الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

Université Mohamed Khider – Biskra

Faculté des Sciences et de la technologie

Département : Génie Electrique

Ref :……

جامعة محمد خيضر بسكرة
كلية العلوم و التكنولوجيا
قسم:الهندسة الكهربائية
المرجع:.....

Thèse présentée en vue de l'obtention
Du diplôme de

Doctorat en sciences

Option : Electronique

La Prédiction des Séries Temporelles utilisant les Paradigmes de Soft Computing

Présentée par :

Raihane Mechgoug

Soutenue publiquement le : 24/06/2013

Devant le jury composé de :

Président	**:S. M. MIMOUNE**	**Professeur**	**Université de Biskra**
Directeur de thèse	**:A. TITAOUINE**	**Professeur**	**Université de Biskra**
Examinateur	**:K .CHAFAA**	**Maître de conférences**	**Université de Batna**
Examinateur	**: A.AISSI**	**Maître de conférences**	**Université de Batna**
Examinateur	**:L. ABDOU**	**Maître de conférences**	**Université de Biskra**
Examinateur	**:L.BENALIA**	**Maître de conférences**	**Université de M'sila**

Je dédie ce travail
À la mémoire de mon père
À Si-Lmebarek
À Chikch
À lui seul

Remerciements

*Je tiens en tout premier lieu à remercier mon directeur de thèse Monsieur : **TITAOUINE A .Nacer** Professeur d'université de Biskra. C'est un agréable plaisir pour moi de l'exprimer mes très vifs remerciements d'avoir accepté de diriger ce travail et de me faire profiter de son expérience.*

Je remercie sincèrement, les membres de jury d'avoir bien voulu accepter de faire partie de la commission d'examinateur.

Je remercie également ma mère, Mes frères et mes sœurs pour leur soutien précieux.

Enfin, je remercie mon cher époux pour son soutien, son aide et sa patience tout au long de la thèse.

Table des matières

V. La conception de prédicteur neuronal par les Algorithmes génétiques réels

Liste des figures

Liste des tableaux

Liste des publications

R. Mechgoug, N. Titaouine, Exchange Rate Prediction using Neural–Genetic Model, Journal of Electrical and Electronics Engineering, Volume 5, Number 2, October 2012

R. Mechgoug, N. Golea and A. Taleb-Ahmed, Times Series Prediction to Basis of a Neural Network Conceived by a Real Genetic Algorithm, Journal Computer Technology and Application, Volume2, Numbre1, 2011

L. Cherroun, R. Mechgoug, M. Boumahrez., Path Following Behavior for an Autonomous Mobile Robot using Fuzzy Logic and Neural Network ‘,Courier du Savoir – N°12, Octobre 2011, pp.63-70.

Liste des conférences

R. Mechgoug, A. Merzouki, la prédiction des séries temporelles à base d’un système flou conçu par algorithme génétique réel Time, Conférence sur le génie, CGE’ 125-26 Décembre 2001, Génie électrique, Borjel Bahri, Algérie

R. Mechgoug, N. Golea, Time series forecasting with fuzzy genetic algorithms, the International Arab Conference on Information Technology ,ACIT2004, Mentouri University, Algeria

L. Cherroun, R. Mechgoug, M. Boumahrez., Path Following Behavior for an Autonomous Mobile Robot using Fuzzy Logic and Neural Network ‘, The Second International Conference on Image and Signal Processing and their Applications, ISPA2010,Biskra, Algeria.

L . Cherroun, R. Mechgoug, Neuro-Fuzzy Controller For The Path Following Behavior And Moving Target Pursing By A Mobile Robot, 1st International Conference on Automation and Mechatronics, CIAM’2011, Oran, November 2011.

R. Mechgoug, A. Taleb Ahmed, and L. Cherroun, Optimization of Neural Predictor for Air Pollution, Proceedings of the World Congress on Engineering , Vol II, WCE 2012, London, U.K.

L . Cherroun, R. Mechgoug, Soft Computing Techniques for the Path Following Behavior and Moving Target Pursing by a Mobile Robot, 2nd International Conference on Information Systems and Technologies 2012 (ICIST’2012) March 24-26 Sousse, Tunisia.

La Prédiction des Séries Temporelles Utilisant les Paradigmes de Soft Computing

Résumé: Dans ce travail, le problème de conception du predicteur neuronal est étudié. Dans une première partie, on présente les differentes series temporelles utilisées dans la partie conception de predicteur. Ensuite , on presente les méthodes stochastiques utilisées pour la prediction des series temporelles stationnaires, non stationaire en moyenne et en variance , saisionnaires ainsi que la méthodologie de Box and Jenkis. Puis un état de l'art des techniques de soft computing : algorithmes génétiques, les réseaux de neurones et la logique floue. Prenant appui sur cet état de l'art nous proposons un predicteur hybride des deux techniques du soft computing RNA/AG. Donc , nous avons présenté dans un premier temps la méthode d'optimisation d'un prédicteur neuronal basée sur un algorithme génétique réel. Cette méthode consiste a l'optimisation simultanée de la topologie de réseaux de neurones, les paramètres de contrôle de réseaux de neurones et les intervalles initiaux des poids synaptiques. D'abord , nous avons proposé une méthode de représentation de l'ensemble des paramètres à optimiser le prédicteur neuronal. Cette représentation fait intervenir un codage en nombre réels. Ensuite , nous avons décrit l'étape d'initialisation des chromosomes en spécifier les sous ensembles de chaque gène dans le chromosome. Enfin ,pour tester l'efficacité de la méthode, des simulations dans 3domaines économie, écologie et météorologie.

Mots-clés : prediction , series des temporelles, predicteur neuronal, Algorithmes génétiques, Algorithmes génétiques, logique flou, le taux de change, pollution d'air .

Time Series Prediction Using Soft Computing Paradigms

Abstract: In this work, the problem of design of neural predictor is studied. In a first part, we present the time series used for the conception of the predicteurs. Than we presente the stochastic methodes used for the prediction of time series and the methodologie of Box and Jinkis. Then a art state of the soft computing techniques: genetic algorithms, the neural networks, fuzzy logic. Taking support on this art state we propose at first the method of optimization of a neuronal predicor based on a real genetic algorithm. This method consists of the simultaneous optimization of the topology of neural networks the control parameters, and the initial intervals of the weights synaptiques. We suggested at first a method of representation of all optimizing neuronal predicteur parameters. This representation used the a coding in real number. Then we described the stage of chromosomes initialization. Finally to test the efficiency of the method, the simulations in 3 domaines of economy , ecology, meteorology.

***Key-Words*:** Prediction, time series, artificial neural network, genetic algorithm, fuzzy logic, foreign exchange rate, air pollution.

تنبأ السلاسل الزمنية باستخدام التقنيات الحاسوبية ناعمة

ملخص : في هذا العمل, نقوم بتقديم مشكلة تصميم المتنبأ العصبي لسلاسل الزمنية. في الجزء الأول نقدم السلاسل الزمنية المختلفة المستعملة في الجزء الخاص بتصميم المتنبأ العصبي. في الجزء الموالي نقوم بتقديم أنماط احتمالية المستخدمة لتنبأ بالسلاسل الزمنية المستقر والغير مستقرة في المتوسط وفى تباين وكذلك تقديم المنهجية بوكس وجنكس. فيما بعد نقوم بشرح مفصل لتقنيات الحوسبة ناعمة :الخوارزميات الجينية, شبكات الخلايا العصبية ,المنطق الغامض. و استنادا لما سبق نقترح في البداية بتقدم طريقة الاستفادة المثلى من إيجاد تحدث تعطلا للمتنبأ العصبي لسلاسل الزمنية حقيقية تقوم على خوارزميات وراثية. هذه الطريقة تتكون من التحسين المتزامن الهيكل من شبكات الخلايا العصبية المعلمات مراقبة شبكات الخلايا العصبية الأولية.

الكلمات المفاتح : التنبؤ, السلاسل الزمنية , الخوارزميات الوراثية, شبكات الخلايا العصبية ,المنطق الغامض شبكات الخلايا العصبية ,المنطق

Introduction générale

I.1. Série temporelles

Prévoir l'évolution d'un marché financier à partir de l'historique des cours de Bourse, faire des prévisions météorologiques à partir des relevés temporels de grandeurs climatiques, générer des alarmes à partir de données temporelles pour la surveillance médicale intensive, analyser des signaux biologiques pour comprendre et contrôler des systèmes vivants complexes faire la prévision de quantité de lait pour savoir les besoins, sont autant de domaines qui nécessitent des techniques et des outils robustes pour le traitement et la prévision de séries temporelles.

Une série temporelle est une suite finie $(x_1,.....,\ x_n)$ de données indexées par le temps. L'indice temps peut être selon les cas : la minute, l'heure, le jour, l'année etc. La discipline qui traite ce type de quantités est *l'analyse des séries temporelles*. L'analyse des séries temporelles en question consiste en opération de *prédiction*, de *modélisation et caractérisation.* L'objectif de prédiction est de prévoir avec précision l'évolution à cours termes de séries temporelles, et la modélisation vise la détermination des caractéristiques du comportement à long terme, la caractérisation à comme but de déterminer les propriétés fondamentales d'une série temporelle. La prédiction est le sujet de ce travail. Nous rappelons que prédiction des séries temporelles a principalement débuté dans les années 70, de nombreuses techniques de prévision de l'évolution des systèmes dynamiques ont été développées pour des domaines aussi variés que l'agriculture, la météorologie, l'automatique, la finance, etc.

Une importante classe du model stochastique utilisé pour la prédiction décrive la relation entre la future valeur de série et les précédentes. Le choix du model dépend de la nature de série (stationnaire, non stationnaire, linaire, non linaire). Pour les séries temporelles stationnaires Box et Jenkins (1970) ont proposé d'utiliser les fonctions d'autocorrélation et d'autocorrélation partielle empiriques. Leur méthodologie repose sur le fait que la fonction d'autocorrélation théorique d'un modèle moyenne mobile $MA(q)$ s'annule à partir du rang $q+1$ et la fonction d'autocorrélation partielle d'un modèle autorégressif $AR(p)$ s'annule à partir du rang $p+1$.

Une alternative pour la sélection de modèle a été introduite par Cleveland (1972). Sa méthode utilise la fonction d'autocorrélation inverse $\rho_i(h)(h \in Z)$, définie à l'aide de l'inverse de la densité spectrale[56]. Cleveland (1972) a introduit deux méthodes différentes, pour estimer cette fonction, basées sur l'estimation de la densité spectrale. La première consiste à estimer tout d'abord la densité spectrale par le périodogramme puis la fonction d'autocorrélation inverse à partir de l'inverse de la densité spectrale estimée. La deuxième méthode utilise le fait que tout processus stationnaire au second ordre peut être approché par un modèle AR(p) où p est choisi suffisamment grand et les cœfficients de ce modèle sont approximés par la méthode des moindres carrés. Du point de vue pratique, cette dernière méthode a été recommandée par plusieurs auteurs : Cleveland (1972), Hipel et al (1977) et Chatfield (1979). Ce dernier a constaté également que la fonction d'autocorrélation inverse possède des propriétés intéressantes qui sont d'une grande utilité pour l'identification et la spécification de modèles de séries temporelles. Il a mis en évidence la dualité existant entre cette fonction et la fonction d'autocorrélation ordinaire, à savoir que la fonction d'autocorrélation inverse d'un processus ARMA(p,q) est identique à la

fonction d'autocorrélation ordinaire d'un processus dual ARMA(q, p). Cette dualité permet en effet de définir et d'estimer la fonction d'autocorrélation partielle inverse à l'aide de l'algorithme de Durbin-Levinson.

De nombreuses recherches ont été consacrées par la suite à l'étude des fonctions d'autocorrélation inverse et d'autocorrélation partielle inverse. Dans le cas des processus linéaires, Bhansali (1980) fut le premier à publier une étude complète concernant la distribution asymptotique de la fonction d'autocorrélation inverse estimée par les deux méthodes évoquées précédemment [54]. Il a établi la convergence et la normalité asymptotique de ces deux estimateurs et les a exploitées pour l'estimation des paramètres d'un modèle moyen mobile. En 1983, il a illustré les résultats obtenus à partir d'expériences de Monte Carlo, puis il a fourni une étude comparative avec des motivations de choix entre les deux méthodes d'estimation. Dans la suite de ces travaux, Bhansali (1983a) a développé une nouvelle approche basée sur les propriétés des espaces de Hilbert pour définir la fonction d'autocorrélation partielle inverse [55]. Par analogie avec la correspondance biunivoque existant entre les paramètres d'un processus autorégressif et les autocorrélations partielles ordinaires, il a montré que les paramètres d'une moyenne mobile sont liés de la même manière aux autocorrélations partielles inverses. Il a établi également la normalité asymptotique de la version empirique de ces dernières, ainsi que leur utilisation pour la construction d'une statistique permettant l'identification de l'ordre d'une moyenne mobile. Bhansali (1987) montre que la fonction d'autocorrélation inverse estimée peut être utilisée comme outil pour la discrimination entre un modèle autorégressif d'ordre m, AR(m), et un modèle moyenne mobile d'ordre h , MA(h), ainsi que pour la sélection de l'ordre du modèle retenu.

Les séries non stationnaire subit un prétraitement avant d'utiliser le modèle ARIMA qui les transforment en séries stationnaire par l'utilisation d'un opérateur selon la nature de la non-stationnarité (non-stationnarité en moyenne, non-stationnarité en variance, saisonnalité), pour les séries non linéaire, les modèles ARCH et GARCH et EAR sont utilisées. Le problème que ces prévisions sont construites sur des séries transformées qui peuvent perdre leur caractère optimal lorsqu'on les exprime de la même façon que les données initiales. Les propriétés des prévisions construites sur la série transformée ne se conservent pas forcément après l'opération de prétraitement inverse. Mais, ces modèles restent très difficiles à appliquer dans la plupart des séries temporelles réelles(cas pratiques).

2. Techniques de soft computing

Les outils intelligents en étaient appliqués au problème de prédiction des séries temporelles et ont démontrées un grand succès. On entend par outils intelligents les techniques de soft computing : les réseaux de neurones, la logique floue et les algorithmes génétiques.

Chacun d'entre eux a des propriétés particulaires qui lui rendent souhaitable pour résoudre une large famille de problèmes. Néanmoins, ils ont aussi quelques limitations et inconvénients qui ne permettent pas leur application individuelle dans quelques cas.

Les réseaux de neurones sont apparus dans les années cinquante mais n'ont reçu cependant un intérêt considérable qu'à partir des années 80 avec l'apparition de l'algorithme de rétropropagation (Rumelhart et McClelland, 1986) [53]. Leur capacité d'apprentissage et d'approximation de fonctions leur procure un intérêt considérable de la part des chercheurs. Il suffit de voir les nombreuses applications industrielles qui en découlent à partir des années 90 et de consulter l'abondante littérature sur le sujet pour s'en convaincre.

La logique floue introduite par Zadeh (1965) dans les années soixante constitue un outil très puisant pour la représentation des termes et des connaissances vagues. Elle est issue de la capacité de l'homme à décider et à agir d'une manière intelligente malgré l'imprécis et

l'incertitude des connaissances disponibles. Son utilisation dans le domaine du contrôle (contrôle flou) a été l'une des premières applications de cette théorie dans l'industrie avec les travaux de Mamdani et Assilian (1975). Depuis, les applications de la logique floue se sont multipliées pour toucher des domaines très divers.

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 (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.

Au cours de ces dernières années, la combinaison de ces techniques a attiré l'attention de beaucoup de chercheurs. Plusieurs hybridations ont été alors proposées dont les plus rencontrées sont: Algorithme Génétique /Contrôleur flou (AG/CF) et Réseau de neurones/Contrôleur flou (RN/CF).

La première combinaison (AG/CF) vise à la conception des CF par algorithmes génétiques. Karr et Gentry (1993) utilisent un algorithme génétique standard (codage binaire, opérateurs de croisement et de mutation simples) pour l'optimisation des paramètres des fonctions d'appartenance. Le nombre de termes linguistiques associé à chaque variable du contrôleur est différent, mais il reste fixe durant tout le processus d'optimisation. Plu sieurs méthodes ont été proposées par la suite, dont certaines diffèrent par le type de codages et/ou des fonctions d'appartenance utilisées (Liska et Melsheimer, 1994; Herrera et al., 1995) [57]. Récemment, les algorithmes hiérarchisés ou AGH sont aussi impliqués. Dans (Acosta et Todorovich, 2003)[58], on utilise un AGH avec un seul niveau de gènes de contrôle pour chercher le nombre minimal de fonctions d'appartenance à associer à chacune des variables du contrôleur flou. Thrift (1991) est le premier à décrire une méthode d'optimisation des règles floues par algorithme génétique standard, en utilisant trois bits pour coder chaque règle[58]. A la même époque, Karr propose une méthode permettant de faire intervenir dans le contrôleur flou à la fois des règles spécifiées par un expert humain et des règles optimisées par un algorithme génétique (Karr, 1991) [60]. Quelques améliorations ont été apportées par la suite (Herrera et al., 1998[61]; Chen et Wong, 2002[62]; Belarbi et al., 2005[63]). Homaifar et MCCormick (1993) proposent une méthode d'optimisation qui prend en compte simultanément les fonctions d'appartenance des variables d'entrées et les règles floues[64]. Les différents paramètres sont codés sur le même chromosome en base 6. Dans (Lee et Takagi, 1990), les auteurs partagent la même idée de Hamaifar et MCCormik (codage de l'ensemble des paramètres dans un chromosome unique) en utilisant un codage binaire[65].

La combinaison (RN/CF), ou tout simplement contrôleur neuro-flou est une combinaison de la logique floue et des réseaux de neurones qui tire profit des deux approches. Plusieurs architectures neuro-floues ont été proposées dans la littérature suivant le type de règles floues qu'elles intègrent (Mamdani ou Sugeno). La puissance de ces structures réside dans la possibilité d'incorporer une base de connaissances, de traiter les données imprécises et vagues par logique floue et en même temps d'introduire l'apprentissage via le réseau de neurones. Pour la plupart des architectures proposées, les procédures d'apprentissage appliquées sont soit supervisées et se basent sur les techniques d'optimisation classique (descente du gradient, les moindres carrés), soit non supervisées et on utilise les algorithmes de classification (K-means).

I.3. Objectifs de ce travail

Les réseaux de neurones ont des propriétés particulaires qui lui rendent souhaitable pour résoudre une large famille de problèmes. Néanmoins, ils ont aussi quelques limitations et inconvénients qui ne permettent pas leur application individuelle dans quelques cas. La combinaison (RN/AG) permettent de fusionner toutes les avantages des deux approches individuelles et ont de plus la capacité de surmonter des difficultés et les limitations qui caractérisent chaque approche.

Divers formes pour combiner le réseau neural et des algorithmes génétiques. Dans la plupart des cas, des algorithmes génétiques sont utilisés pour : l'apprentissage de réseau, trouver une structure ou topologie convenable. L'étude de la combinaison de réseau neurones et des algorithmes génétiques peut résumer à trois aspects généraux : apprendre génétiquement le réseau neurones, optimisation génétique de topologie de réseau et optimisation des paramètres de contrôle. Dans ce travail on s'intéresse aux trois combinaisons ensemble c à d on va faire l'apprentissage ainsi que le choix de topologie et la sélection des paramètres de contrôle par les algorithmes génétique réel.

