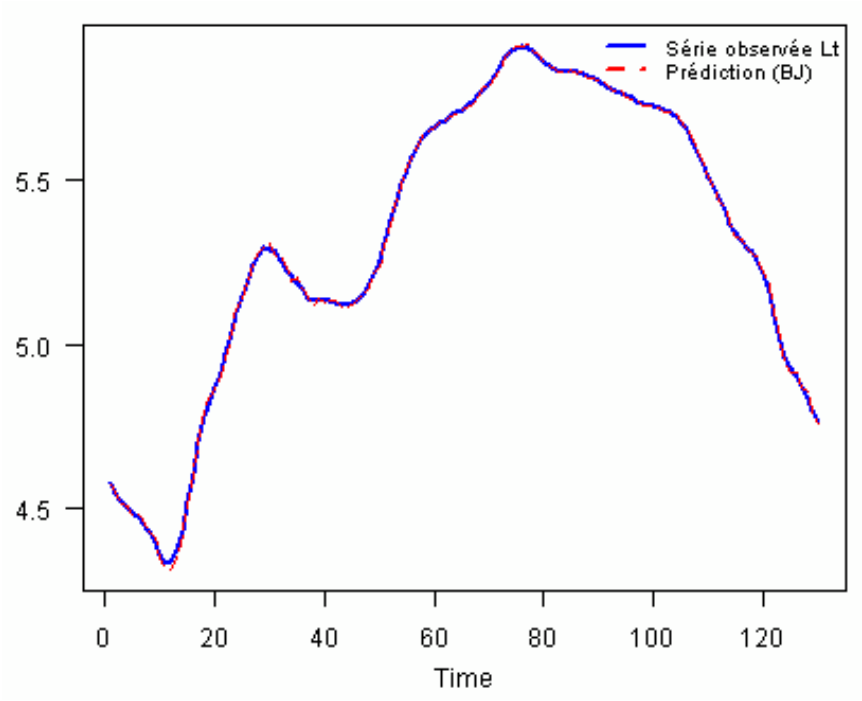

fig. 4.7 – The adjustment of the series by the model $ARIMA(2,1,0)$

Hybridization of the model using genetic algorithms

After obtaining the model, we used genetic algorithms to obtain the best parameters of the model, where the parameters represent the chromosomes (we used the real coding). The best results are :
Best (ϕ_1 *and* ϕ_2), Which was obtained under the conditions (Table 4.7) :

$$(\phi_1\ ,\ \phi_2) = (1.3623294021\ ,\ 0.4199999997)$$

table 4.7 – The conditions for obtaining the best result ($\phi_1$and ϕ_2)

NB of Iterations = 100	the initial population size
NB of Iterations = 1000	nombre d'iteration of genetic algorithms
Probacroisement =90 %	crossing probability
Probamutation =10 %	probability of mutation
$\varepsilon = 10^{-6}$	stop criterion : $mse < \varepsilon$

The adjusted by the model and the real series are graphically almost identical graph(4.8).

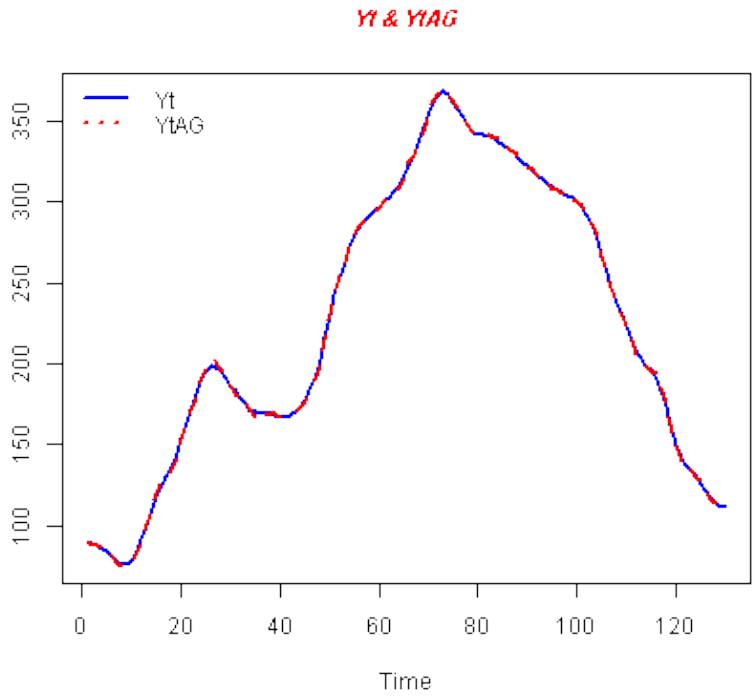

fig. 4.8 – Y_t and Y_t adjusted by GAs

To validate the model we will study the residues. This enables us to conclude that