I.3.1 Algorithmes Génétiques pour l'apprentissage des réseaux de neurones

L'algorithme de rétro propagation est un algorithme de descente de gradient, qui minimise d'erreur une fonction d'erreur, l'erreur quadratique moyenne totale entre la sortie de réseau et la sortie réelle. Cette erreur est utilisée pour ajuster les poids synaptique de réseau. L'algorithme de rétro propagation a quelques inconvénients en raison de son incapacité de s'échapper du minimum local et trouver le minimum global.

Considérant la procédure d'apprentissage comme une procédure d'optimisation qui peuvent entre appliquer comme un algorithme au réseau de neurones, où l'objectif de recherche évolutionnaire est de chercher les intervalles de variation initiale des poids synaptique de différentes couches (couches cachée et couche de sortie) qui minimise l'erreur d'apprentissage. Les GA, travaillant avec une population de points de solution différents, ont la capacité de surmonter le problème des minimums locaux en cherchant simultanément beaucoup de régions dans l'espace de recherche.

Dans la formulation générale de cette approche, le mécanisme génétique est appliqué à une population d'individus qui représentent l'intervalle de variation des poids synaptique du réseau neurones.

I.3.2. Algorithmes Génétiques pour l'optimisation de la topologie des réseaux de neurones

Dans la section précédente nous avons vu comment les algorithmes génétiques peuvent être utilisés pour l'apprentissage du réseau de neurones pour une topologie donnée. Le problème inverse est aussi très intéressant, Cela doit exécuter une recherche génétique pour trouver la meilleure topologie pour une auto apprentissage de réseau neurones. Le choix de topologie de réseau est très important à cause de son grand degré d'influence sur la capacité de performance de réseau. C'est quoi la meilleure structure neurale pour un problème donnée c'est la question dans la plupart de cas et il n'y a aucune loi générale pour la trouver. Elle dépend des caractéristiques et la taille de données. Si la topologie est très petite le réseau ne peut pas apprendre les exemples d'apprentissage, tandis qu'une grande topologie limite la capacité de généralisation du réseau. Une méthode sûre pour obtenir la topologie convenable est basée les algorithmes génétiques qui peuvent fournir une approche séduisante pour résoudre ce problème.

Il y'a deux type de codage pour la conception de l'architecture d'un réseau neural : le directeur ''le codage direct ou bas niveau'' et ''le codage indirect ou haut niveau''.

Le codage direct : toutes les informations disponibles sur l'architecture sont directement représentées par un codage binaire. Dans la plupart des cas le chromosome représente les connexions entre toutes les couches. Une matrice $C=(c_{ij})_{N\times N}$ décrive la présence des connexions. Si $c_{ij}=1$ alors il y a une connexion entre le neurone i et le neurone j, tandis que la valeur $c_{ij}=0$ dénote l'absence de la connexion correspondant. Le chromosome dans cette sorte de codage indique ligne par ligne la forme de la structure de connexion du réseau.

Le codage indirect : détermine seulement les caractéristiques de la topologiques les plus importantes, nombre de couche cachée, le nombre des neurones dans chaque couche, la présence et l'absence biais et dans pour notre application le nombre d'entrées de réseau, le temps entre deux entrée successif.

Cette approche est plus intéressante et avantageuse puisqu'il exige plus de connaissance et, en conséquence, moins d'espace de recherche pour l'architecture neurale.

I.3.3. Algorithmes Génétiques pour l'optimisation des paramètres de contrôle

Les algorithmes génétiques sont utilisés dans ce cas pour optimiser le coefficient d'apprentissage, le coefficient d'amortissement d'algorithmes de rétro propagation qui doivent être normalement choisit par l'utilisateur. Cela peut être fait en ajoutant ces paramètres dans la structure de chromosome.

En combinant ces trois approches, des algorithmes génétiques peuvent être aussi utilisés pour déterminer l'intervalle de variation des poids synaptique, la topologie, les paramètres de contrôle simultanément.

L'objectif de cette thèse est de concevoir un prédicteur hybrides à base des deux techniques du soft computing réseaux de neurones et les algorithmes génétiques.

Ce prédicteur peut extraire automatiquement l'intervalle de variation des poids synaptique, la structure de réseaux de neurones, les paramètres de contrôle de réseaux à partir des valeurs précédentes de série temporelle cela on utilisant les algorithmes génétiques réel.

Les algorithmes génétiques utilisée sont à codage réel dont les gènes de chromosome sont les caractéristiques de la topologie de réseau de neurone (nombre de couches cachées, nombre de neurones dans chaque couches, la forme de la fonction d'activations de chaque couche, nombre d'entrée de réseaux, retards, Présence ou l'absence de biais b, coefficient d'apprentissage η, coefficient d'inertie α, l'intervalle de variation des poids synaptique de chaque couche). Ce type de structure de chromosome favorise :

- L'élimination des minimums locaux
- L'apprentissage des poids synaptiques
- La minimisation de la complexité de la structure du réseau de neurone.

La fonction objective utiliser pour évaluer le prédicteur neuronal est l'erreur quadratique moyenne MSE.

Enfin, un sujet de recherche comme la prédiction de séries temporelles ne peut pas se borner à une étude purement théorique mais demande un volet expérimental et applicatif. Un dernier

objectif sera donc de tester cette méthode sur des séries temporelles réelles, issues de 3 domaines
1. Météorologie : la température journalière, température hebdomadaire.

2. Ecologie : pollution d'air (Concentration de dioxyde carbone, Concentration d'ozone).

3. Finance : les taux de change journalier d'US Dollar (USD) contre Euro Européen, Dollar Canadien (CAD), Pound Britannique (GBP), Yen Japonais (JPY).

4. Présentation des chapitres

Cette thèse est organisée en quatre chapitres comme suit:

- 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 l'histoire de l'analyse des séries temporelles.
- Le troisième chapitre est divisé en trois parties. La première est consacrée aux systèmes flous. Nous rappelons d'abord les notions de bases sur lesquelles reposent ses systèmes puis nous décrivons leur principe de fonctionnement et leurs différentes composantes. La deuxième partie traite les réseaux de neurones et leur définition, la modélisation des réseaux de neurones, les différents types d'apprentissage et les règles utilisées. La troisième partie est une description détaillée des algorithmes génétiques dans laquelle nous rappelons les définitions relatives à leur fonctionnement.
- Le quatrième chapitre est consacré à présenter le prédicteur neuronal optimisée par les algorithmes génétiques, et les simulations effectuées.

Donnée

II.1. Introduction

Les progrès de la science ont permis de comprendre plusieurs aspects de notre environnement et, par conséquent, la prédictibilité d'un grand nombre d'événements a augmenté [8]. De plus, la technologie informatique permet d'enregistrer et de traiter une énorme quantité d'observations à moindre coût, ce qui n'a pas toujours été le cas. Ces deux éléments ont donc renforcée notre capacité à prévoir les résultats d'un certain nombre de phénomènes.

Pourquoi faire des prévisions? *‘’understanding the past to forecast the future"* [6]. La prévision est fondamentale dans la mesure où elle est à la base de l'action." dit Guy Melard [4]. Le désir de prévoir le futur serait donc motivé par la volonté de l'améliorer ou de s'y préparer. En effet, une entreprise commerciale tente de prévoir les ventes futures d'un produit pour faire face à la demande. Les producteurs d'électricité s'intéressent à la consommation des jours à venir pour adapter leur production. Une entreprise pharmaceutique tente de prévoir les effets secondaires d'un médicament avant de le commercialiser. Nos choix sont donc généralement dirigés par l'anticipation des conséquences de différentes actions. En d'autres termes, la prise de décision est généralement basée sur des prévisions.

Comment faire des prévisions? Pour pouvoir effectuer des prévisions, on a besoin d'informations concernant le passé et on doit pouvoir supposer que certains comportements du passé continueront à se reproduire dans le futur : c'est ce qu'on appelle l'hypothèse de continuité [8]. Notre rôle sera donc d'analyser et d'utiliser cette information passée pour tenter d'en extraire des particularités, qui seront ensuite utilisées pour effectuer des prévisions.

II.2. Définition (série temporelle)

Une série temporelle est une succession d'observations $(x_1, x_2, ..., x_n)$ indexées par le temps. L'indice temps peut être selon les cas : la minute, l'heure, le jour, l'année etc.... Le nombre n est appelé la longueur de la série [1].

une série temporelle est donc toute suite d’observations correspondant a la même variable : il peut s’agir de données macroéconomiques (le PIB d’un pays, l’inflation, les exportations…), microéconomiques (les ventes d’une entreprise donnée, son nombre d’employés, le revenu d’un individu, le nombre d’enfants d’une femme…), financières (le prix d’une option d’achat ou de vente, le cours d’une action), météorologiques (la pluviosité, le nombre de jours de soleil par an…), politiques (le nombre de votants, de voix reçues par un candidat…), démographiques (la taille moyenne des habitants, leur age…). En pratique, tout ce qui est chiffrable et varie en fonction du temps. La dimension temporelle est ici importante car il s’agit de l’analyse d’une chronique

historique : des variations d'une même variable au cours du temps, afin de pouvoir comprendre la dynamique. La périodicité de la série n'importe en revanche pas : il peut s'agir de mesures quotidiennes, mensuelles, trimestrielles, annuelles… voire même sans périodicité.

On représente en général les séries temporelles sur des graphiques de valeurs (ordonnées) en fonction du temps (abscisses). Lorsqu'une série est stable autour de sa moyenne, on parle de **série stationnaire**. Inversement, on trouve aussi des **séries non stationnaires**. Lorsqu'une série croit sur l'ensemble de l'échantillon et donc possède une moyenne qui n'est pas constante, on parle de **tendance**. Enfin lorsqu'on observe des phénomènes qui se reproduisent à des périodes régulières, on parle de **phénomène saisonnier**.

II.3. Intérêt des séries temporelles

Il est d'usage de considérer l'intérêt des séries temporelles selon trois perspectives : descriptive, explicative et prévisionnelle.

II.3.1.Description

- L'analyse temporelle permet de connaître la structure de la série de données étudiée
- Elle peut être utilisée pour comparer une série à d'autres séries (varicelle et oreillons, par exemple) ;

II.3.2.Explication

- Les variations d'une série peuvent être expliquées par une autre série (exposition météorologique, pollution atmosphérique, etc.) ;
- il est possible de modéliser une intervention externe grâce à l'analyse de séries temporelles;
- Ces analyses permettent de réaliser des scénarios pour la période contemporaine : en agissant sur une variable explicative, il est possible d'observer le comportement de la variable expliquée ;

II.3.3. Prévision

- La prévision a priori permet la planification ;
- La prévision a posteriori permet d'estimer l'impact d'une perturbation (dépistage, par exemple) sur la variable expliquée ;
- Des scénarios pour le futur, enfin, peuvent être réalisés.

II.4. Domaines d'application

On trouve des exemples de séries chronologiques dans de très nombreux domaines.

La liste suivante n'est qu'un ´échantillon :

- Finance et économétrie : évolution des indices boursiers, des prix, des données économiques des entreprises, des ventes et achats de biens, des productions agricoles ou industrielles,

- Assurance : analyse des sinistres,
- Médecine/biologie : suivi des évolutions des pathologies, analyse d''électroencephalogrammes et d''electrocardiogrammes,
- Sciences de la terre et de l'espace : indices de marées, variations des phénomènes physiques (météorologie), évolution des taches solaires, phénomènes d'avalanches.
- Traitement du signal : signaux de communications, de radars, de sonars, analyse de la parole,
- Traitement des données : mesures successives de position ou de direction d'un objet mobile (trajectographie),
- Démographie : analyse de l''evolution d'une population
- Météorologie : analyse de données climatiques. . .
- Géophysique : analyse de données sismiques. . .
- Traitement d'images : analyse d'images satellites, médicales.
- Energie : prévision de la consommation d'électricité.
- Agriculture : production mensuelle de lait par vache, la quantité d'un produit

II.5. Sources des séries temporelles utilisées

II.5.1. Les séries temporelles écologiques (pollution d'air)

La pollution d'air est un indicateur de la qualité d'air qui repose sur la concentration de 4 polluants (dioxyde d'azote, ozone, dioxyde de soufre, dioxyde de carbone).Les données de pollution atmosphérique sont prises de plusieurs cas d'études de « *time series modelling of water resources and environmental systems* » par K.W. Hipel et A.I. Mcleod (1994), publié par Elsevier, Amsterdam. 1994. ISBN 0-444-89270-2. Le but majeur de mesure de la pollution de est la réduction de la pollution l'air est et soutenir et protéger la santé publique, le bien-être et des ressources écologiques.

II.5.1.1 Ozone (O_3) : Les données d'ozone consistent en maximum mensuel données de concentration d'ozone de janvier 1965 jusqu'à décembre 1997 à Azora, Suisse. Comme peut voir sur la figure II.1, le maximum mensuel de niveau de concentration d'ozone est typiquement élevé dans les mois d'été et diminue dans les mois d'hiver. La courbe de forme de sinus montre la saisonnalité des fluctuations d'ozone concentration d'une année à l'autre.

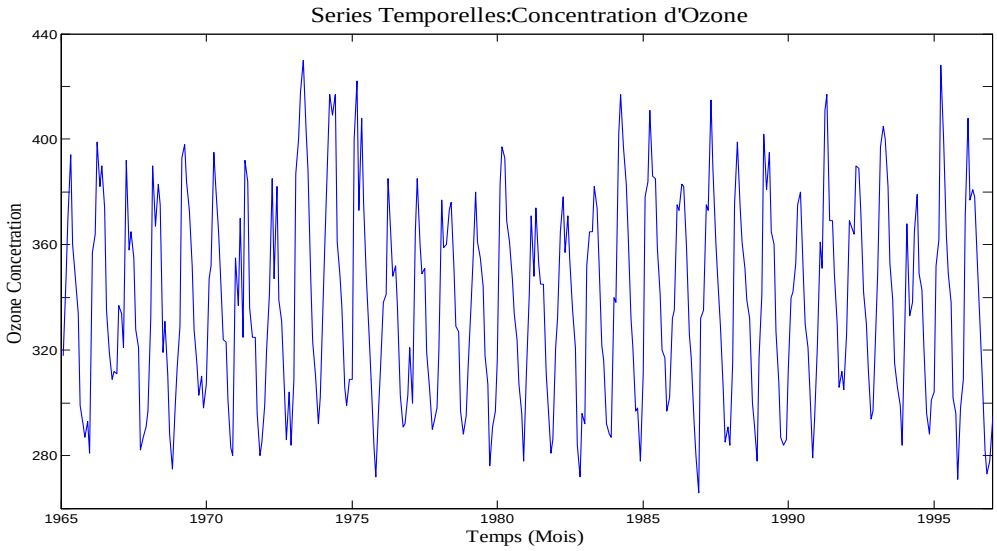


Figure II.1. La concentration d'ozone à Arosa, Suisse, de 1965-1997

II.5.1.2 Dioxyde de carbone (CO_2) : Les données de dioxyde de carbone consistent en maximum mensuel dioxyde de carbone de 1 heure des données de concentration (CO_2) de janvier 1965 jusqu'à décembre 1997 à Arosa.

De la figure II.2 nous pouvons voir que les maximum mensuelles concentrations de dioxyde de carbone de 1 heure, contrairement avec la concentration de l'ozone, à tendance haute en hiver et basse en étés. Les concentrations de dioxyde de carbone ont continué à augmenter au cours de la période. À la fin de 1997, la moyenne annuelle a augmenté par plus de 20 % depuis 1965.

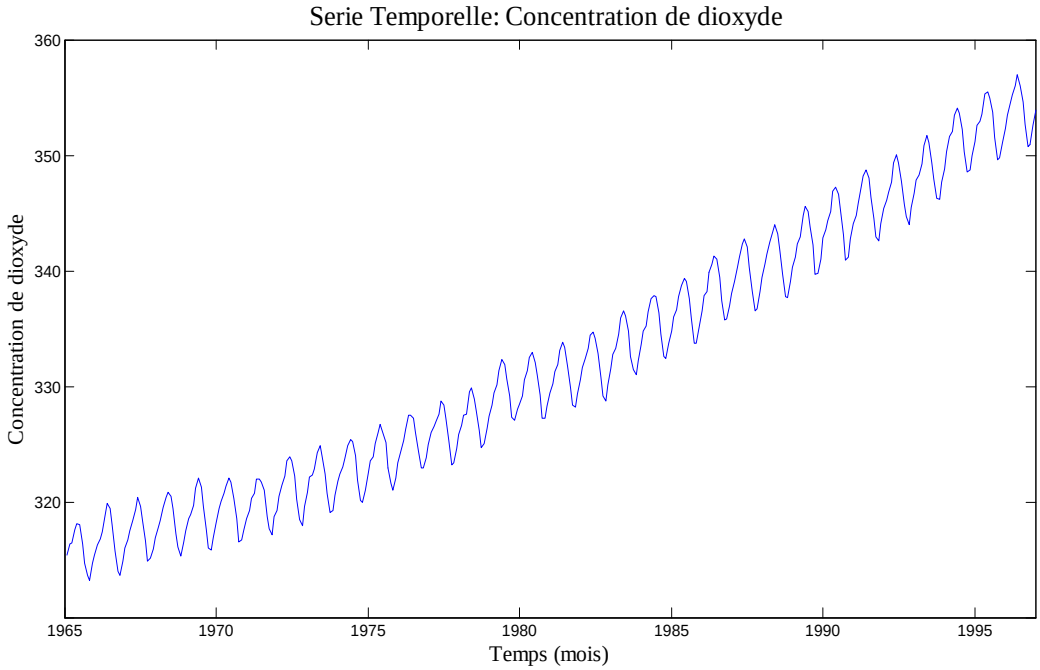


Figure II.2. Concentration de dioxyde à Arosa, Suisse, de 1965-1997

II.5.2. Les séries temporelles météorologiques

Les données météorologiques comprennent la précipitation mensuelle, la température de l'air minimale journalière et la température de l'air hebdomadaire

II.5.2.1. Température : Les températures journalières et hebdomadaires sont disponibles de 1980 jusqu'au 1990 au Melbourne Australie. Ces données ont été rassemblées du site « time series library ».

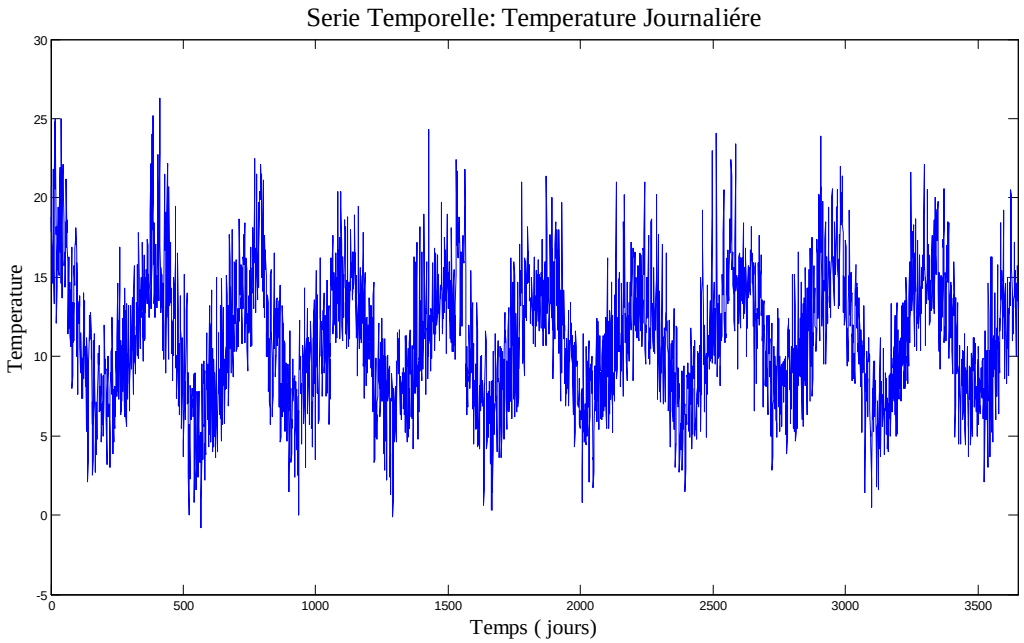


Figure II.3. La température journalière à Melbourne Australie de 1980-1990

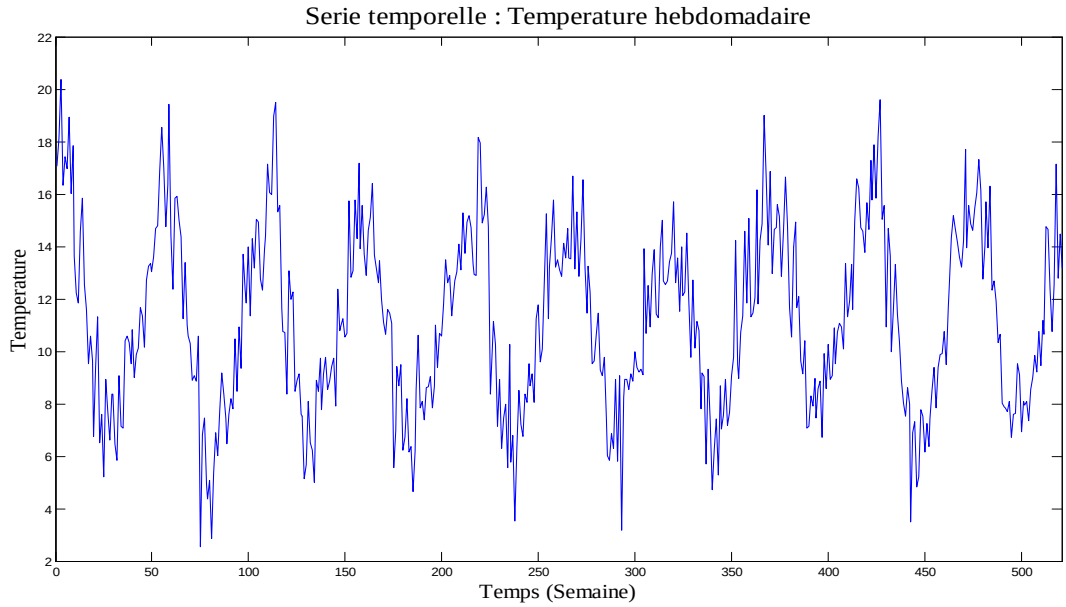


Figure II.4. La température hebdomadaire à Melbourne Australie de 1980-1990

II.5.3. Les séries temporelles économiques (financières)

Les données utilisées dans cette étude sont les taux de change de Dollar des États-Unis (USD) contre : Euro Européen, Livre britannique (GBP), Dollar Canadien (CAD), Yen Japonais (JPY) de 02/01/2003 à 31/12/2010 trouvées sur le site Web http: **http://pacific.commerce.ubc.ca/xr**.

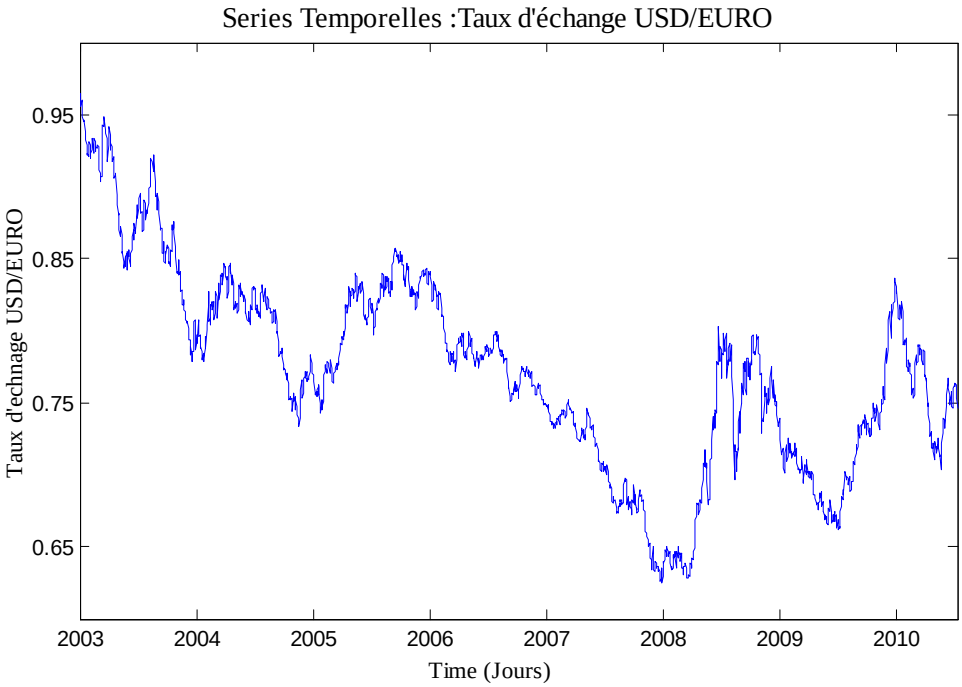


Figure II.6. Les taux de change USD contre EURO

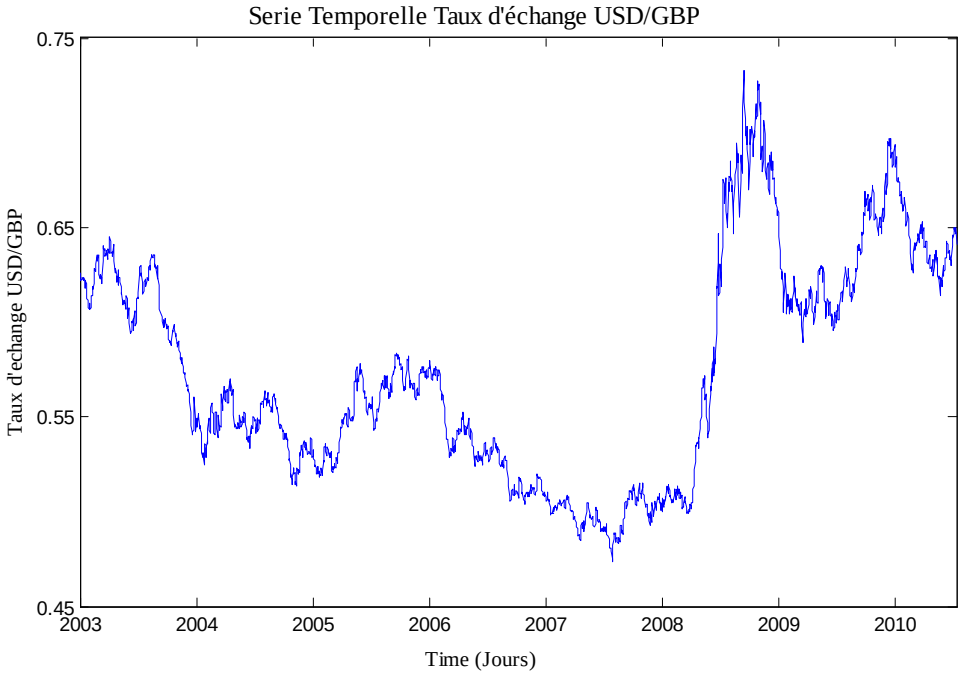


Figure II.7. Les taux de change USD contre GBP

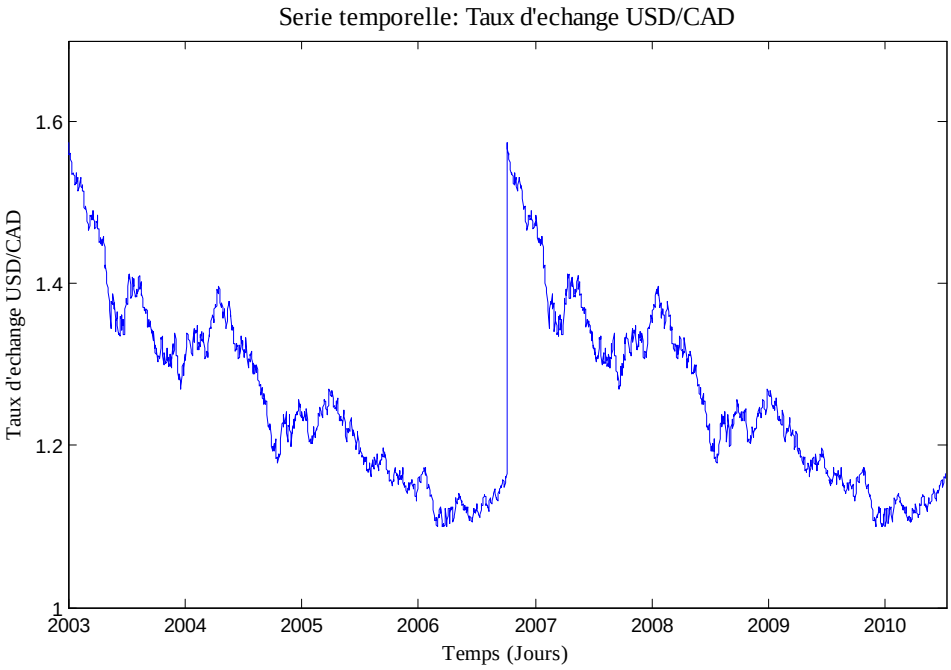


Figure II.8. Les taux de change USD contre CAD

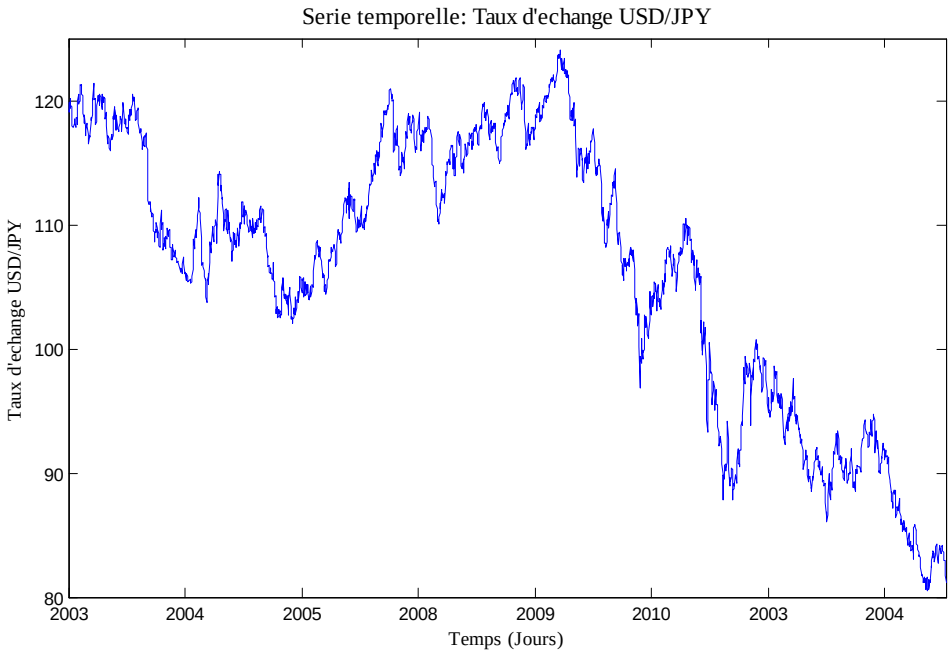


Figure II.9.Les taux de change USD contre JPY

Séries temporelles

III.1. Introduction

La prédiction de l'évolution future des phénomènes a reçu depuis très longtemps un intérêt spécial ce qui a accéléré le développement des techniques de prédiction et aussi a donné une variété des valeurs scientifiques entre ces techniques. Les domaines d'applications de prédiction sont très large tel que: économie, atmosphère, socio-politique, astrologie.

La prédiction traditionnelle exige la connaissance des informations précédentes sur le comportement du système à prédire. Si le calcule du comportement futur est exacte, une telle méthode de prédiction est entièrement déterministe. Mais en réalité, plusieurs facteurs rend le calcule exacte du comportement futur impossible. Cependant il est possible de générer un modèle qui peut être utilisé pour calculer la probabilité de futur comportement entre deux limites spécifiées. Tel modèle est appelé *modèle stochastique* ou *processus stochastique*. Une importante classe de modèles stochastiques est utilisée pour la description des séries temporelles stationnaires appelée la classe des modèles stochastiques stationnaires. Ces modèles supposent que les propriétés de série temporelle sont invariantes par la translation temporelle; parmi ces modèles il y a: AR, MA, ARMA. Les processus utilisés pour la description des séries temporelles non stationnaires (en moyenne, en variance, cessionnaire) sont: ARIMA, SARIMA.

Et pour la construction des modèles quelles que soient leurs classes, Box et Jenkins ont introduit une méthodologie utilisée pour l'obtention d'un modèle linéaire qui s'ajuste au mieux à une série temporelle. Cette méthodologie se décompose en trois étapes: l'identification du modèle, l'estimation des paramètres et la validation du modèle [1].

III.2. Historique de l'analyse de séries temporelles

On peut devisées l'historique de l'analyse des séries temporelles en trois périodes

III.2.1. Première période : visualisation graphique des séries temporelles

C'est en astronomie qu'apparaissent les premières séries temporelles constituées volontairement à des fins d'analyse.

X^{e} siècle

Premier graphique connu de séries temporelles représente l'inclinaison des orbites de sept planètes en fonction du temps (figure III.1)[14]. Il semble toutefois que ce graphique soit un événement isolé. Les graphiques de séries temporelles ne sont réapparus dans les écrits scientifiques que durant le $XVIII^{e}$ siècle [15].

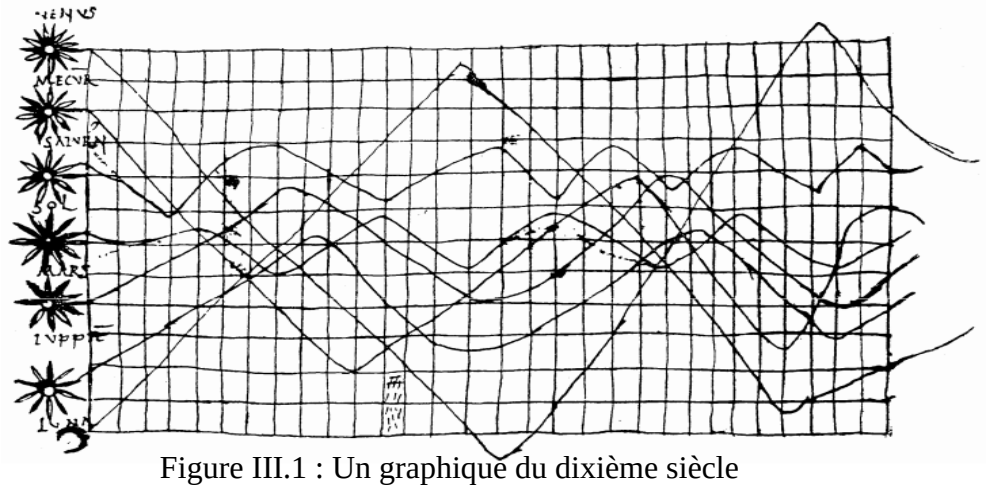


Figure III.1 : Un graphique du dixième siècle

XVI^{e} - XVII^{e} siècles

Accumulation et analyse informelle de séries astronomiques. L'astronome danois Tycho Brahe (1546-1601) enregistre des données nombreuses et précises sur les orbites des planètes (1572). Ces données sont utilisées par son assistant Johannes Kepler (mathématicien et astronome allemand, 1571-1630) pour formuler les lois du mouvement des planètes qui portent son nom (1609).

1760-80 Johann Heinrich Lambert

Le mathématicien et philosophe suisse publie des graphiques d'une grande qualité pour toutes sortes de séries temporelles, par exemple, son graphe (publié en 1779) de la variation de la température du sol en relation avec la profondeur sous la surface (figure II.2.).

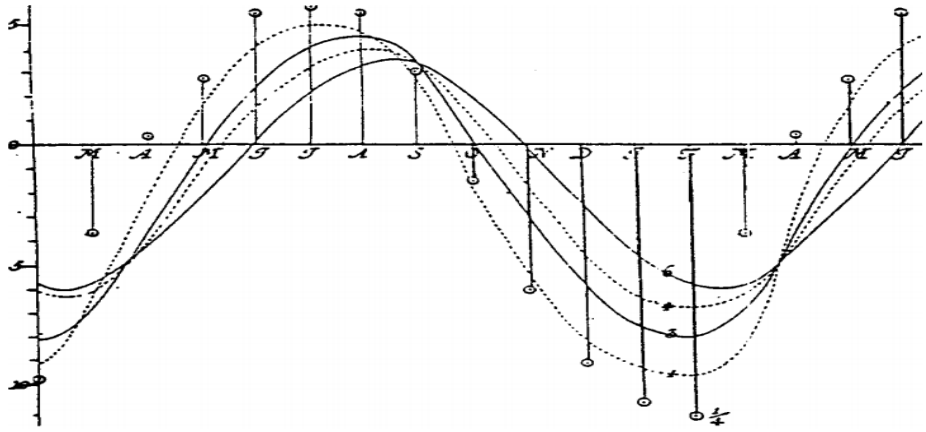


Figure III.2. : Variation de la température du sol en relation avec la profondeur sous la surface (Lambert, 1779, Berlin)

II.2.2. Deuxième période : les techniques déterministes

A partir des 18eme et 19eme siècles, on passe de la visualisation graphique aux premières techniques déterministes. Citons les voies importantes :

1772 Méthode mathématique pour détecter les périodicités (Lagrange, 1772).

Louis de Lagrange (astronome et mathématicien français, 1736-1813) propose une méthode mathématique pour mesurer les périodicités dans les données astronomiques. Il applique sa méthode à l'étude de l'orbite d'une comète. Sa méthode est basée sur l'utilisation des fonctions rationnelles. Comme d'autres méthodes analogues (Dale, 1914 ; Prony), elle est complexe à utiliser, sauf pour des séries courtes, de même que très sensible aux observations à l'écart [9].

1807 Analyse harmonique (Fourier).

Joseph Fourier (mathématicien français, 1768-1830) propose, en 1807, que toute fonction périodique peut s'écrire comme une somme (ou série) de fonctions sinusoïdales et cosinusoïdales. En d'autres termes, si $f(t)$ est une fonction périodique de période P, elle peut s'écrire sous la forme :

$$f(t) = \sum_{n=0}^{\infty} \left[A_n \cos(2\pi nt / P) + B_n \sin(2\pi nt / P) \right] \quad \text{(III.1)}$$

1841 Etude de fluctuation saisonnière (W. Gilbart)

Le banquier anglais James W. Gilbart étudie les fluctuations saisonnières dans la circulation des billets de banque. Gilbart est peut-être l'un des premiers à examiner de façon précise des fluctuations saisonnières.