table 4.8 – Tests residues

Test	Formula	$p-value$
Kolmogorov-Smirnovles	$lillie : test(residus)$	0.3175
Student t	$test(residus; mu = 0; conf : level = 0 : 95)$	0.88
Ljung - Box	$Box.test(residus; lag = 50; type = "LjungBox")$	0.6273

the residues form a white noise (Table 4.8), where model validation.
The graphic (4.9) summarizes the quality of prediction and hence the performance of both models. Visually, there is a clear difference between the qualities of predictions, where the second validation Hybrid model(AG-BJ).

table 4.9 – comparison between BJ model and Hybrid model(AG-BJ)

	MAPE	*MSE*
AG-BJ	1.158333	1.231707
B-J	1.160062	1.233228

4.3 Conclusion

In this study, we found that the use of a hybrid model between genetic algorithms and that of Box-Jenkins makes it possible to obtain very good results compared to the results given by the Box-Jenkins method (in particular,for the real chronicle). Therefore, the conclusion of this study is not only proposal to continue the way of

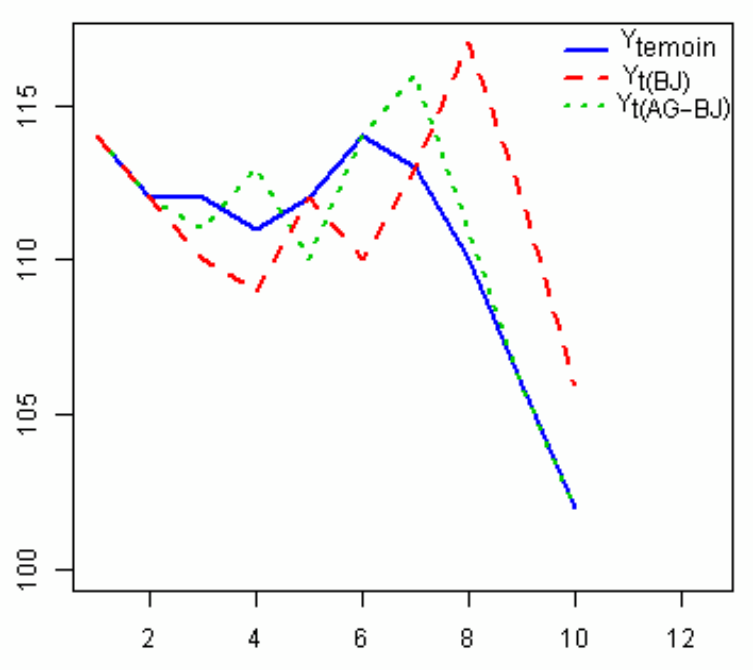

fig. 4.9 – Performance of both *BJ* versus Hybrid model(*AG-BJ*)

using Genetic Algorithms in the field of time series (parameter estimation), and thus predict, but we can say that genetic algorithms alone are not very effective in solving a problem. However, they provide (GAs) fairly quickly an acceptable solution. Nevertheless, it is possible to improve effectively enough by combining it with a deterministic algorithm.

Conclusion et Perspectives

Dans cette thèse nous avons présenté un panorama des méthodes de prévision des séries temporelles. Nous sommes partis des méthodes classiques de modélisations, nous avons montré que ces méthodes, malgré leur simplicité, sont moins efficaces dans les modèles non linéaires, pour arriver aux approches basés sur l'intelligence artificielle, en particulier sur les réseaux de neurones et les algorithmes génétiques.

Nous avons montré l'intérêt de ces approches par des résultats expérimentaux et nous avons apporté des explications théoriques à ces résultats.