1856 Mesure de composant irrégulier (Charles Babbage)

L'ingénieur Charles Babbage (1856) étudie des données quotidiennes sur les opérations de << clearing >> (durant l'année 1839) et cherche à identifier des cycles entre semaines et entre mois. Il cherche aussi à mesurer la composante irrégulière de la série.

1860 Analyses empiriques de données économiques (Clément Juglar)

Le médecin français Clément Juglar, dans son ouvrage intitulé Des crises commerciales et de leur retour périodique (publié en 1860), interprète les crises commerciales comme des phénomènes récurrents plutôt que des crises accidentelles. Analysant des séries temporelles de prix, de taux d'intérêt et de masses monétaires, Juglar conclut à une périodicité d'environ dix ans. Son analyse est l'une des premières études empiriques de données économiques en vue d'étudier les cycles d'affaires [10]

1884 Moyennes mobiles. Corrélation sérielle (Poynting).

En 1884, le physicien Poynting propose l'utilisation de moyennes mobiles arithmétiques pour éliminer les variations accidentelles ou périodiques. Cette idée sera ensuite fortement utilisée dans le cadre des méthodes de désaisonnalisation. Dans le même article, Poynting fait référence à la notion de corrélation sérielle en étudiant les relations entre, d'une part, les mouvements du prix du blé en Angleterre, en France et au Bengale et, d'autre part, l'importation de coton et de soie en Grande-Bretagne. Toutefois, Poynting n'introduit pas explicitement le coefficient de corrélation sérielle.

1894-1906 Périodeogramme (Schuster).

Dans une série d'articles, Schuster (1894, 1897, 1898, 1900, 1904, 1906) développe considérablement l'analyse harmonique empirique. En 1897, il montre que le cycle lunaire détecté par Knott (1897) ne peut être considéré comme statistiquement significatif. En 1898, il propose le périodogramme comme technique pour détecter les périodicités importantes dans une série temporelle. Il applique sa technique à des données météorologiques (1898) ainsi qu'à des données sur les taches solaires (1900, 1906). Pour des données $x_t, t = 1,2,.....,n$, le périodogramme de Schuster est défini par la fonction [11] :

$$I_n(P) = \frac{1}{n}\left\{\left[\sum_{t=1}^{n} x_t \cos(2\pi t / P)\right]^2 + \left[\sum_{t=1}^{n} x_t \sin(2\pi t / P)\right]^2\right\} \quad \text{(III.2)}$$

Si la série contient un terme périodique de période P_0 , le périodogramme a un pic pour $P = P_0$. Souvent aussi, on considère la fonction équivalente

$$\bar{I}_n(v) = I_n(1/v) = \frac{1}{n}\left\{\left[\sum_{t=1}^{n} x_t \cos(2\pi v t)\right]^2 + \left[\sum_{t=1}^{n} x_t \sin(2\pi v t)\right]^2\right\} \quad \text{(III.3)}$$

On appelle $\bar{I}_n(v)$ un *spectrogramme*. Schuster propose aussi un test pour décider si un pic dans le périodogramme est statistiquement significatif. D'autres tests de cette nature ont par la suite été proposés par Walker (1914, 1925, 1927, 1931), Fisher (1929) et Bartels (1935).[9]

1914 Utilisation de la différenciation pour éliminer la tendance d'une série temporelle (Student, Moore).

En 1914, William Sealy Gosset (Student) suggère qu'on peut éliminer la tendance d'une série temporelle en différenciant celle-ci une ou plusieurs fois. Cette méthode a ensuite été développée par Beatrice M. Cave et Karl Pearson(1914), Oscar Anderson (1914, 1923, 1927, 1929), Warren M. Persons (1916, 1917, 1922), G. U. Yule (1921) et Gerhard Tintner (1935, 1940). Dans le même esprit et afin d'estimer des courbes de demande, Henry Moore (1914) utilise les taux changements de quantités et de prix, plutôt que les données en niveaux, afin d'éliminer la tendance [9,10].

II.2.3. Troisième période : techniques aléatoires

A partir du 20eme siècle, l'aléatoire est pris en compte, notamment à l'aide des travaux de Yule (1927). Ces travaux sont issus de l'observation du mouvement oscillatoire d'un pendule bombardé de petits pois lancés par un enfant. Il y a ensuite de nombreux contributeurs aux méthodes aléatoires.

1926-27 Processus de moyenne mobile MA (Yule, 1926 ; Slutsky, 1927).

Yule (1926) démontre que le fait de différencier un bruit aléatoire introduit des autocorrélations artificielles. Dans la même veine, indépendamment de Yule, Slutsky (1927, 1933, 1938) montre qu'en calculant une moyenne mobile à partir d'un bruit blanc, on obtient une série dont les observations ne sont pas indépendantes [9].

1927 Modèle autorégressif AR (Yule).

En 1927, Yule propose le modèle autorégressif et trouve qu'un modèle autorégressif d'ordre 2 représente assez bien le comportement des données de Wolfer sur les taches solaires. Yule montre que le modèle autorégressif peut conduire à l'apparition de fluctuations cycliques. Toutefois, les caractéristiques d'amplitude et de phase de ces cycles sont variables.

1930 Analyse harmonique généralisée (Wiener).

Les résultats de Wiener incluent des définitions précises de la fonction d'autocovariance et du spectre d'une série Temporelle, le développement de représentations spectrales pour la fonction d'autocovariance et la série elle-même, ainsi qu'une étude mathématique des opérations de filtrage. Ces résultats se prêtent bien à une utilisation empirique. Ainsi, la méthode de Wiener est rapidement appliquée par Kenrick (1929).[9,12]

1933 Distinction entre chocs et mécanismes de propagation (Frisch).

Sur la base des analyses théoriques de Wicksell, Frisch (1933) souligne l'importance de faire la distinction entre chocs et mécanismes de propagation dans la construction de modèles économiques dynamiques.

1934 Critique de la décomposition des séries temporelles (Roos).

C. F. Roos (1934) critique la pratique qui consiste à enlever la tendance d'une série temporelle avant de l'étudier. Si on n'explique pas la tendance, cela signifie que des facteurs importants sont laissés de côté. La pratique de la décomposition est aussi critiquée par Haberler (1963).

1941 Livre de H. T. Davis, The Analysis of Economic Time Series.

Premier ouvrage d'importance sur l'analyse des séries économiques. Davis y présente notamment plusieurs périodogrammes originaux. Le livre de Davis est le dernier ouvrage qui tente d'établir le périodogramme comme un instrument majeur en économie. Après les années 30, les économistes se sont désintéressés de ce genre d'analyse en économie. Les raisons principales en sont :

1. L'absence apparente de périodicités exactes en économie (Mitchell) ;
2. Le fait que le nombre de cycles suggérés par les périodogrammes est habituellement très grand ;
3. Les coûts de calcul importants ;

1946 -70 Développement de l'analyse spectrale.

Après le développement de la théorie des processus stationnaires et l'introduction de la notion de spectre d'une série stationnaire durant les années 1925-45, l'analyse spectrale empirique se développe rapidement et les applications se multiplient, notamment en physique, en génie (électrique, particulièrement), en acoustique, en géophysique, en médecine, en économie, en biologie, en psychologie et en analyse numérique.

1970 Le processus AutoRégressif, Moyenne mobile ARMA

Développés par Box & Jenkins en 1970. Un processus ARMA est en fait constitué d'une partie autorégressive notée AR (AutoRegressive Process) qui est une combinaison linéaire finie en t des valeurs passées du processus, et d'une partie moyenne mobile notée MA (Moving Average Process) qui est une combinaison linéaire finie en t des valeurs passées d'un bruit blanc,

III.3. Modélisation stochastique des séries temporelles

Une série temporelle est une séquence de quantités mesurées à la sortie d'un système à des intervalles de temps régulier. Beaucoup de phénomènes physiques et économiques peuvent être décrit au moyen de processus stochastique. De façon formelle le processus stochastique est une famille de variables aléatoires $\{Y_{t\ XR},\ t \in T\}$ définies sur le même espace de probabilité (Ω, a, P)[2].

Le processus est dit à temps discret si T est fini ou dénombrable, et on dira qu'il est à temps continu dans le cas contraire. Le bruit blanc est un exemple important de processus stochastique à temps discret souvent utilisé en prévision de séries temporelles. On définit un bruit blanc comme étant une suite de variables aléatoires non corrélés ayant même distribution et indépendants. Les éléments constructifs d'un bruit blanc sont appelés *innovations*.

La marche aléatoire est un autre exemple de processus stochastique qui est souvent utilisé en modélisation des rendements actifs boursiers. On dira que $\{a_t, t \in N\}$ est une marche aléatoire si $a_t = a_{t-1} + E_t$ ou $\{E_t, t \in N\}$ est un bruit blanc de variance finie, la marche aléatoire est l'accumulation des chocs [1].

III. 3. 1. Stationnarité

La stationnarité traduit le fait que les propriétés statistiques du processus soient invariantes par la translation temporelle. De manière plus précise, on dira qu'un processus $\{Y_t\}$ est stationnaire si la distribution conjointe des variables $Y_{t_1} ..... Y_{t_\tau}$ coïncide avec celle des variables $Y_{t_1-k} ..... Y_{t_\tau-k}$ et cela quelle que soient k et τ. Il clair que le bruit blanc est un processus stationnaire, mais la marche aléatoire n'est pas stationnaire car la moyenne et variance dépendent de temps. En effet, on a :

$$E(a_t) = \mu t \quad \text{(III. 1)}$$

$$V(a_t) = \sigma^2 t \quad \text{(III. 2)}$$

Où μ et σ^2 désignent respectivement l'espérance et la variance des variables E_t. On relâche souvent l'hypothèse de stationnarité est en parle de stationnarité au sens large (aussi appelée stationnarité du second ordre, ou encore stationnarité faible). Un processus est stationnaire du second ordre si :

$$\forall t \in T, E(Y_t) = \mu \quad \text{(indépendant de t) (III. 3)}$$

$$\forall t \in T, E[(Y_t - \mu)(Y_{t-k} - \mu)] = \gamma_k \text{ (indépendant de t) (III. 4)}$$

et de plus, tous les coefficients d'autocorrélations ρ_k définies par :

$$\rho_k = \frac{\gamma_k}{V(Y_t)V(Y_{t-k})} = \frac{\gamma_k}{\gamma_0} \qquad \text{(III. 5)}$$

Les coefficients ρ_k sont indépendants de t : elles ne dépendent que du retard (lag) k.

Un processus stochastique est stationnaire $\{Y_t\}$ est dite normal (ou gaussien) si la distribution conjointe des variables $Y_{t1},\ldots Y_{tm}$ est de loi normal quels que soient $t_1 < t_2 \ldots < t_m$. On appellera coefficient d'autocorrélation (*ACF*) la fonction définie par :

$$ACF(k) = \rho_k \quad \text{(III. 6)}$$

La fonction *ACF* permet de mesurer à quel point une valeur observée est corrélée avec les précédentes valeurs. Elle reflète en quelque sort la «mémoire» de processus.

Remarquons qu'en pratique, nous ne disposons jamais des véritables valeurs des auto- corrélations ρ_k. Tout ce dont nous disposons, c'est la trajectoire observée $\{y_t$ t =1….n$\}$ et il nous est impossible d'en observer une autre. L'hypothèse nécessaire pour estimer les coefficients d'autocorrélations à partir des valeurs observées $\{y_t\}$. Ainsi si $\bar{y}$ désigne la moyenne échantillonnée $\frac{1}{n}\sum_{t=1}^{n} y_t$ du processus Y_t, les autocorrélations empiriques,

$$\hat{\rho}_k = \frac{\frac{1}{n-k}\sum_{t=k+1}^{n}(y_t - \bar{y})(y_{t-k} - \bar{y})}{\frac{1}{n}\sum_{t=k+1}^{n}(y_t - \bar{y})^2} \quad \text{(III. 7)}$$

III. 3. 2. Filtre linéaire

Un théorème fondamental en analyse des séries temporelles est celui de Wold qui peut s'énoncer comme suit: tout processus stochastique $\{Y_t\}$ stationnaire peut s'écrire sous la forme [3],[4]:

$$Y_t - \mu = \sum_{j=0}^{+\infty} \psi_j E_{t-j} \quad \text{(III. 8)}$$

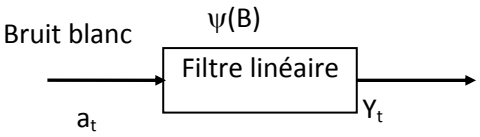


Figure.III.1 Représentation d'un filtre

Ou $\{E_t\}$ est un bruit blanc de variance σ^2 ,$\{\Psi_j\}$ est une suite réelle telle que $\Psi_0=1$.On introduit généralement l'opérateur retard *L* (Lag opérator) défini par : $L(Y_t) = Y_{t-1}$ donc la composition de Wold peut alors s'écrire :

$$Y_t - \mu = \sum_{j=0}^{+\infty} \psi_j L^j(E_t) \quad \text{(III. 9)}$$

L'opérateur $\sum_{j=0}^{+\infty} \psi_j L^j$ est appelé *le filtre linéaire* et la décomposition de Wold est appelée écriture *moyenne mobile infinie* du processus $\{Y_t\}$. La convergence de la suite $\sum_{j=0}^{+\infty} \psi_j$ assure la stationnarité de Y_t comme la montre l'équation suivante :

$$V(Y_t) = \sigma^2 \sum_{j=0}^{\infty} \psi_j^2 \quad \text{(III. 10)}$$

Puisque le processus $\{Y_t\}$ est stationnaire nous savons que les coefficients d'autocorrélations du processus sont finit et indépendants de t, on peut les calculer en fonction des ψ_j :

$$\rho_k = \frac{E[(Y_t - \mu)(Y_{t-k} - \mu)]}{E[(Y_t - \mu)^2]}$$

$$= \frac{E(\sum_{j=0}^{+\infty} \psi_j E_{t-j} \sum_{i=0}^{+\infty} \psi_i E_{t-k-i})}{\sigma^2 \sum_{j=0}^{+\infty} \psi_j^2}$$

$$= \frac{\sum_{j=0}^{+\infty} \psi_j \psi_{k+j}}{\sum_{j=0}^{+\infty} \psi_j^2} \quad \text{(III. 11)}$$

L'opération de filtrage linéaire est la base de la construction des processus stochastiques linéaires. Ceux-ci sont obtenus en particularisant le filtre (III.9) de tel sorte qu'il soit utilisable pratiquement, c'est à dire qu'il possède un nombre fini des paramètres [2],[3].

III.3.3. Processus ARMA

La représentation (III.9) de filtre linéaire général est difficile à implanter, en pratique, à cause de l'infinité du nombre de paramètres utilisés ψ_j. Maintenant en va considérer une importante classe de filtre linéaire très répandu en modélisation stochastique de séries temporelles. Ils résultent de la combinaison de processus *AutoRégressif* (AR) et *moyenne mobile* (MA). On supposera, pour la facilité d'écriture de ces filtres linéaires, que $\mu = 0$. Cette condition n'est pas restrictive car si un processus stationnaire ne vérifie pas cette condition, il suffit de considérer le processus $\{Y_t - \mu\}$

III.3.3. 1. Processus Autorégressif

Le processus autorégressif est un filtre linéaire particulier, dans ce processus, la présente valeur du processus égal à la somme de pondérations des précédentes valeurs et les innovations de bruit blanc. Ecrivons la décomposition de Wold dans le cas ou $\mu = 0$ et $\psi_j = \phi^j$:

$$Y_t = \sum_{j=0}^{+\infty} \phi^j E_{t-j} \quad \text{(III. 12)}$$

$$= \phi Y_{t-1} + E_t$$

Le processus qui en résulte est appelé *le processus autorégressif d'ordre* 1 et sera noté AR(1), cette appellation vient du fait que la dernière équation est l'analogue d'un modèle de régressions linéaires à la différence que les répresseurs sont ici les valeurs

observées des *p* derniers variables retardées $Y_{t-j}(j=1,\ldots\ldots,\ p)$. Notons que la condition de stationnarité est la convergence de $\sum_{j=1}^{+\infty}\phi^{j}$, $|\phi| \prec 1$.

Dans le cas AR(1) stationnaire, le coefficient d'autocorrélation *ACF* défini par l'équation en (III.4) devient:

$$\rho_k = \phi^k \quad \text{(III.13)}$$

La fonction *ACF* décroît géométriquement vers le 0 en valeur absolue la décroissance sera plus d'autant plus lente que $|\phi|$ est proche de la frontière du disque de stationnarité [-1,1].

La figure III.2 présente deux processus AR(1). Pour le premier ϕ=0.75, et pour le deuxième ϕ=-0.75. Dans les deux cas, la variance σ^2 de bruit blanc vaut 1, et $Y_1 = 0$. Lorsque ϕ est positif, les valeurs de Y_t et Y_{t+1} sont positivement corrélées, ce qui traduit sur la figure par une conservation de la tendance. Au contraire, lorsque ϕ est négatif, la corrélation négative a pour effet d'inverser la tendance, ce qui produit un graphe plus haché que dans le cas précédent.

En utilisant l'opérateur retard *L*, un modèle AR(1) s'écrira :

$$(1-\phi L)Y_t = E_t \quad \text{(III.14)}$$

On peut généraliser ce qui précède au cas d'un modèle AR(*p*), un modèle autorégressif d'ordre *p*. Un tel modèle s'écrira :

$$(1-\phi_1 L-\phi_2 L^2-\ldots-\phi_P L^P)Y_t = E_t \quad \text{(III.15)}$$

Ou encore :

$$\Phi_p(L)Y_t = E_t \quad \text{(III.16)}$$

où $\Phi_p(L)$ est le polynôme autorégressif de degré *p* en *L*.

Dans le cas général d'un modèle AR(p), la racine 1/ϕ est obligatoirement située à l'extérieur de l'intervalle [-1,1]. Dans le cas général d'un modèle AR(*p*), la condition de stationnarité est équivalente au fait que toutes les racines $\Phi_p(L)$ soient situées à l'extérieur du cercle unité [10]. Pour calculer Le coefficient *ACF* dans le processus AR(*p*), on supposera que le processus est stationnaire [5].

$$\rho_k = \phi_1\rho_{k-1} + \phi_2\rho_{k-2}\ldots\ldots\phi_p\rho_{k-p} \quad \text{(III.17)}$$

La relation (III.17) engendre un système d'équations appelées équations de Yule Walker:

$$\begin{cases} \rho_1 = \phi_1 + \phi_2\rho_1 + \ldots\phi_{p-1}\rho_{p-1} \\ \rho_2 = \phi_1\rho_1 + \phi_2 + \ldots\phi_{p-1}\rho_{p-2} \\ \vdots \ = \qquad \vdots \\ \rho_p = \phi_1\rho_{p-1} + \phi_2\rho_{p-2} + \ldots\phi_{p.} \end{cases} \quad \text{(III.18)}$$

L'inversion de ce système permet d'obtenir les paramètres ϕ_j du modèle en fonction des *p* coefficients de corrélation, la valeur de ϕ_p est donnée par:

$$\phi_p = \frac{\begin{vmatrix} 1 & \rho_1 & \cdots & \rho_{p-2} & \rho_1 \\ \rho_1 & 1 & \cdots & \rho_{p-3} & \rho_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho_{p-1} & \rho_{p-2} & \cdots & \rho_1 & \rho_p \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \cdots & \rho_{p-2} & \rho_{p-1} \\ \rho_1 & 1 & \cdots & \rho_{p-3} & \rho_{p-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho_{p-1} & \rho_{p-2} & \cdots & \rho_1 & 1 \end{vmatrix}} \qquad \text{(III.19)}$$

ϕ_p Est le coefficient de corrélation partielle des variables Y_t et Y_{t-p} qui mesure la corrélation entre ces deux variables.

La valeur de la fonction d'autocorrélation partielle (*PACF*) π_k est souvent utilisée pour déterminer l'ordre d'autocorrélation d'un modèle autorégressif [7].

Pour les modèles simples AR(1) et AR(2) la définition (III.12) donne :

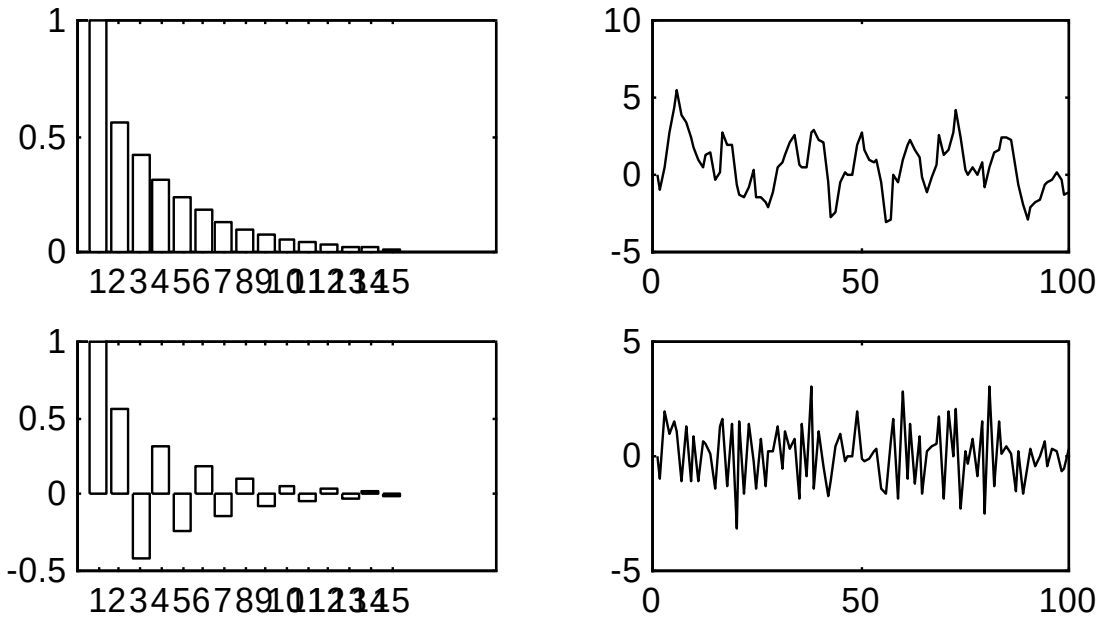


Figure III.2. *A droite*: deux trajectoires provenant de processus AR(1). *A gauche*: les corrélogrammes correspondants.

$$\text{AR(1):}\ \pi_1 = \rho_1, \qquad \pi_k = 0 \quad k > 1$$

$$\text{AR(2):}\ \pi_1 = \rho_1\ , \qquad \pi_2 = \frac{\rho_2 - \rho_1^{\,2}}{1 - \rho_1^{\,2}}, \qquad \pi_k = 0 \quad k > 2$$

De façon générale, pour un modèle AR(p), la fonction *PACF* sera nulle pour k>p [2].

III.3.3.2. Processus moyenne mobile

Considérons maintenant la décomposition de Wold dans laquelle μ=0, ψ_1=- θ, ψ_j=0 si j>1 :

$$\begin{aligned} Y_t &= E_t - \theta E_{t-1} \\ &= (1 - L\theta)E_t \end{aligned} \qquad \text{(III.20)}$$

Le processus qui en résulte est appelé *processus moyenne mobile d'ordre* 1 et sera noté MA(1) (Moving Average Process), le coefficient d'autocorrélation *ACF* défini en (III.7) devient :

$$\begin{aligned} \rho_k &= \frac{-\theta}{1+\theta^2} \quad si \quad k = 1 \\ &= 0 \quad si \quad k \succ 1 \end{aligned} \qquad \text{(III.21)}$$

on peut généraliser ce qui précède au cas d'un modèle MA(q), un modèle moyenne mobile d'ordre q un tel modèle s'écrira :

$$\begin{aligned} Y_t &= (1 - \theta_1 L^3 - \theta_2 L^2 ..... \theta_q L^q)E^t \\ &= \Theta_q(L)E_t \end{aligned} \qquad \text{(III.22)}$$

Où $\Theta_q(L)$ désigne le polynôme moyen mobile de degré q en L. La condition de stationnarité est toujours vérifiée pour un modèle MA(q) puisque le filtre linéaire ne comporte ici qu'un nombre fini de termes.

En réalité, la stationnarité traduit la condition sous la quelle le processus $\{Y_t\}$ admet une écriture de type moyenne mobile. Si l'on cherche la condition permettant une écriture autorégressive du processus, alors on parlera de la condition d'inversibilité. Evidemment tout processus autorégressif est inversible, mais ce n'est pas le cas pour un processus moyen mobile. En effet, essayons d'inverser le modèle MA(1) :

$$\begin{aligned} E_t &= Y_t + \theta E_{t-1} \\ &= Y_t + \theta Y_{t-1} + \theta^2 E_{t-2} \\ &= Y_t + \theta Y_{t-1} + \theta^2 Y_{t-2} + \theta^3 Y_{t-3} ..... \end{aligned} \qquad \text{(III.23)}$$

La condition d'inversibilité est bien sur la racine (1/θ) de $\Theta_1(L)$ soit située à l'extérieur de cercle unité. Dans le cas d'un modèle MA(q), la condition de d'inversibilité est que toutes les racines du polynôme moyen mobile $\Theta_q(L)$ soient situées à l'extérieur du cercle unité [10].

Le coefficient d'autocorrélation d'un processus MA(q) est donné par :

$$\begin{aligned} \rho_k &= \frac{-\theta_k + \theta_1\theta_{k-1} + ...\theta_q\theta_{k-q}}{1+\theta_1^2 + ...\theta_q^2} \quad si \quad k = 1,2...q \\ &= 0 \quad si \quad k > q \end{aligned} \qquad \text{(III.24)}$$

Dans le cas d'un processus MA(q) inversible. La fonction *ACF* est nulle pour le cas de k>q alors que la fonction *PACF* peut prendre une infinité de valeurs non nulles. L'allure de la fonction *PACF* est similaire à celle de la fonction *ACF* pour un modèle autorégressif, elle résulte d'une combinaison de décroissance géométrique et de sinusoïde amortie.

III.3.3. 3. Le processus mixte autorégressif, moyen mobile

Un processus ARMA(p, q) résulte d'une combinaison d'un modèle AR(p)et MA(q).

Il s'exprime sous forme :

$$\Phi_p(L)Y_t = \Theta_q(L)E_t . \qquad \text{(III.25)}$$

Un tel processus est stationnaire si la partie autorégressive du processus est stationnaire, c'est-à-dire si les racines du $\Phi_p(L)$ sont situées à l'extérieur du cercle unité, et est inversible si la partie moyenne mobile du processus est inversible, c'est–à-dire les racines de $\Theta_q(L)$sont situées à l'extérieur du cercle unité[9] ,[11].
Considérons par exemple le cas d'un processus ARMA(1,1) :

$$(1-\phi L)Y_t = (1-\theta L)E_t . \qquad \text{(III.26)}$$

L'écriture moyenne mobile du processus est possible si les racines de $\Phi_p(L)$ sont situées à l'extérieur du cercle unité, puisqu'en ce cas, la partie AR possède une écriture moyenne mobile

$$Y_t = \frac{(1-\theta L)}{(1-\phi L)}E_t \qquad \text{(III.27)}$$

Le coefficient d'autocorrélation *ACF* d'un processus ARMA(1,1) est donné par :

$$\rho_1 = \frac{(1-\phi\theta)(\phi-\theta)}{(1+\theta^2-2\theta\phi)} \qquad \text{(III.28)}$$

En conclusion, la fonction *ACF* d'un modèle ARMA(1,1) sera similaire à celle d'un modèle AR(1) excepté le fait que la décroissance géométrique des ρ_k commence à partir de ρ_1 ou lieu de ρ_0 , De façon générale, la fonction *ACF* d'un modèle ARMA(p,q) aura la même allure que la fonction *ACF* d'un modèle AR(p) à partir de k > p+1, et la fonction *PACF* d'un processus ARMA(p,q) aura la même allure que la fonction *PACF* d'un modèle MA(q) à partir de k > q+1.

III. 4. Processus ARIMA et SARIMA

La classe des modèles ARMA, présentée précédemment, suppose que le processus stochastique ayant engendré la trajectoire observée est stationnaire. Ceci implique que les moments d'ordre1 et 2 sont invariants dans le temps par translation temporelle. En particulier, la moyenne et la variance de Y_t sont supposées constantes. Cependant, en pratique, l'hypothèse de stationnarité apparaît souvent comme peu conforme à la réalité et il est habituel, par exemple en économie et l'industrie, d'observer des séries exposent un comportement non stationnaire dont la moyenne et la variance changent au cours du temps[14],[6].

III. 4. 1. Non stationnarité en moyenne

Une des causes évidentes de non stationnarité d'une série temporelle est la présence d'une tendance. La figure. III.3(a) montre le graphe d'une série temporelle (Série A) relative à la température journalière enregistrée à midi sur Ben Nevis, du 1 février au 18 août1884 [4]. La trajectoire observée ne peut provenir d'un processus $\{Y_t\}$ de moyenne constante au cours du temps. On dira que la série A présente une légère tendance à la hausse[12].

Supposons une série Y_t puisse se composer en une composante déterministe (son niveau moyen μ_t) et une composante aléatoire stationnaire E_t.

$$Y_t = \mu_t + E_T \qquad \text{(III.29)}$$

où μ_t est un polynôme $P_d(t)$ de degré d en t. Dans ce cas, si les coefficients du polynôme $P_d(t)$ sont constants, on dira que le processus présente un trend polynomial déterministe d'ordre *d.*

Pratiquement, on se débarrasse d'une telle tendance en appliquant à la série chronologique le filtre différence d'ordre *d* défini par:

$$\nabla^d = (1-L)^d \qquad \text{(III.30)}$$

Par exemple, pour un trend linéaire (d=1) on a :

$$Y_t = a + bt + E_t \qquad \text{(III.31)}$$

En appliquant le filtre différence d'ordre 1 à la série Y_t.

$$\nabla Y_t = \nabla(a + bt + E_t)$$

$$Y_t - Y_{t-1} = b + E_t - E_{t-1} \qquad \text{(III.32)}$$

Cette dernière équation d'un modèle $ARMA(1,1)$ avec $\phi = 1, \theta = 1$. On sait que ce modèle n'est pas stationnaire, ni inversible puisque les racines des polynômes autorégressif et moyenne mobile sont sur la frontière du domaine de stationnarité a figure III.3(b) présente la série des différences d'ordre 1 relatives à la série A. On voit que cette opération a gommé la tendance linéaire à la hausse. Généralement, un filtrage d'ordre *d* =1 ou *d* =2 est suffisant pour aborder la tendance. Lorsque la série filtrée $\nabla^d(Y_t)$ est stationnaire on dit que le processus $\{Y_t\}$ *est intégrée d'ordre d.*

Lorsqu'une série est intégrée d'ordre *d*, un modèle $ARMA(p,q)$ ARMA (*p*,*q*) appliquée au différence

$\nabla^d(Y_t)$ engendre le modèle $ARIMA(p,d,q)$

$$\Phi_p(L)\nabla^d(Y_t) = \Theta_q(B)E_t \qquad \text{(III.33)}$$

Il est clair que cette relation analogue d'un $ARMA(p+d,q)$ pour lequel le polynôme autorégressif $\Phi_{p+d}(L)$ admet 1 comme racine d'ordre d. On peut donc écrire un modèle $ARIMA(p,d,q)$ sous la forme.

$$\Phi_{p+d}(L)(Y_t) = \Theta_q(B)E_t \qquad \text{(III.34)}$$

III.4. 2. Non stationnarité en variance

Relâchons l'hypothèse concernant les erreurs E_t en supposant que le processus $\{E_t\}$ n'est pas stationnaire en variance, c'est à dire que sa variance n'est pas constante. Plus précisément, on suspecte une dépendance vis-à-vis du niveau moyen μ_t de la forme :

$$V(E_t) = \sigma^2 h^2(\mu_t) \qquad \text{(III.35)}$$

Ou h est une fonction connue.

On désire stabiliser la variance du processus à l'aide d'une transformation g dérivable. En développant $g(Y_t)$ autour de μ_t et en notant $g'(\mu_t)$ la dérivée première de g évaluée en μ_t, il devient :

$$g(Y_t) \cong g(\mu_t) + (Y_t - \mu_t)g'(\mu_t) \qquad \text{(III.36)}$$

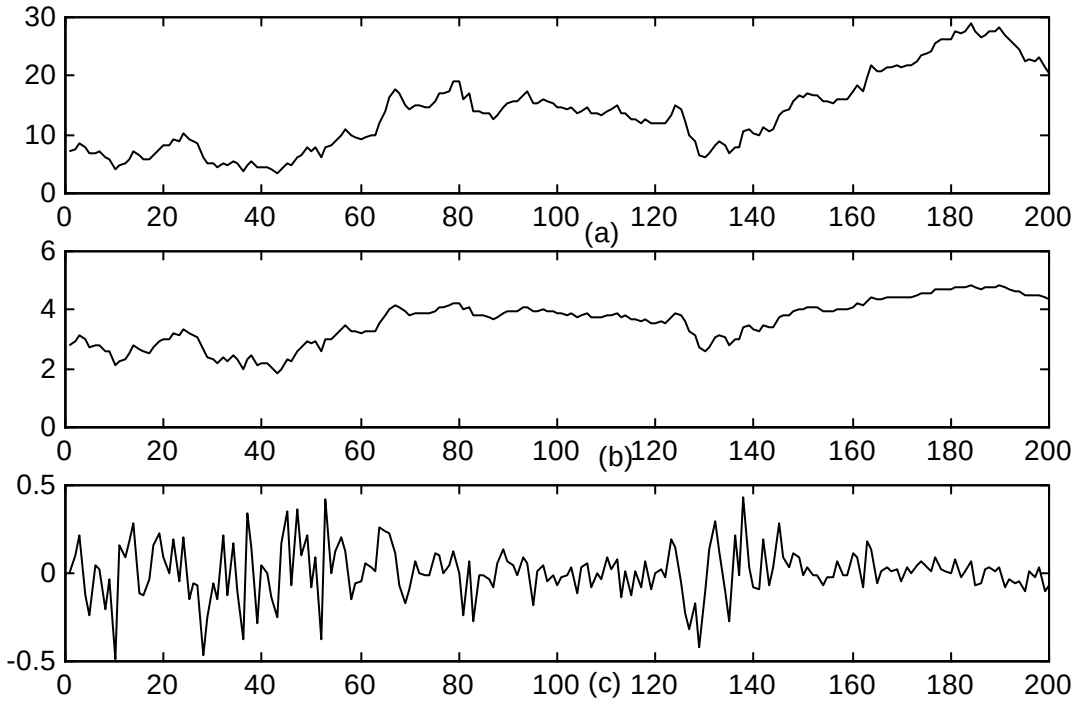


Figure III.4. (a) Série B. (b) Représentation du logarithme des valeurs constitutives de la série.

De ce fait, la variance de $g(Y_t)$ peut être approchée par :

$$V(g(Y_t)) \cong \sigma^2[g'(\mu_t)]^2 h^2(\mu) \qquad \text{(III.37)}$$

Afin de stabiliser la variance, il faut donc choisir g tel que

$$g'(\mu_{tt}) = \frac{1}{h(\mu_t)} \qquad \text{(III.38)}$$

par exemple, lorsque h(μt)=μt, l'écart type de Yt est proportionnel à μt :

Au plus le niveau moyen μt augmente, au plus les variations observées sur la série sont importantes. Dans ce cas, en supposant les valeurs observables du processus, la transformation requise pour stabiliser la variance est le logarithme. C'est le cas de la série B représentée à la figure III. 4.(a).

Cette série se rapporte à l'évolution mensuelle de l'indice boursier FTA pour la période janvier 1965 à décembre 1990 [12],[1]. On distingue clairement une tendance non linéaire à la hausse sur laquelle prennent place les variations de prix, augmentant en amplitude avec le niveau moyen de la série. Une transformation logarithmique est ici

appropriée pour stabiliser la variance et linéairiser la tendance (cette transformation est représentée à la figure III.4.(b) après quoi une différenciation d'ordre 1 élimine la tendance. La figure III.4.(c) représente la série transformée. On voit maintenant que la série semble stationnaire en moyenne et en variance. La transformation logarithmique est sans doute la plus utilisée pour les séries économiques.

Cette transformation, connue ce le nom de *transformations de Box-Cox,* est définie par :

$$g(Y_t) = \ln(Y_t) \qquad \text{(III.39)}$$

Ou Y_t supposé positif.

III.4. 2. Saisonnalité

Une autre opération de prétraitement concerne des séries temporelles présentant un caractère cyclique comme par exemple le chiffre d'affaire mensuel d'un grand magasin ou encore la température moyennes mensuelles, on parle alors de la saisonnalité ou de tendance saisonnière[7] ,[8].

Une différenciation saisonnière de période *s* (s'étant la duré de saison) et d'ordre *D* définie par :

$$\nabla_s^D = (1-L^s)^D \qquad \text{(III.40)}$$

est envisageable pour rendre une série saisonnière stationnaire. La figure I.5.(a) représente la série C relative aux températures aux moyennes mensuelles enregistrées au château de Nottingham pour la période janvier 1920 à décembre 1939 [4]. Une différentiation saisonnière de période s =12 et d'ordre D = 1 permet de stationnariser la série (figure III. 5(b)).

Deux façons de procéder sont envisageables en présence saisonnalité

1-Soit appliquer un modèle ARIMA après différenciation saisonnière :

$$\Phi_p(L)\nabla^d\nabla_s^D(Y_t) = \Theta_q(L)E_t \qquad \text{(III.41)}$$

2- Soit on a inclut le caractère saisonnier directement dans le modèle ARIMA :

$$\Phi_p(L)\Phi_P(L^s)\nabla^d\nabla^D(Y_t) = \Theta_q(L)\Theta_Q(L^s)E_t \qquad \text{(III.42)}$$

Où p, P, q, Q et d, D sont des entiers non négatifs, $\Phi_p(L)$ et $\Theta q(L)$ sont des polynômes en L où $\Phi_p(L^s)$ et $\Theta q(L^s)$ sont les polynômes saisonniers en L^s. Ce dernier modèle est appelé *Structurel Time Series.* Un tel modèle est stationnaire, si les racines des polynômes $\Phi_p(L^s)$ et $\Phi_p(L)$ sont à l'extérieur du cercle unité. Si les racines de $\Theta q(L)$ et $\Theta q(L^s)$ sont à l'extérieur de cercle unité le processus est inversable [11].