L'étude de l'état de l'art nous a permis de constater l'importance des *RNN* dans les problématiques de prévision. En partant de ce constat, nous souhaitons évaluer la contribution des méta-méthodes. Ce fut le cas pour la méthode du *MLP*, qui a été testée et a montré son impact positif sur la prévision en utilisant des modèles non linéaires.

Les expérimentations qui ont été menées nous ont permis de déduire l'importance de l'architecture des réseaux de neurones (nombre de neurones, nombre de couches, ..., etc.), nous avons utilisé le critère de *AIC* pour résoudre ce problème, cette méthode à donnée un avantage au réseaux de neurones par rapport au modèle *ARMA*.

Pour l'approche d'algorithmes génétiques que nous avons étudier, a montrer que les résultats sont presque identiques pour des modèles linaires simules , C'est la raison qui nous a poussés à proposer une combinaison entre la méthode de *Box- Jenkins* et les *AGs* ; premièrement nous avons modéliser la série par la première méthode, puis améliorer les paramètres obtenu par les *AGs*.

Perspectives de recherche

Dans les perspectives de notre travail, plusieurs axes de recherche nous semblent prometteurs : Nous sommes conscients que les travaux réalisés au cours de cette thèse comportent des limites. Pour cela nous envisageons des techniques complémentaires qui pourraient améliorer nos résultats. Nous présentons dans ce qui suit une discussion ouvrant la voie sur quelques perspectives de recherche.

Pour la méthode proposée dans la partie expérimentale , nous pourrons poursuivre cette idée, on utilisant une hybridation entre les algorithmes génétiques et les réseaux de neurones.

Nous pouvons envisager d'autres méthodes heuristiques pour la sélection d'attributs inspirées de la nature telles que les colonies de fourmis [Dorigo et al. (2002)] ou les essaims de particules [Kennedy et al. (2001)]. Une voie de recherche promise à un grand avenir est celle fondée sur les algorithmes génétiques non restreints, appelés « programmation génétique », pour rechercher des formes fonctionnelles reproduisant le mieux une série de données.

Bibliographie

Aburto, L. and Weber, R. (2007). Improved supply chain management based on hybrid demand forecasts. *Applied Soft Computing*, 7(1) :136–144.

Alliot, J.-M. (1998). Techniques d'optimisation stochastique appliquéesa certains problemes du trafic aérien. *Rapport de Recherche, Ecole Nationale de l'Aviation Civile, Toulouse*, 27.

Alliot, J.-M. and Durand, N. (2005). Algorithmes génétiques. *Centre d'Etudes de la Navigation Aérienne*.

Argon, Y. (2009). *Series Temporelles*. Ed. Techniques Ingénieur.

Atiya, A. F., El-Shoura, S. M., Shaheen, S. I., and El-Sherif, M. S. (1999). A comparison between neural-network forecasting techniques-case study : river flow forecasting. *IEEE Transactions on neural networks*, 10(2) :402–409.

Axelrod, R. et al. (1987). The evolution of strategies in the iterated prisoner's dilemma. *The dynamics of norms*, pages 1–16.

Back, A. D. and Tsoi, A. C. (1991). Fir and iir synapses, a new neural network architecture for time series modeling. *Neural Computation*, 3(3) :375–385.

Back, T. (1993). Optimal mutation rates in genetic search. In *Proceedings of the fifth international conference on genetic algorithms*, pages 2–8. Morgan Kaufmann, San Mateo, CA.

Bäck, T. (1995). Generalized convergence models for tournament- and (mu, lambda)-selection. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 2–8, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Bäck, T. and Schwefel, H.-P. (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary computation*, 1(1) :1–23.

Balkin, S. D. and Ord, J. K. (2000). Automatic neural network modeling for univariate time series. *International Journal of Forecasting*, 16(4) :509–515.

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3) :930–945.

Benoudjit, N. and Verleysen, M. (2003). On the kernel widths in radial-basis function networks. *Neural Processing Letters*, 18(2) :139–154.

Bera, A. K., Jarque, C. M., and Lee, L.-F. (1984). Testing the normality assumption in limited dependent variable models. *International Economic Review*, pages 563–578.

Besse, P. (2009). Apprentissage statistique & data mining. *Toulouse : INSA*.