III. 5. Méthodologie de Box et Jenkis

Box et Jenkis [10] on introduit une méthodologie pour la construction d'un modèle linéaire qui s'ajuste au mieux à une série temporelle. Cette méthodologie se décompose

en trois étapes: l'identification du modèle, l'estimation des paramètres et la validation du modèle. La figure.III.6 résume la procédure pour un modèle ARIMA[13] , [14].

III. 5. 1. Identification

Cette étape consiste à déterminer les paramètres p, d, q ainsi que les P, D, Q à partir des graphes des fonctions ACF (corrélogrammes) et PACF (Partiels corrélogrammes). Cette étape est la plus délicate car dans la pratique, les corrélogrammes observés, identification du modèle, n'engendrent pas un choix évident. Les allures (théoriques) des fonctions ACF et PACF ont été reprises au tableau III.1

On remarque qu'en pratique, on dispose jamais des véritables valeurs des coefficients d'autocorrélations *ACF* noté ρ_k ni des coefficients autocorrélations partielles P*ACF* notés π_k. Mais nous pouvons en donner des estimations $\hat{\rho}_k$ et $\hat{\pi}_k$ tel que :

$$\hat{\rho}_k = \frac{\frac{1}{n-k}\sum_{t=k+1}^{n}(y_t-\bar{y})(y_{t-k}-\bar{y})}{\frac{1}{n}\sum_{t=1}^{n}(y_t-\bar{y})^2} \qquad \text{(III.43)}$$

Il ne faut donc pas perdre de vue le comportement théorique présenté dans le tableau III.1 correspondent aux fonctions théoriques alors que ce que nous représentons sur les corrélogrammes correspond aux estimations des coefficients de corrélations et coefficients partiels, ce qui rend souvent le diagnostic difficile.

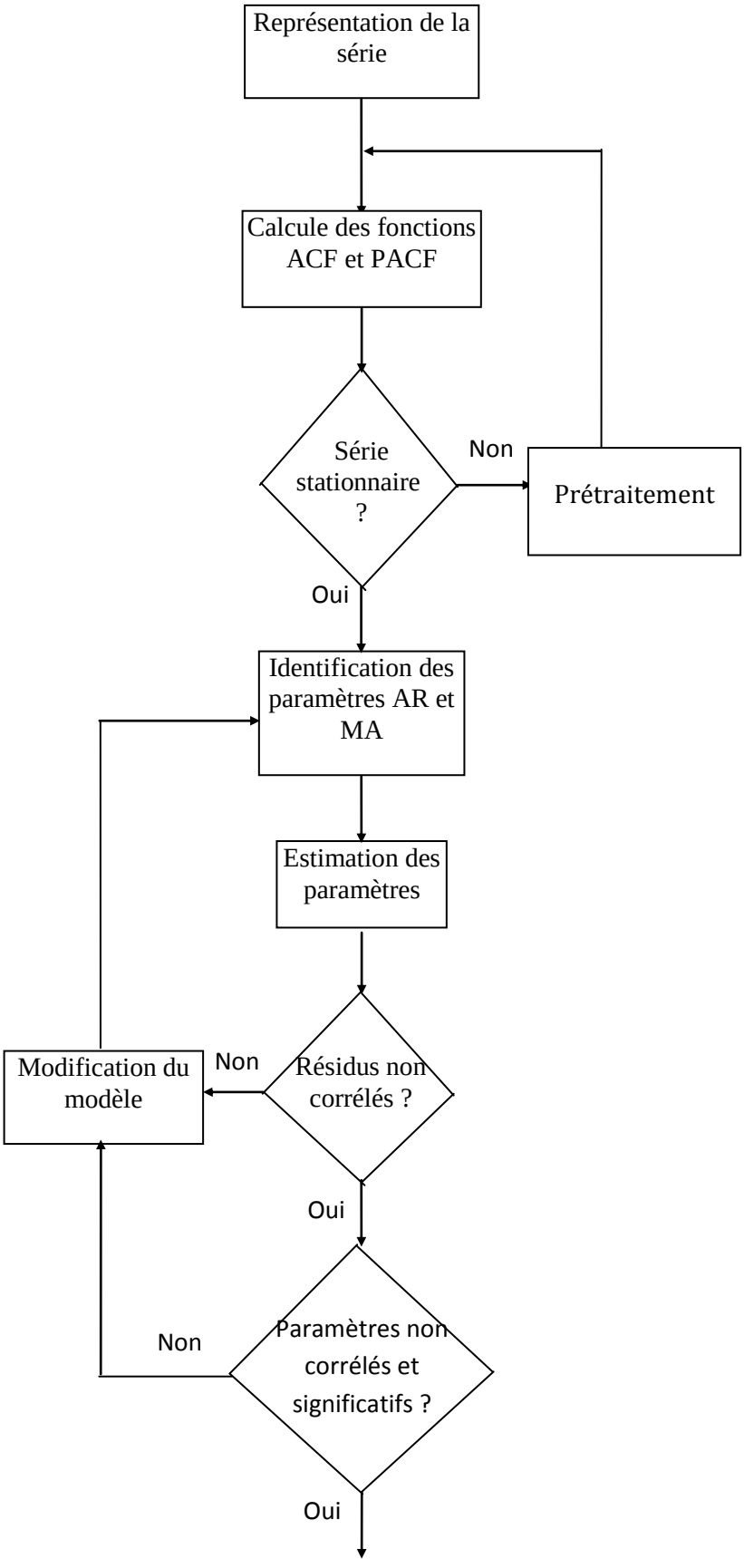


Figure III.6.Méthodologie Box et Jenkins

	ACF	***PACF***
AR(p)	Mélange de sinusoïdes amorties et décroissances géométriques	Nulle au-delà de p
MA(q)	Nulle au-delà de q	Mélange de sinusoïdes amorties et dé croissances géométriques
ARMA(p,q)	Mélange de sinusoïdes amorties et décroissances géométriques au-delà de q.	Mélange de sinusoïdes amorties et décroissances géométriques au-delà de p.

Tableau III.1. Identification des modèles AR, MA, ARMA à partir des fonctions *ACF* et *PACF*

Les figures III.7,……, III.12 présentent les corrélogrammes et les corrélogrammes partiels relatifs à trois processus stationnaires. Il s'agit de processus AR(4), MA(2) et ARMA(4,2). Si pour les modèles AR(4) et MA(2), les observations des fonctions *ACF* et *PACF* permettent de déterminer facilement l'ordre des modèles, il est plus difficile de deviner les ordres *p* et *q* pour le modèle ARMA car nous savons pour ce cas les *q* premiers ρ de *ACF* sont perturbés par la composante MA et les *p* premiers π de *PACF* sont perturbés par la composante AR. En outre, dans le cas de séries réelles les corrélogrammes ne sont pas aussi nets que ceux présentés dans ces trois simulations.

Pour compléter l'observation des corrélogrammes, il est utile de déterminer si les $\hat{\rho}_k$ et $\hat{\pi}_k$ observés sont significativement différent de 0. Pour un modèle AR(*p*), on peut représenter la suite des $\hat{\pi}_k$ et voir à partir de quelle valeur de *k* les autocorrélations partielles restent dans la bande de non-signification [14].

$$\pm 1.96/\sqrt{n} \qquad \text{(III.44)}$$

On pourra identifier l'ordre du modèle MA(*q*), en examinant à partir de quelle valeur du retard k la suite des coefficients d'autocorrélations partielles restent dans la bande non-signification [9].

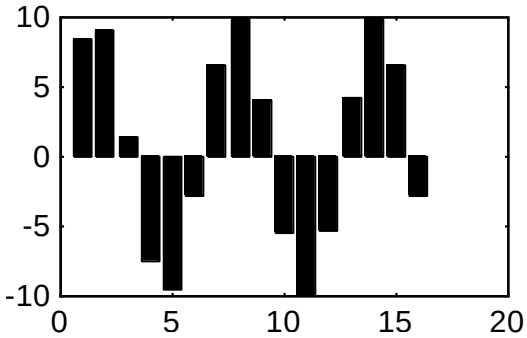


Figure III.7. *ACF AR* (4)

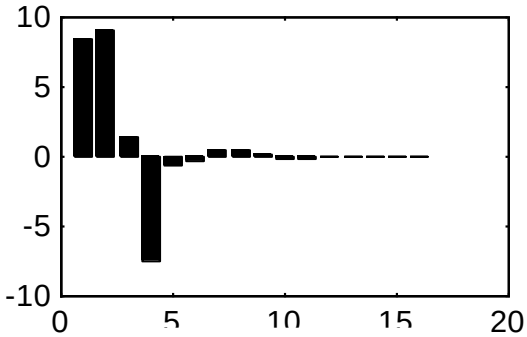


Figure III.8. *PACF AR* (4)

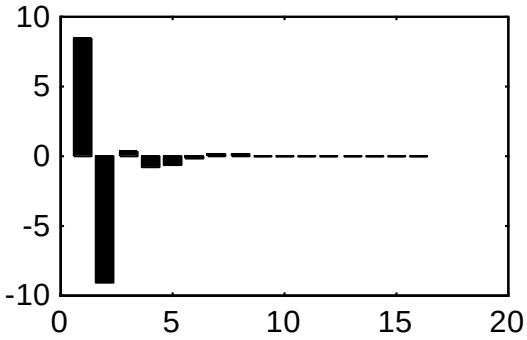


Figure III.9. *ACF MA*(2)

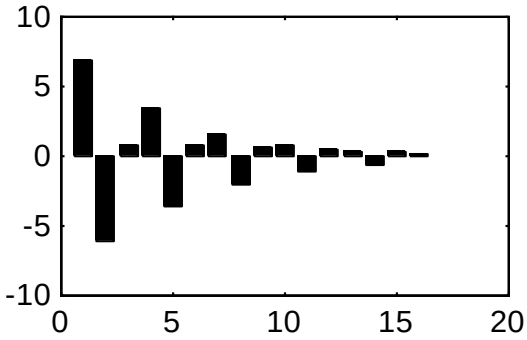


Figure III.10. *PACF MA*(2)

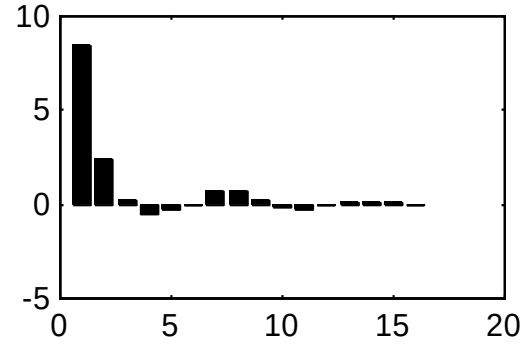


Figure III.11. *ACF ARMA* (4,2)

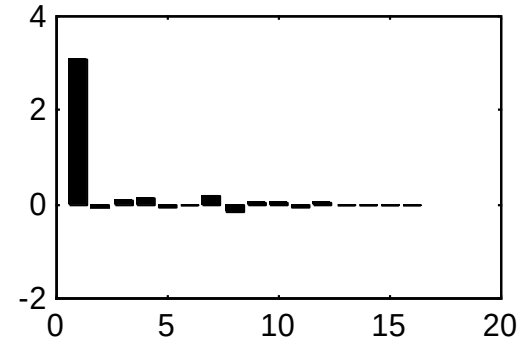


Figure III.12. *PACF ARMA*

$$\pm 1.96/\sqrt{\frac{1}{n}(1+2\sum_{t=1}^{k}\hat{\rho}_t^{\,2})} \qquad \text{(III.45)}.$$

Pour les premiers autocorrélations, cette bande correspond à la bande (III.44) et ensuite s'élargit progressivement avec k. Considérons la série B, pour laquelle les fonctions *ACF* et *PACF* sont représentées aux figures III.13 et III.14. La lente décroissance d'ACF est un signe caractéristique de non-stationnarité. Nous avons déjà fait remarquer qu'une transformation logarithmique suivie d'une différentiation d'ordre 1 permet de stationnariser la série (figure III.15, III.16).

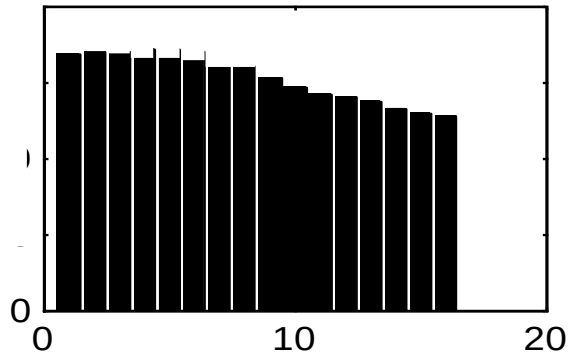


Figure III.13. Fonction *ACF* pour la série B

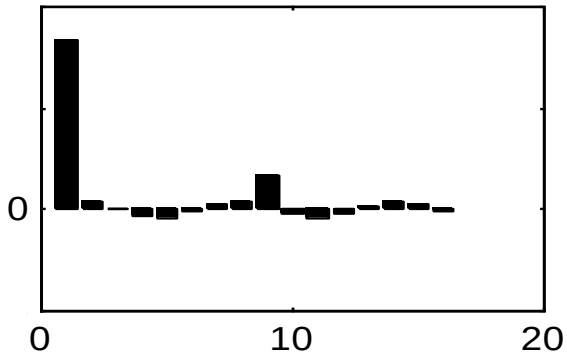


Figure III.14**.** Fonction *PACF* pour la série

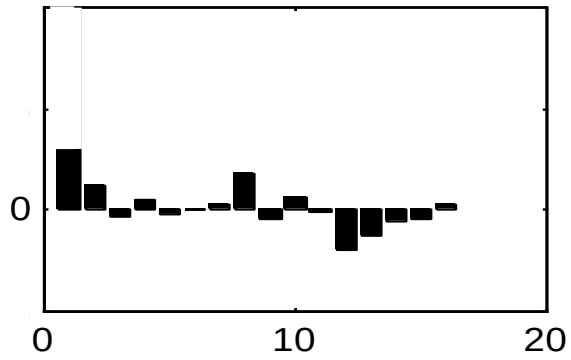


Figure III.15**.** Fonction *ACF* pour la série B transformé

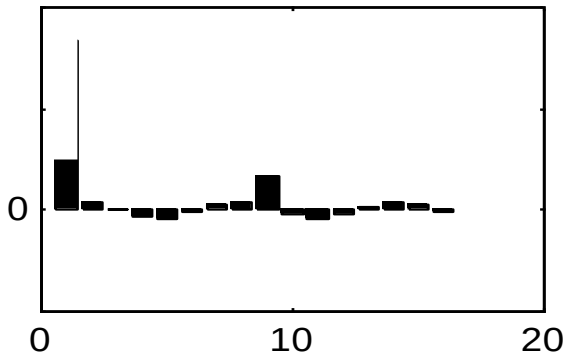


Figure III.16**.** Fonction *PACF* pour la série B transformé

III. 5. 2. Estimation

Les méthodes utilisées pour l'estimation des paramètres du modèle reposent sur le principe du maximum de vraisemblance (ML). Moyennant certaines simplifications, on est ramené à un problème d'optimisation de type moindre carré non linéaire et on utilise généralement la méthode de Marquardt [13].

Le fait que l'optimisation soit non-linéaire en les paramètres liés la composante moyenne mobile du modèle car on sait que pour un modèle purement autorégressif, les équations linéaires de Yule-Walker (I.18) permettant l'estimation des paramètres.

Expliquons plus précisément ce qui précède, en calculant les estimateurs ML des paramètres dans le cas d'un modèle MA(q) de paramètre θ. Pour un MA(1), on a les relations suivantes :

$$\begin{aligned} E_0 &= E_0 \\ E_1 &= Y_1 - \theta E_0 \\ E_2 &= Y_2 - \theta E_1 = \theta Y_1 + Y_2 + \theta^2 E_0 \\ E_3 &= Y_3 - \theta E_2 = \theta^2 Y_1 - \theta Y_2 + Y_3 - \theta^3 E_0 \\ &\vdots \end{aligned} \qquad \text{(III.46)}$$

sous la forme matricielle, les relations précédentes s'écrivent :

$$\underset{(n+1,1)}{E} = \begin{pmatrix} 0 \\ A(\theta) \\ (n,n) \end{pmatrix} \underset{(n,1)}{Y} + \begin{pmatrix} 1 \\ B(\theta) \\ (n,n) \end{pmatrix} \underset{(1,1)}{E}^{(0)}$$

ou (III.47)

$$E = \begin{pmatrix} E_0 \\ E_1 \\ \vdots \\ E_n \end{pmatrix}, Y = \begin{pmatrix} Y_0 \\ Y_1 \\ \vdots \\ Y_n \end{pmatrix}, E^{(0)} = E_0$$

et

$$A(\theta) = \begin{pmatrix} 1 & & & & \\ -\theta & 1 & & & \\ \theta^2 & -\theta & 1 & & \\ -\theta^3 & \ddots & \ddots & \ddots & \\ \ddots & -\theta^3 & \theta^2 & -\theta & 1 \end{pmatrix}; B(\theta) = \begin{pmatrix} -\theta \\ \theta^2 \\ -\theta^3 \\ \vdots \\ (-1)^n \theta^n \end{pmatrix}$$

Remarquons que ce système nécessite la connaissance d'une condition initiale E_0 à fixer, ce qui est tout à fait normal, étant donnée l'équation de récurrence suivante, caractérisant un MA(1) fait intervenir l'erreur au temps précédent. L'équation précédente peut aussi s'écrire

$$\underset{(n+1,1)}{E} = \underset{(n+1,n)}{P(\theta)} \underset{(n,1)}{Y} + \underset{(n+1,1)}{Q(\theta)} \underset{(1,1)}{E} \qquad \text{(III.48)}$$

l'équation (III.48) peut se généraliser au cas d'un MA(q) ; il suffit d'adapter les dimensions des matrices. On a alors $\underset{(n+q,n)}{P}$, $\underset{(n+q,q)}{Q}$, $\underset{(n+q,1)}{E}$ et dans ce cas la connaissance d'un vecteur initial $E^{(0)} = (E_0, \cdots, E_{1-q})'$ est requise.