Biffignandi, S. (1984). Short term forecasting : An introduction to the box-jenkins approacn odonovan. *Journal of Forecasting*, 3(4) :466–467.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3) :307–327.

Boné, R. (2000). *Réseaux de neurones récurrents pour la prévision de séries temporelles*. PhD thesis. Thèse de doctorat dirigée par Asselin de Beauville, Jean-Pierre Informatique Tours 2000.

Boné, R., Thillier, R., Yvon, F., and de Beauville, J. A. (1998). Optimization by a genetic algorithm of stochastic linear models of time series. In *Bio-Mimetic Approaches in Management Science*, pages 153–162. Springer.

Bourazza, S. (2006). *Variantes d'algorithmes génétiques appliquéees aux problèmes d'ordonnancement*. PhD thesis, Université du Havre.

Bourbonnais, R. and Usunier, J.-C. (2013). *Prévision des ventes : théorie et pratique*. Economica.

Box, G. E. and Jenkins, G. M. (1976). *Time series analysis : forecasting and control*. Holden-Day.

Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time series analysis : forecasting and control*. John Wiley & Sons.

Box, G. E. and Pierce, D. A. (1970). Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American statistical Association*, 65(332) :1509–1526.

Box, G. E. P. and Jenkins, G. M. (1970). Time series analysis : forecasting and control, 1976. *ISBN : 0-8162-1104-3*.

Bridges, C. L. and Goldberg, D. E. (1987). An analysis of reproduction and crossover in a binary-coded genetic algorithm. *Grefenstette*, 878 :9–13.

Brockwell, P. J. and Davis, R. A. (2013). *Time series : theory and methods*. Springer Science & Business Media.

Brodmeier, T. and Pretsch, E. (1994). Application of genetic algorithms in molecular modeling. *Journal of Computational Chemistry*, 15(6) :588–595.

Bui, T. N. and Moon, B.-R. (1994). Analyzing hyperplane synthesis in genetic algorithms using clustered schemata. In *International Conference on Parallel Problem Solving from Nature*, pages 108–118. Springer.

Cadenas, E., Rivera, W., Campos-Amezcua, R., and Heard, C. (2016). Wind speed prediction using a univariate arima model and a multivariate narx model. *Energies*, 9(2) :109.

Caruana, R. A. and Schaffer, J. D. (1988). Representation and hidden bias : Gray vs. binary coding for genetic algorithms. In *Machine Learning Proceedings 1988*, pages 153–161. Elsevier.

Catoni, O. (1990). *Large deviations for annealing*. PhD thesis, University Paris XI.

Cerf, R. (1994). *Une théorie asymptotique des algorithmes génétiques*. PhD thesis, Montpellier 2.

Cerf, R. (1996a). The dynamics of mutation-selection algorithms with large population sizes. In *Annales-Institut Henri Poincare Probabilities et Statistiques*, volume 32, pages 455–508. Gauthier-Villars.

Cerf, R. (1996b). A new genetic algorithm. *The Annals of Applied Probability*, pages 778–817.

Chan, W. and Tong, H. (1986). On tests for non-linearity in time series analysis. *Journal of forecasting*, 5(4) :217–228.

Charpentier, A. (2006). Cours de séries temporelles : théorie et applications. *Université Paris Dauphine*.

Chen, Y.-W., Nakao, Z., Arakaki, K., and Tamura, S. (1997). Blind deconvolution based on genetic algorithms. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 80(12) :2603–2607.

Chikr El Mezouar, Z. and Gasmi, L. (2015). Application of gas to determine the parameters of a time series model. *International Journal of Scientific Research and Reviews*, 4(1) :31 – 40.

Chouchani, I. (2010). Utilisation d'un algorithme génétique pour la composition de services web.

Chu, C.-W. and Zhang, G. P. (2003). A comparative study of linear and nonlinear models for aggregate retail sales forecasting. *International Journal of production economics*, 86(3) :217–231.

Cigizoglu, H. K. (2004). Estimation and forecasting of daily suspended sediment data by multi-layer perceptrons. *Advances in Water Resources*, 27(2) :185–195.

Connor, J. T., Martin, R. D., and Atlas, L. E. (1994). Recurrent neural networks and robust time series prediction. *IEEE transactions on neural networks*, 5(2) :240–254.

Cottrell, M., Girard, B., Girard, Y., Mangeas, M., and Muller, C. (1995). Neural modeling for time series : a statistical stepwise method for weight elimination. *IEEE transactions on neural networks*, 6(6) :1355–1364.

Crucianu, M. (1997). Algorithmes d'évolution pour les réseaux de neurones. *Rapport de recherche*, 187.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4) :303–314.

Davis, T. E. (1991). A simulated annealing like convergence theory for the simple genetic algorithm. In *Proc. the 4th International Conference on Genetic Algorithms*, pages 174–181.

Dawid, H. (2011). *Adaptive learning by genetic algorithms : Analytical results and applications to economic models*. Springer Science & Business Media.

De Gooijer, J. G. and Kumar, K. (1992). Some recent developments in non-linear time series modelling, testing, and forecasting. *International Journal of Forecasting*, 8(2) :135–156.

De Jong, K. (1980). Adaptive system design : a genetic approach. *IEEE Transactions on Systems, Man, and Cybernetics*, 10(9) :566–574.

Dorigo, M., Gambardella, L. M., Middendorf, M., and Stutzle, T. (2002). Guest editorial : special section on ant colony optimization. *IEEE Transactions on evolutionary computation*, 6(4) :317–319.

Dréo, J., Pétrowski, A., Siarry, P., and Taillard, E. (2003). *Métaheuristiques pour l'optimisation difficile*. Eyrolles.

Durand, N. (1996). *Optimisation de trajectoires pour la résolution de conflits aériens en route*. PhD thesis, INPT.

Durand, N. (2004). *Algorithmes Génétiques et autres méthodes d'optimisation appliqués à la gestion de trafic aérien*. PhD thesis, INPT.

Dutta, G., Jha, P., Laha, A. K., and Mohan, N. (2006). Artificial neural network models for forecasting stock price index in the bombay stock exchange. *Journal of Emerging Market Finance*, 5(3) :283–295.

Elman, J. L. (1989). Representation and structure in connectionist models. Technical report, California Univ.San Diego La Jolla Center for Rechearch in Language.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica : Journal of the Econometric Society*, pages 987–1007.

Fogel, L. J., Owens, A. J., and Walsh, M. J. (1966). Artificial intelligence through simulated evolution.

Fredlin, M. and Wentzell, A. (1984). Random perturbations of dynamical systems springer. *New York*.

Freidlin, M. I. and Wentzell, A. D. (1998). Random perturbations. In *Random Perturbations of Dynamical Systems*, pages 15–43. Springer.

Gingras, D. F. and Gerstoft, P. (1995). Inversion for geometric and geoacoustic parameters in shallow water : Experimental results. *The Journal of the Acoustical Society of America*, 97(6) :3589–3598.

Goldberg, D. (1989a). Genetic algorithms in search, optimization, and machine learning. *Reading, MA : Addison-Wesley*.

Goldberg, D. E. (1989b). Genetic algorithms and walsh functions : Part i and 2, a gentle introduction. *Complex systems*, 3(2) :129–171.

Goldberg, D. E. (1991). Real-coded genetic algorithms, virtual alphabets, and blocking. *Complex systems*, 5(2) :139–167.

González-Romera, E., Jaramillo-Morán, M., and Carmona-Fernández, D. (2008). Monthly electric energy demand forecasting with neural networks and fourier series. *Energy Conversion and Management*, 49(11) :3135–3142.

Gorr, W. L. (1994). Research prospective on neural network forecasting. *International Journal of Forecasting*, 10(1) :1–4.

Granger, C. W. (1993). Strategies for modelling nonlinear time-series relationships. *Economic Record*, 69(3) :233–238.

Haykin, S. (1998). *Neural Networks : A Comprehensive Foundation*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition.

Haykin, S. (1999a). Neural networks : a comprehensive foundation. *Prentice-Hall*.

Haykin, S. (1999b). Neural networks : A comprehensive foundation, vol. 2. *Segundo, Pentrice Hall. Espa ña*.

Haykin, S. (2010). Neural networks : a comprehensive foundation, 1999. *Mc Millan, New Jersey*.