La méthode ML se simplifie si on suppose que la distribution *Et* suit la loi normale de variance σ^2. Dans ce cas, calculons la densité du vecteur aléatoire $\begin{pmatrix} E^{(0)} \\ Y \end{pmatrix}$ relatif au modèle MA(q). La relation III.27 peut s'écrire: $E = \begin{pmatrix} Q & P \end{pmatrix} \begin{pmatrix} E^{(0)} \\ Y \end{pmatrix}$

la matrice de transformation $\begin{pmatrix} Q & P \end{pmatrix}$ est triangulaire inférieure, la densité *p* de vecteur $\begin{pmatrix} E^{(0)} & Y \end{pmatrix}'$ s'obtient facilement à partir de celle de ***E :***

$$p((E^{(0)} \quad Y)') = (2\pi\sigma^2)^{-(n+q)/2} \exp\left\{-\frac{1}{2\sigma^2}(PY + QE^{(0)})'(PY + QE^{(0)})\right\} \quad \text{(III.49)}$$

La fonction de vraisemblance $l(Y,\theta_1,\cdots\theta_q,\sigma^2)$ s'obtient en intégrant la densité III.28 par rapport à $\boldsymbol{E}^{(0)}$, il vient :

$$l(Y,\theta,\sigma^2) = (2\pi\sigma^2)^{-n/2}\{\det(Q'Q)\}^{-1/2}\exp(\frac{-S(\theta)}{2\sigma^2})$$

$$L(Y,\theta,\sigma^2) = -\frac{-S(\theta)}{2\sigma^2} - \frac{n}{2}\ln(\sigma^2) - \frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln(\det(Q'Q)) \qquad \text{(III.50)}$$

avec

$$S(\theta) = \left\|PY + QE^*\right\|^2; E^* = (Q'Q)^{-1}Q'PY.$$

Il faut maintenant optimiser cette vraisemblance par rapport aux paramètres σ^2 et θ. On effectue généralement l'optimisation en deux temps; on cherche d'abord la valeur optimale de σ^2 en fixant le paramètre θ et on reporte ensuite cette valeur dans la log-vraisemblance L, ce qui permet d'obtenir une fonction à optimiser. On obtient

$$\sigma^2 = \frac{S(\theta)}{n}$$

$$L(Y,\theta,\sigma^2) = (-1/2)\left\{n(1+\ln(2\pi)) + n\ln(\frac{S(\theta)}{n}) + \frac{1}{2}\ln(\det(Q'Q))\right\} \qquad \text{(III.51)}$$

En d'autres termes, il s'agit de minimiser la fonction

$$f(\theta) = n\ln(S(\theta)) + \ln(\det(Q'Q)) \qquad \text{(III.52)}$$

III. 5. 3. Validation

Cette phase vise à savoir, si le modèle choisi en phase d'identification et dont les paramètres ont été estimés peut être considéré comme valable. En d'autre terme, il s'agit de savoir si la trajectoire observée peut provenir du modèle identifié. Dans l'affirmative, les résidus observés E_t sont la réalisation de bruit blanc. En plus d'examen visuel des fonctions *ACF* et *PACF*, on se réfère habituellement à certains tests connus sous le nom de tests de bruit blanc pour valider le modèle construit[2].

Ainsi, sous l'hypothèse que {*Et*} est un bruit blanc, Box et Pierce montrent que pour un modèle $ARMA(p,q)$ la statistique

$$Q_1(K) = n\sum_{k=1}^{K}\hat{\rho}_k \qquad \text{(III.53)}$$

Suit asymptotiquement $(n \to +\infty)$ une loi de X^2_{K-p-q}

Cependant, on a pu mettre empiriquement en évidence des différences assez grandes entre la loi asymptotiques de $Q_1(K)$ et sa distribution pour n fini, même pour les grandes valeurs de n. Ljung et Box ont proposé une statistique modifiée pour tenir compte de cette différence :

$$Q_2 = n(n+2)\sum_{k=1}^{K}(n-k)^{-1}\hat{\rho}_k \approx \chi^2_{K-p-q} \qquad \text{(III.54)}$$

Le choix de K dans les deux équations précédentes reste un problème délicat. Habituellement, les K sont choisis assez grands (entre 15 et 20) de telle sorte que les poids $\hat{\rho}_k$ soient négligeables au-delà de K. En procédant de la sorte, on accumule dans Q_1 la majeure partie des corrélations entre résidus. Une grande valeur de Q_1 ou Q_2 (c'est à dire une valeur statistiquement significative) signifié que le modèle n'est pas approprié. Cependant, une petite valeur de ces statistiques ne signifie pas forcément que le modèle est adéquat car il se peut qu'un $\hat{\rho}_k$ soit significatif mais qu'il passe inaperçu dans les équations de Q_1 et Q_2 si la valeur choisie pour K est trop grande. Une façon d'éviter ce problème est de calculer ces statistiques de manière cumulative, c'est-à-dire en les calculant pour plusieurs valeurs consécutives de K.

III. 6. Prévision

Lorsque la construction du modèle linéaire est terminée, on est à même de générer des prévisions, on notera *h* l'horizon de prévision, ce qui signifie que le modèle va être utilisé au temps t pour prévoir la valeur du processus au temps *t+h.* Pour ce qui suit, il est nécessaire de distinguer les variables aléatoires Y_t, E_t de leurs réalisations y_t, e_t

On notera $\hat{y}_t(h)$ la prévision faite en t concernant y_{t+h} (la valeur de processus au temps *t+h.* $\hat{y}_t(h)$ désignera l'estimateur correspondant. On distinguera aussi la variable aléatoire « erreur de prévision » $E_t(h)$ définie par $E_1(h) = y_{t+h} - \hat{y}_t(h)$ [2] ,[7].

Considérons tout d'abord le cas de la prévision à l'horizon *h*=1 à partir d'un modèle ARMA(1,1). On a la relation suivante :

$$Y_{t+1} = \phi Y_t - \theta E_t + E_{t-1} \qquad \text{(III.55)}$$

Le meilleur estimateur de y_{t+1} au sens des moindres carrés est donné par l'espérance conditionnelle $E(Y_{t+1} \setminus F_t)$. L'estimateur optimal de y_{t+1} est donc :

$$\hat{Y}_t(1) = E(\phi Y_t - \theta E_t + E_{t+1} \setminus F_t) \qquad \text{(III.56)}$$

la prévision optimale peut s'écrire :

$$\hat{y}_t(1) = \phi y_t - \theta e_t \qquad \text{(III.57)}$$

Notons au passage qu'en remplaçant e_t dans l'équation précédente, la dernière erreur observé $y_t - \hat{y}_{t-1}(1)$ et qu'on supposant $(0 \prec \theta \prec 1)$et $(\phi = 1)$, la formule précédente de prévision se ramène à la méthode de prévision appelée lissage exponentiel simple:

$$\hat{y}_t(1) = (1-\theta) y_t + \theta \hat{y}_{t-1}(1) \qquad \text{(III.58)}$$

Notons que la formule précédente est obtenue à partir d'un processus $\{Y_t\}$ non stationnaire $(\phi = 1)$ mais inversible $(0 \prec \theta \prec 1)$. D'autres méthodes de lissage existent, comme par exemple les lissages généralisés ou les méthodes dites de Holt-Winters. Leur domaine d'utilisation est essentiellement la prévision à court terme.

Dans le cas général d'un processus $ARIMA(p,q,d)$, on obtient les prévisions à horizon h on utilisant l'écriture $ARMA(p+d,q)$ déjà donnée

$$Y_{t+h} = \varphi_1 Y_{t+h-1} + \ldots + \varphi_{p+d} Y_{t+h-p-d} + E_{t+h} - \theta_1 E_{t+h-1} - \ldots - \theta_q E_{t+h-q} \qquad \text{(III.59)}$$

En prenant l'espérance conditionnelle dans les deux membres, il vient :

$$\hat{Y}_t(h) = \sum_{j=j\min}^{jamx} \varphi_j \mathrm{E}(Y_{t+j}/F_t) - \sum_{k=k\min}^{k\max} \theta_k \mathrm{E}(E_{t+k}/F_t) \qquad \text{(III.60)}$$

avec :

$j_{\min} = \min(h-p-d, h-1)$, $j_{\max} = \max(h-p-d, h-1)$

$k_{\min} = \min(h-q, h-1)$, $k_{\max} = \max(h-q, h-1)$

Pour calculer des espérances conditionnelles, il faut distinguer quatre cas selon le signe de j et k. On remarquera que Y_{t+j} (j<0) et E_{t+k} (k<0) sont observables donc mesurables que le temps t.

Ainsi, la prévision $\hat{y}_t(h)$ est donnée par :

$$\hat{y}_t(1) = \sum_{j=j\min}^{j\max} \phi_j y^*_{t+j} - \sum_{k=k\min}^{k\max} \theta_k e^*_{t+k} \qquad \text{(III.61)}$$

$y^*_{t+j}(h) = y_{t+j}(h)$ si j<1

$e^*_{t+k}(h) = y_{t+k} - \hat{y}_{t+k-1}(1)$ si k<1

$y^*_{t+j} = \hat{y}_t(j)$ si j>0

$e^*_{t+k}(h) = 0$ si k>0

III. 7. Quelques compléments sur le modèle ARIMA

1. On peut montrer que les procédures de prétraitement visant à stationnariser la série introduisent un biais dans la prévision qu'on ne peut pas toujours définir. En effet, les prévisions optimales sont construites sur la série transformée et peuvent perdre leur caractère optimal lorsqu'on les exprime de la même façon que les données initiales [2], [13].

2. La méthodologie de Box et Jenkins ne permet pas toujours d'identifier un seul et unique modèle. On est parfois amené à considérer plusieurs étant données les allures des fonctions *ACF* et P*ACF*.

3. Habituellement, lorsque plusieurs modèles sont candidats, on utilise des critères d'information pour les départager. Les critères les plus usités sont les critères AIC et BIC définis par :

$$AIC(p,q) = \ln(\hat{\sigma}^2) + 2\frac{p+q}{n} \quad \text{(III.62)}$$

$$BIC(p,q) = \ln(\hat{\sigma}^2) + (p+q)\frac{\ln(n)}{n} \quad \text{(III.63)}$$

Où $\hat{\sigma}^2$ est une estimation de la variance des erreurs de prévision. Ces critères établissent un compromis entre l'adéquation du modèle sur les n valeurs $\{y_t\}$ observées dans le passé qui ont servi à estimer le modèle et sa complexité calculée en fonction du nombre de paramètres. Ces critères sont en cela conformes au principe de parcimonie veillant à construire des modelés les moins complexes possible afin d'éviter la redondance.

III.8 Quelques extensions

Dés leurs introductions, les modèles ARIMA ont connus un large succès et la méthodologie de Box et Jenkins à été appliquée dans de nombreux domaines nécessitant l'obtention de prévisions ou la construction de modèles linéaires stochastiques. C'est le cas par exemple en économie ou encore en théorie de contrôle de processus, où des extensions ARMAX ont été proposées par les ingénieurs [8].

Parmi les autres extensions des modèles ARIMA, on notera celle de Box et Tiao. Leur modèle appelé « ARIMA avec intervention » prend en compte des processus avec changement de moyenne ou de tendance. Ces ruptures sont présentées comme la conséquence de chocs aléatoires ayant un effet permanent ou transitoire.

Des extensions multidimensionnelles des modèles ARMA ont aussi été proposées, la plus populaire étant le modèle VAR(p) (Vector Autorégressif).

$$Y_t = \phi_1 Y_{t-1} + ... + \phi_1 Y_{t-1} + E_t \quad \text{(III.64)}$$

ou Y_t et E_t sont des vecteurs aléatoires de dimension k les coefficients ϕ_j sont des matrices carrées d'ordre k. de façon plus générale, un processus VARMA est défini par :

$$\Phi(L)Y_t = \Theta(L)E_t \quad \text{(III.65)}$$

Où E_t est un bruit blanc multivarié et où $\Phi(L)$ et $\Theta(L)$ sont des matrices.

III. 9. Processus stochastiques non linéaire

Une généralisation de la formulation ARIMA à été proposée par Priesley avec le modèle SDM(p,d) est de la forme

$$Y_t + \sum_{i=1}^{p} \phi_i(x_{t-1})Y_{t-i} = \mu(x_{t-1}) + E_t + \sum_{j=1}^{q} \theta_j(x_{t-1})E_{t-j} \quad \text{(III.66)}$$

où x_t représente le vecteur d'état défini par :

$x_t = (E_t, \ldots\ldots E_{t-q+1}, Y_t \ldots\ldots Y_{t-p+1})$.

Le processus $\{E_t\}$est un bruit blanc de variance σ^2 et les paramètres du modèle μ, ϕ_j, et θ_j sont ici des fonctions du vecteur d'état à l'instant précédent.

Cette modélisation est intéressante car elle permet de retrouver bon nombre de modèles stochastiques non linéaires en particularisant les fonctions inconnues μ, ϕ_j, et θ_j . Bien sur lorsque les fonctions inconnues sont des constantes, on retrouve la forme linéaire ARMA.

III. 9. 1. Le modèle bilinéaire BL

Si l'on choisit μ et ϕ_j constant et

$$\theta_j(x_{t-1}) = \theta_j + \sum_{k=1}^{Q} c_{jk} Y_{t-k} \qquad \text{(III.67)}$$

Alors de processus stochastique non linéaire devient

$$Y_t - \sum_{i=1}^{p} \phi_i Y_{t-i} = \mu + E_t - \sum_{j=1}^{q} E_{t-j} - \sum_{j=1}^{q}\sum_{k=1}^{Q} c_{jk} Y_{t-k} E_{t-j} \quad \text{(III.68)}$$

cette équation décrit un modèle bilinéaire particulier BL(p,q,P,Q) ou $P=q$. Dans le cas général, un modèle est linéaire séparément en Y et E, mais n'est pas linéaire conjointement en ces variables étant donné la présence du dernier terme bilinéaire en Y et E.

III. 9. 2. Le modèle Autorégressif exponentiel EAR

En choisissant μ constant, les θ_j nuls et

$$\phi_j(x_{t-1}) = \phi_j + c_i \exp(-\lambda Y_{t-1}^2) \quad \text{(III.69)}$$

On obtient le modèle EAR(p) (*Exponentiel Auto Régressive Model*) :

$$Y_t + \sum_{i=1}^{p} \left\{\phi_i + c_i \exp(-\lambda Y_{t-1}^2)\right\} Y_{t-i} = \mu + E_t \qquad \text{(III.70)}$$

Le modèle EAR(p) a été introduit par Ozaki et Oda, l'estimation des paramètres du modèle se ramenant à un problème d'optimisation non-linéaire complexe, Osaka et Oda ont proposé une alternative pour la phase d'estimation consistant à faire varier p et λ sur une grille et à estimer les autres paramètres par régression linéaire ordinaire. On se reportera à l'article de Saikkonen et Luukkonen pour une critique de cette méthode dans le cadre des méthodes BL.

III. 9. 3. Modèle ARCH et GARCH

Les modèles ARCH (AutoRegressive Conditionnel Heteroskedastic) et GARCH (Generalized ARCH) n'appartiennent pas à la classe des modèles SMD. Ils ont été introduits par Engle et Bollerselev. La non-linéarité de ces modèles provient du fait que la variance conditionnelle du processus {Et} n'est pas constante dans le temps (on parle de modèle à variance stochastique).

On peut incorporer un processus à variance conditionnelle stochastique dans n'importe quel modèle de régression linéaire :

$$E_t = X_t\beta + E_t \qquad \text{(III.71)}$$

Où Xt représente la ligne t de la matrice des données.

Supposons que la distribution conditionnelle de Yt sous *Ft-1* soit normale de moyenne Xt β et de variance ht, où ht dépend du passé comme suit:

$$h_t = a_0 + a_1 E_{t-1}^2 \qquad \text{(III.72)}$$

Ce modèle est appelé modèle de régression linéaire avec erreurs ARCH(1). Dans un modèle ARCH, les erreurs *E*t forment une suit de variables aléatoires non corrélées de moyenne nulle et de variance constante.

Le modèle ARCH(q) est une généralisation de ce qui précède ou cas où:

$$\mathrm{h_t} = \mathrm{a_0} + \mathrm{a_1 E_{t-1}^2} + \ldots \mathrm{a_{t-q} E_{t-q}^2} \qquad \text{(III.73)}$$

Depuis l'introduction des modèles ARCH(p,q), beaucoup d'extensions sont apparues. Les modèles GARCH (p,q) constituent sont doute la généralisation la plus populaire des modèles ARCH(q). on obtient un modèle GARCH on remplaçant (III.73) par :

$$\mathrm{h_t} = \mathrm{a_0} + \sum_{\mathrm{i=1}}^{\mathrm{q}} \mathrm{E_{t-1}^2} + \sum_{\mathrm{i=1}}^{\mathrm{p}} \mathrm{h_{t-i}} \qquad \text{(III.74)}$$

III. 10. Modèles ARMAX

Une spécification importante du modèle ARMA connue sous le nom de modèle ARMAX (AutoRegressive Moving Average eXogenous inputs). A l'aide de ce modèle les économètres peuvent construire des scénarios sur le comportement futur des endogènes à partir de certaines hypothèses sur l'évolution possible des exogènes

$$A(l)Y_t = \sum_{i=1}^{k} B_i(L)X_{ti} + \Theta(L)E_t \qquad \text{(III.75)}$$

Les modèles ARMAX occupent une place prépondérante en théorie du contrôle de processus, notamment en contrôle prédictif. Dans ce contexte, les variables exogènes $\mathrm{X_{ti}}$ sont appelées variables d'entrée ou de commande. La sortie du système $\mathrm{Y_t}$ est influencée par ses valeurs retardées $\mathrm{Y_{t-j}}$(partie AR), les variables d'entrées (partie x du modèle) et des perturbations aléatoires $\mathrm{E_{t-j}}$(partie MA) appelées entrées secondaires ou parasites. En présence de perturbations, on parle de système bruité. Dans le cas d'un ARMAX, puisque le bruit est filtré, on parlera de système à bruit coloré. Nous savons que le fait que la sortie soit influencée par ses propres valeurs retardées $\mathrm{Y_{t-j}}$ crée un système dynamique dans le processus d'adaptation. Les ingénieurs parlent dans ce cas de systèmes en boucle fermée, de systèmes dynamiques ou encore

de systèmes avec (boucle de) rétroaction. Une extension au cas non linéaire a été proposée par Leontaritis et Billings avec le modèle NARMAX.

On notera que plusieurs modelés parmi lesquels, les modèles ARMAX et la famille de modèles de type espace d'états (comprenant les filtres de Kalman comme cas particuliers) ont été initialement proposée par les ingénieurs et sont maintenant intensivement utilisées en économétrie

Techniques de soft computing

IV.1. Introduction

Les techniques de soft computing sont de plus en plus utilisés dans la prédiction, la conception, la modélisation et la commande de systèmes complexes tels que les robots, les procédés biologiques, les véhicules routiers . ... On entend par les techniques de soft computing ou aussi dite outils intelligents: la logique floue, les réseaux de neurones, et les algorithmes génétiques.

- La logique floue introduite par Zadeh (1965) dans les années soixante constitue un outil très puisant pour la représentation des termes et des connaissances vagues. Elle est issue de la capacité de l'homme à décider et à agir d'une manière intelligente malgré l'imprécis et l'incertitude des connaissances disponibles. Son utilisation dans le domaine du contrôle (contrôle flou) a été l'une des premières applications de cette théorie dans l'industrie avec les travaux de Mamdani et Assilian (1975). Depuis, les applications de la logique floue se sont multipliées pour toucher des domaines très divers. L'utilisation de la commande floue est particulièrement intéressante lorsqu'on ne dispose pas de modèle mathématique précis, voir inexistant, du système à commander ou lorsque ce dernier présente de fortes non linéarités. Contrairement aux approches classiques de l'automatique, qui se basent en grande partie sur un modèle mathématique, la commande par logique floue, repose sur une collection de règles linguistiques de la forme " Si · · · Alors " qui traduisent la stratégie de contrôle d'un opérateur humain.
- Les réseaux de neurones sont apparus dans les années cinquante mais n'ont reçu cependant un intérêt considérable qu'à partir des années 80 avec l'apparition de l'algorithme de rétropropagation (Rumelhart et McClelland, 1986). Leur capacité d'apprentissage et d'approximation de fonctions leur procure un intérêt considérable de la part des chercheurs. Il suffit de voir les nombreuses applications industrielles qui en découlent à partir des années 90 et de consulter l'abondante littérature sur le sujet pour s'en convaincre.
- 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 (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.