Hill, T., Marquez, L., O'Connor, M., and Remus, W. (1994). Artificial neural network models for forecasting and decision making. *International journal of forecasting*, 10(1) :5–15.

Ho, S., Xie, M., and Goh, T. (2002). A comparative study of neural network and box-jenkins arima modeling in time series prediction. *Computers & Industrial Engineering*, 42(2) :371–375.

Holland, J. H. (1962). Outline for a logical theory of adaptive systems. *Journal of the ACM (JACM)*, 9(3) :297–314.

Holland, J. H. (1975). Adaptation in natural and artificial systems. an introductory analysis with application to biology, control, and artificial intelligence. *Ann Arbor, MI : University of Michigan Press*, pages 439–444.

Hoptroff, R. G. (1993). The principles and practice of time series forecasting and business modelling using neural nets. *Neural Computing & Applications*, 1(1) :59–66.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2) :251–257.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5) :359–366.

Hurvich, C. M. and Tsai, C.-L. (1989). Regression and time series model selection in small samples. *Biometrika*, pages 297–307.

Hyndman, R. J. and Athanasopoulos, G. (2014). *Forecasting : principles and practice*. OTexts.

Jacques, J. (2013). Introduction aux séries temporelles. *Polytech Lille*.

Jamshidi, M., Krohling, R. A., Coelho, L. d. S., and Fleming, P. J. (2002). *Robust control systems with genetic algorithms*, volume 3. CRC Press.

Jordan, M. (1986). Attractor dynamics and parallelism in a connectionist sequential machine. In *Eighth Annual Conference of the Cognitive Science Society, 1986*, pages 513–546.

Kaastra, I. and Boyd, M. (1996). Designing a neural network for forecasting financial and economic time series. *Neurocomputing*, 10(3) :215–236.

Karunasinghe, D. S. and Liong, S.-Y. (2006). Chaotic time series prediction with a global model : Artificial neural network. *Journal of Hydrology*, 323(1-4) :92–105.

Kassam, S. A. and Cha, I. (1993). Radial basis function networks in nonlinear signal processing applications. In *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, pages 1021–1025. IEEE.

Kelo, S. and Dudul, S. (2012). A wavelet elman neural network for short-term electrical load prediction under the influence of temperature. *International Journal of Electrical Power & Energy Systems*, 43(1) :1063–1071.

Kennedy, J., Eberhart, R. C., and Shi, Y. (2001). Swarm intelligence. 2001. *Kaufmann, San Francisco*, 1 :700–720.

Khashei, M. and Bijari, M. (2010). An artificial neural network (p, d, q) model for time series forecasting. *Expert Systems with applications*, 37(1) :479–489.

Kouam, A. (1993). *Approches connexionnistes pour la prevision des séries temporelles*. PhD thesis. Thèse de doctorat dirigée par Fogelman, Francoise Sciences et techniques communes Paris 11 1993.

Koza, J. R. (1992). *Genetic Programming II, Automatic Discovery of Reusable Subprograms*. MIT Press, Cambridge, MA.

Krishnakumar, K. and Goldberg, D. E. (1992). Control system optimization using genetic algorithms. *Journal of Guidance, Control, and Dynamics*, 15(3) :735–740.

Lagnoux, A. (1996). Séries chronologiques. *ISMAG, Master1-MI00141X, Université de Toulouse Le Mirail, 53p.*

Lapedes, A. and Farber, R. (1987). Nonlinear signal processing using neural networks : Prediction and system modelling. Technical report.

Lin, L., Cao, L., and Zhang, C. (2005). Genetic algorithms for robust optimization in financial applications. In *IASTED International Conference on Computational Intelligence*. ACTA Press.

Liou, C.-Y., Huang, J.-C., and Yang, W.-C. (2008). Modeling word perception using the elman network. *Neurocomputing*, 71(16-18) :3150–3157.

Ljung, G. M. and Box, G. E. (1978). On a measure of lack of fit in time series models. *Biometrika*, 65(2) :297–303.

MacKinnon, J. G. (1994). Approximate asymptotic distribution functions for unit-root and cointegration tests. *Journal of Business & Economic Statistics*, 12(2) :167–176.

Mahfoud, S. and Mani, G. (1996). Financial forecasting using genetic algorithms. *Applied artificial intelligence*, 10(6) :543–566.

Makridakis, S., Andersen, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., Newton, J., Parzen, E., and Winkler, R. (1982). The accuracy of extrapolation (time series) methods : Results of a forecasting competition. *Journal of forecasting*, 1(2) :111–153.

Marco, N., Godart, C., Désidéri, J.-A., Mantel, B., and Périaux, J. (1996). *A genetic algorithm compared with a gradient-based method for the solution of an active-control model problem.* PhD thesis, INRIA.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4) :115–133.

Michalewicz, Z. (1994). Evolutionary computation techniques for nonlinear programming problems. *International Transactions in Operational Research*, 1(2) :223–240.

Michalewicz, Z. and Janikow, C. Z. (1991). Handling constraints in genetic algorithms. In *ICGA*, pages 151–157.

Mitchell, T. M. (1997). *Machine Learning.* McGraw-Hill, Inc., New York, NY, USA, 1 edition.

Mohamed, Benmbarek, M., Fadela, B., and Mounir, K. (2015). Optimization of the wind turbines location in kaberten wind farm in algeria. *Energy Procedia*, 74 :122–129.

Moody, J. (1995). Economic forecasting : Challenges and neural network solutions.

Mori, H. and Urano, S. (1996). Short-term load forecasting with chaos time series analysis. In *Intelligent Systems Applications to Power Systems, 1996. Proceedings, ISAP'96., International Conference on*, pages 133–137. IEEE.

Nix, A. E. and Vose, M. D. (1992). Modeling genetic algorithms with markov chains. *Annals of mathematics and artificial intelligence*, 5(1) :79–88.

Orr, M. (1998). Optimising the widths of radial basis functions. In *Neural Networks, 1998. Proceedings. Vth Brazilian Symposium on*, pages 26–29. IEEE.

Packard, N. H. (1990). A genetic learning algorithm for the analysis of complex data. *Complex Systems*, 4(5) :543–572.

Palmer, R., Arthur, W. B., Holland, J. H., and LeBaron, B. (1999). An artificial stock market. *Artificial Life and Robotics*, 3(1) :27–31.

Pan, Z., LishanKang, Y., Liu, G., et al. (1995). Parameter estimation by genetic algorithms for nonlinear regression. *High Technology*, 946 :953.

Piper, J. (1995). Genetic algorithm for applying constraints in chromosome classification. *Pattern Recognition Letters*, 16(8) :857–864.

Pisoni, E., Farina, M., Carnevale, C., and Piroddi, L. (2009). Forecasting peak air pollution levels using narx models. *Engineering Applications of Artificial Intelligence*, 22(4) :593–602.

Qi, M. and Zhang, G. P. (2001). An investigation of model selection criteria for neural network time series forecasting. *European Journal of Operational Research*, 132(3) :666–680.

Rahal, R., C. E. Z. and Abdelkader, G. (2017). Forecasting performance of arma models by correction errors with genetics algorithms. *International Journal of Statistics & Economics*, 8(1) :1–15.

Rahmat-Samii, Y. and Michielssen, E. (1999). Electromagnetic optimization by genetic algorithms. *Microwave Journal*, 42(11) :232–232.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1985). Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Rumelheat, D. (1986). Learning internal representations by error propagation. *Parallel distributed processing*, 1 :319–362.

Safi, S. K. and White, A. K. (2017). Short and long-term forecasting using artificial neural networks for stock prices in palestine : a comparative study. *Electronic Journal of Applied Statistical Analysis*, 10(1) :14–28.

Samanta, B. and Nataraj, C. (2008). Prognostics of machine condition using soft computing. *Robotics and Computer-Integrated Manufacturing*, 24(6) :816–823.

Santini, M. and Tettamanzi, A. (2001). Genetic programming for financial time series prediction. In *European Conference on Genetic Programming*, pages 361–370. Springer.

Saxén, H. (1996). Nonlinear time series analysis by neural networks : a case study. *International journal of neural systems*, 7(02) :195–201.

Sheta, A. F. and De Jong, K. (2001). Time-series forecasting using ga-tuned radial basis functions. *Information Sciences*, 133(3-4) :221–228.

Sim, J.-J., Tan, G. W.-H., Wong, J. C., Ooi, K.-B., and Hew, T.-S. (2014). Understanding and predicting the motivators of mobile music acceptance–a multi-stage mra-artificial neural network approach. *Telematics and Informatics*, 31(4) :569–584.

Smith, R. E., Goldberg, D. E., and Earickson, J. A. (1991). Sga-c : A c-language implementation of a simple genetic algorithm.

Song, Q. (2011). Robust initialization of a jordan network with recurrent constrained learning. *IEEE transactions on neural networks*, 22(12) :2460–2473.

Spears, W. M. and De Jong, K. A. (1991). An analysis of multi-point crossover. In *Foundations of genetic algorithms*, volume 1, pages 301–315. Elsevier.

Tenti, P. (1996). Forecasting foreign exchange rates using recurrent neural networks. *Applied Artificial Intelligence*, 10(6) :567–582.

Trouve, A. (1993). *Parallélisation massive du recuit simulé*. PhD thesis, Paris 11.

Valenzuela, O., Rojas, I., Rojas, F., Pomares, H., Herrera, L. J., Guillén, A., Marquez, L., and Pasadas, M. (2008). Hybridization of intelligent techniques and arima models for time series prediction. *Fuzzy Sets and Systems*, 159(7) :821–845.

Venables, W. N. and Smith, D. M. (2009). R development core team (2008) an introduction to r : Notes on r, a programming environment for data analysis and graphics.

Vesin, J.-M. and Grüter, R. (1999). Model selection using a simplex reproduction genetic algorithm. *Signal Processing*, 78(3) :321–327.

von Sachs, R. and Van Bellegem, S. (2005). Séries chronologiques.

Vose, M. D. (1991). Generalizing the notion of schema in genetic algorithms. *Artificial Intelligence*, 50(3) :385–396.

Wang, D. and Lu, W.-Z. (2006). Forecasting of ozone level in time series using mlp model with a novel hybrid training algorithm. *Atmospheric Environment*, 40(5) :913–924.

Wedding II, D. K. and Cios, K. J. (1996). Time series forecasting by combining rbf networks, certainty factors, and the box-jenkins model. *Neurocomputing*, 10(2) :149–168.

Weigend, A. S., H. B. A. and Rumelhart, D. E. (1992). Predicting sunspots and exchange rates with connectionist networks. in casdagli, m. et eubank, s. *Nonlinear modeling and forecasting*, 12 :395–432.

Wright, A. H. (1991). Genetic algorithms for real parameter optimization. In *Foundations of genetic algorithms*, volume 1, pages 205–218. Elsevier.

Yang, G., Reinstein, L., Pai, S., Xu, Z., and Carroll, D. (1998). A new genetic algorithm technique in optimization of permanent 125i prostate implants. *Medical Physics*, 25(12) :2308–2315.

Yves, A. (2011). Séries temporelles avec r : Méthodes et cas.

Zalzala, A. M. and Fleming, P. J. (1997). *Genetic algorithms in engineering systems*, volume 55. Iet.

Zeiri, Y. (1995). Prediction of the lowest energy structure of clusters using a genetic algorithm. *Physical Review E*, 51(4) :R2769.

Zhang, G., Patuwo, B. E., and Hu, M. Y. (1998). Forecasting with artificial neural networks : : The state of the art. *International journal of forecasting*, 14(1) :35–62.

Zhang, G. P. (2001). An investigation of neural networks for linear time-series forecasting. *Computers & Operations Research*, 28(12) :1183–1202.

Zhang, G. P. (2003). Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50 :159–175.

Zhang, G. P., Patuwo, B. E., and Hu, M. Y. (2001). A simulation study of artificial neural networks for nonlinear time-series forecasting. *Computers & Operations Research*, 28(4) :381–396.

Zhang, G. P. and Qi, M. (2005). Neural network forecasting for seasonal and trend time series. *European journal of operational research*, 160(2) :501–514.

Zhou, H., Su, G., and Li, G. (2011). Forecasting daily gas load with oihf-elman neural network. *Procedia Computer Science*, 5 :754–758.

Zilinskas, A. and Zhigljavsky, A. (1991). Methods of the global extreme searching.