Au cours de ces dernières années, la combinaison de ces techniques a attiré l'attention de beaucoup de chercheurs. Plusieurs hybridations ont été alors proposées dont les plus rencontrées sont: Algorithme Génétique /Contrôleur flou (AG/CF) et Réseau de neurones/

Contrôleur flou (RN/CF).

IV.2. Logique floue

IV.2.1 Définition

La logique floue est une technique utilisée en intelligence artificielle. Formalisée par Lotfi Zadeh, elle a été utilisée dans des domaines aussi varies que la robotique, la gestion de la circulation routière, le contrôle aérien, l'environnement (météorologie, climatologie, sismologie, analyse du cycle de vie), la médecine (aide au diagnostic) etc.…
Elle s'appuie sur la théorie mathématique des ensembles flous. Cette théorie, introduite par Zadeh, est une extension de la théorie des ensembles classiques pour la prise en compte d'ensembles définis de façon imprécise. C'est une théorie formelle et mathématique dans le sens ou Zadeh, en partant du concept de fonction d'appartenance pour modéliser la définition d'un sous-ensemble d'un univers donne, a élabore un modèle complet de propriétés et de définitions formelles. Il a aussi montre que cette théorie des sous-ensembles flous se réduit effectivement a la théorie des sous-ensembles classiques dans le cas ou les fonctions d'appartenance considérées prennent des valeurs binaires {0,1}.
Les sous-ensembles flous ont été introduits donc pour modéliser la représentation humaine des connaissances, et ainsi améliorer les performances des systèmes de décision qui utilisent cette modélisation. Ils sont utilises soit pour modéliser l'incertitude et l'imprécision, soit pour représenter des informations précises sous forme lexicale assimilable par un système expert.

IV.2.2 Systèmes flous / Commande floue

En automatique, la majorité des approches de la commande non linéaire exige la disponibilité d'un modèle mathématique du système et ceci n'est pas toujours réalisable a cause de l'imprécision et l'incertitude liées aux paramètres mal connus, difficilement identifiables et/ou des dynamiques négligées [21], [25]. D'autre part, les méthodes de modélisation traditionnelles s'avèrent souvent incapables de refléter le comportement global d'un système donne. L'utilisation des contrôleurs bases sur l'expertise humaine peut être une alternative a la commande de ce type de systèmes. Ils présentent l'avantage de tolérer l'incertitude du modèle et compensent son effet. Parmi ces approches, nous distinguons celles utilisant la logique floue (type-1 et type-2) [16], [20]. Les incertitudes, les non linéarités négligées et les différentes contraintes peuvent être ainsi compensées [22], [23], [17]. Ces contrôleurs ont connu beaucoup de succès et sont devenus un sujet principal dans le domaine de la recherche des systèmes intelligents

IV.2.3 Principe

La logique floue est née de la constatation que la plupart des phénomènes ne peuvent pas être représentée à l'aide de variables booléennes qui ne peuvent prendre que deux valeurs (0 ,1). Peut on considérer un eau à 18C comment étant chaude ou froide? N'est-elle pas ni vraiment chaude, ni vraiment froide mais tout simplement tiède.

Pour répondre à ce type de question, la logique floue considère la notion d'appartenance d'un objet à un ensemble non plus comme une fonction booléenne mais comme une fonction qui peut prendre toutes valeurs entre 0 et 1.

IV.2.4 Ensemble flou

Dans la théorie des ensembles, un élément appartient ou n'appartient pas à un ensemble. La notion d'ensemble est à l'origine de nombreuses théories mathématiques. Cette notion

essentielle ne permet cependant pas de rendre comptez de situations pourtant simples est rencontrées fréquemment. Parmi des fruits, il est facile de définir l'ensemble des pommes. Par contre, il est sera plus difficile de définir l'ensemble des pommes mûres. On conçoit bien que la pomme mûrit progressivement la notion de pomme mure est donc graduelle. C'est pour prendre en compte de telles situations qu'a été crée la notion d'ensemble flou. La théorie des ensembles flous repose sur la notion d'appartenance partielle : chaque élément appartient partiellement ou graduellement aux ensembles flous qui ont été définis. [16]

Mathématiquement, Un ensemble flou A dans un univers de discours X est défini par une fonction d'appartenance qui associe à chaque élément x de X, le degré $\mu_A(x)$, compris entre 0 et 1 . Donc un ensemble flou peut être représenté par un ensemble de paires ordonnées :

$$A = \{(x, \mu_A(x)) / x \in X\} \quad \text{(IV.1)}$$

Tel que : $\mu_A(x)$ La fonction d'appartenance d'un variable x.

X : est appelé l'univers de discours il peut contenir des valeurs continues ou discrètes.

Notation:

Si A est discret : $A = \sum \mu_A(x)/x$ (IV.2)

Si A est continu : $A = \int \mu_A(x)/x$ (IV.3)

Exemples : Evaluation de la température d'un corps (figure IV.1)

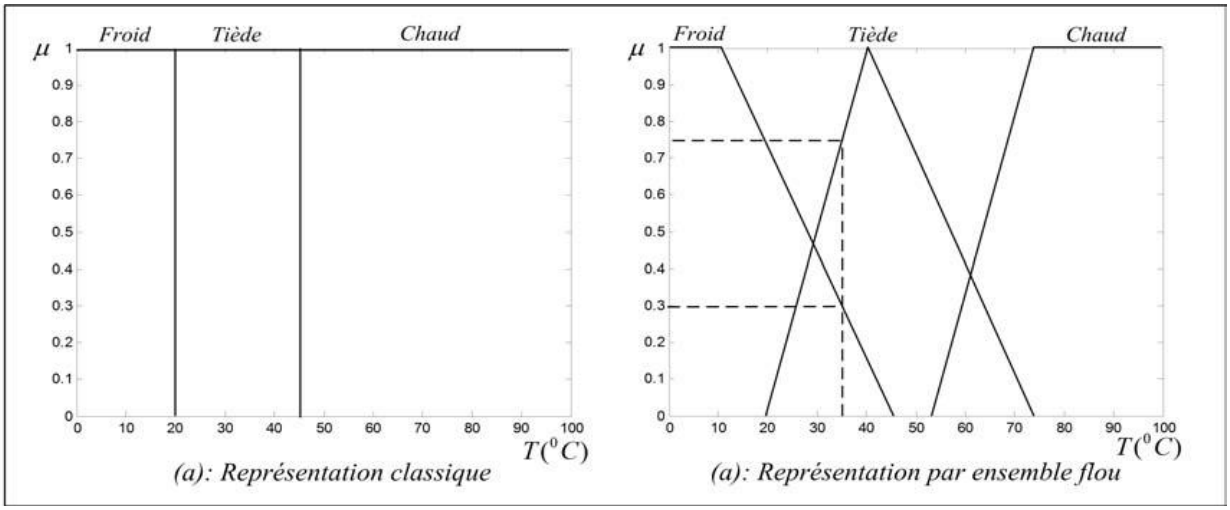


Figure IV.1. Représentation de la température d'un corps par les ensembles classiques et flous

- En logique classique (Fig IV.1.a), le degré d'appartenance μ ne peut prendre que deux valeurs (0 ou 1). Dans ce cas le corps peut être :
  - Froid: $\mu_{froid} = 1, \mu_{Tiede} = 0, \mu_{Chaud} = 0$
  - Tiède: $\mu_{froid} = 0, \mu_{Tiede} = 1, \mu_{Chaud} = 0$
  - Chaud: $\mu_{froid} = 0, \mu_{Tiede} = 0, \mu_{Chaud} = 1$

La température du corps ne peut pas prendre deux qualificatifs à la fois.

- En logique floue, le degré d'appartenance devient une fonction qui peut prendre une valeur réelle intermédiaire comprise entre 0 et 1 inclus. Dans ce cas, pour le qualificatif tiède, le corps peut être considéré à la fois, comme froid avec un degré d'appartenance de 0.3 et comme tiède avec un degré d'appartenance de 0.75 (figure IV.1.b).

Pour T=35 0C : $\mu_{froid}(T) = 0.3, \mu_{Tiede}(T) = 0.75, \mu_{Chaud}(T) = 0$

IV.2.5 Opérateurs logiques flous

Dans la théorie des ensembles classiques, on utilise différentes opérations tel que compliment, union, intersection. Les mêmes opérations sont également définies dans la théorie des ensembles flous. Comme donc, la valeur de la fonction d'appartenance ne sont pas limiter au valeur {0,1} mais peut prendre n'importe valeur de l'intervalle [0,1]. Ces opérations ne peuvent pas être définies de la même manière que celles dans les ensembles classiques. Il existe de nombreuses variantes dans ces opérateurs. Cependant, les plus répandus sont ceux dit de «Zadah » décrits ci-dessous.

Soient A, B et C des ensembles flous dans U décrits par leurs fonctions d'appartenances $\mu_A(u)$, $\mu_B(u)$, $\mu_C(u)$ respectivement.

- L'union de A et B, noté par A ∪B, est définie par
$$\mu_{A\cup B}(u) = \max[\mu_A(u), \mu_B(u)] \quad \text{(IV.4)}$$

- L'intersection de A et B, noté par A∩B, est définie par
$$\mu_{A\cap B}(u) = \min[\mu_A(u), \mu_B(u)] \quad \text{(IV.5)}$$
- Le complément de A, noté par Ā, définies par:
$$\mu_{\bar{A}}(u) = 1 - \mu_A(u) \quad \text{(IV.6)}$$

- Le produit cartésien de A, B, et C, noté AxBxC, et définie par :
$$\mu_{AxBxC}(u_1, u_2, u_3) = \min[\mu_A(u), \mu_B(u), \mu_C(u)] \quad \text{(IV.7)}$$

Les opérations *minimum*, *maximum* et *complémentation à 1* ont été choisies pour définir respectivement *l'intersection, l'union* et le *complément d'ensembles flous* Parce qu'ils préservent presque toute la structure de la théorie classique des ensembles [18].

IV.2.6 Règles floues

Une base des règles floues est composée de règles qui sont généralement utilisées parallèle. Une règle est de type : **SI « prédicat » ALORS « conclusion »**

Exemple : **SI** la température élevée et la pression forte **ALORS** ventilation forte et soupape grande ouverte.

Prédicat : encore appelé prémisse ou condition est une combinaison de propositions par des opérateurs ET, OU, NON. Les propositions « température élevée » et « pression forte » sont combinées par un opérateur ET pour former le prédicat de règle.
Conclusion : la conclusion d'une règle floue est une combinaison de propositions liées par un opérateur ET.

IV.2.6. Structure générale d'une commande floue

De manière classique, le fonctionnement interne d'un système flou repose sur la structure présentée par la figure IV.2 qui inclut quatre blocs:

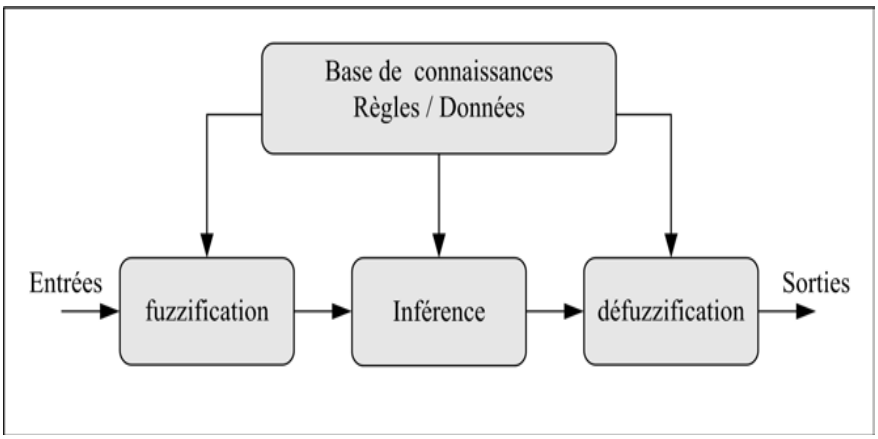


Figure IV.2 : Structure d'un système flou.

1. Base de connaissances floues

Elle est constituée d'une *base de données* et une *base de règles.* La base de données contient les fonctions d'appartenance des ensembles flous, et la base de règles est une collection de règles floues **IF-THEN** qui définissent la relation entre une antécédente qui représentent la description de l'état de système et La conséquente qui exprime l'action de l'opérateur qui contrôle le système.

2. Fuzzification

L'entrée *x* varie dans un domaine appelé univers de discours *X,* divise en un nombre fini d'ensembles flous de telle sorte que dans chaque zone il y a une situation dominante. Afin de faciliter le traitement numérique et l'utilisation de ces ensembles, on les décrit par les fonctions d'appartenance. Elles admettent comme argument la position de *x* dans l'univers de discours, et comme sortie le degré d'appartenance de *x* a la situation décrite par la fonction. La fuzzification proprement dite consiste à définir des fonctions d'appartenances pour les différentes variables linguistiques. Le but est la conversion d'une grandeur physique en une linguistique. Il s'agit d'une projection de la variable physique sur les ensembles flous caractérisant cette variable. Cette opération permet d'avoir une mesure précise sur le degré d'appartenance de la variable d'entrée à chaque ensemble flou.

3. Mécanisme d'inférence:

Consiste d'une part à calculer le degré de vérité des différentes règles du système et d'autre part à associer à chacune de ces règles une valeur de sortie. Cette valeur de sortie dépend de la partie conclusion des règles qui peut prendre plusieurs formes. Il peut s'agir d'une proposition floue, et l'on parlera dans ce cas de règle de type Mamdani:

$Si(\ldots\ldots\ldots.)$ *Alors* Y *est* B,

B ensemble flou

Il peut également s'agir d'une fonction réelle des entrées, et l'on parlera dans ce cas de règle de type Sugeno:

$Si(\ldots\ldots\ldots.)$ *Alors* $Y = f(x_1, x_2, \ldots\ldots, x_n)$

$x_1, x_2, \ldots\ldots, x_n$ sont les valeurs réelles des variables d'entrées.

Dans ce dernier cas, la valeur de sortie de la règle est tout simplement donnée par: $\omega \times f(x_1, x_2, ..., x_n)$. ω Représente le degré de vérité de la règle. Dans le cas d'une règle de type Mamdani, la sortie est un sous ensemble flou obtenu à partir de celui présent dans la conclusion de la règle [19].

4. Déffuzification

Consiste à remplacer l'ensemble des valeurs de sorties des différentes règles résultant de l'inférence par une valeur numérique unique représentative de cet ensemble. Dans le cas des règles de type Sugeno, le calcul se fait simplement par une somme normalisée des valeurs associées aux règles floues. Dans le cas de règles de Mamdani, le calcul de la valeur numérique de sortie s'effectue en deux étapes:

a .Composition des règles : Une fois la phase d'inférence terminée, il s'agit de regrouper (par union) les sous ensemble flous issus de l'inférence pour en obtenir un seul ensemble représentatif des différentes conclusions des règles floues. Comme méthode de composition, on peut citer en particulier les compositions MAXIMUM (en général couplée avec l'inférence MINIMUM) et SOMME (en général couplée avec l'inférence PRODUIT).La première consiste à caractériser l'ensemble de sorties par une fonction d'appartenance égale au maximum des fonctions d'appartenance des sous ensembles flous. La deuxième consiste à faire la somme de fonctions d'appartenance des sous ensembles issus de l'inférence [20].

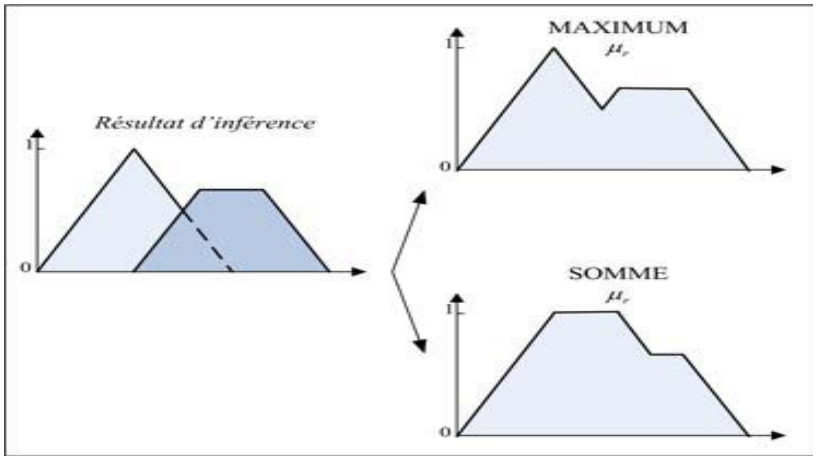


Figure IV.3 Composions des ensembles flous issus de l'inférence

b. Déffuzzifucation : C'est la phase de défuzzification proprement dite qui permet de générer une valeur numérique à partir de l'ensemble obtenu par composition des règles. Il existe plusieurs méthode de défuzzification les plus utilisées sont:

- méthode du centre de gravite (cog): La méthode du centre de gravité est une des méthodes les plus mentionnées dans la littérature. L'abscisse du centre de gravité peut être déterminée en utilisant la formule générale :

$$y = \frac{\sum_{l=1}^{M} v^l \mu_{B^l}(v^l)}{\sum_{l=1}^{M} \frac{\mu_{B^l}(v^l)}{\sigma^{l^2}}} \qquad \text{(IV.8)}$$

Où v^l désigne le centre de gravité de la fonction d'appartenance de l'ensemble flou B^l et de defuzzificateur évalue premièrement $\mu_{B^l}(v^l)$ et σ^l est une mesure de support de la fonction d'appartenance pour la l^{ieme} règle. Pour les fonctions d'appartenance triangulaires et trapézoïdales, σ^l représente la base de triangle ou de trapèze. Tandis que, pour les fonctions d'appartenance gaussienne σ^l est l'écart type [21].

- La méthode de maximum : Cette méthode, s'applique uniquement dans le cas où la fonction d'appartenance associée à l'ensemble de sortie n'admet qu'un seul maximum. On choisit comme sortie l'abscisse y^* correspondant à ce maximum.
- méthode de moyenne maximum (mom)**:** Cette méthode génère une commande précise en calculant la moyenne des valeurs pour les quelles l'appartenance est maximale.

$$y = \sum_{i=1}^{N} \frac{v_i}{N} \qquad \text{(IV.9)}$$

Où les v_i sont les valeurs de v pour le quelle μ_B atteint le maximum, et N est le nombre de telles valeurs.

IV.2.7. Etapes de conception des systèmes flous

IV.2.7.1. Variables d'entrées et sorties

Le premier pas dans la conception d'un contrôleur flou consiste : à définir les entrées et des sorties de contrôleur. Généralement, dans le cas d'un prédicteur flou, les variables d'entrée sont les valeurs de série à prédire retardées.

IV.2.7.2 Choix de la partition floue

Après le choix des variables d'entrée et de sortie, on associe à chaque variable un nombre de termes linguistiques, chaque terme est défini par une fonction d'appartenance. Généralement, ce nombre est impair entre 3 et 9.

IV.2.7.3. Choix de la forme des fonctions d'appartenances

En général, les fonctions d'appartenances de forme triangulaire, trapézoïdal, gaussienne sont les plus utilisées à cause de l'efficacité et la facilité d'implantation. Chaque fonction d'appartenance d'un ensemble floue doit avoir un intervalle de chevauchement avec ses voisinages avec un pourcentage de 10 à 50 du support de la fonction voisine [21].

IV.2.7.4. Choix des règles floues

Le choix des règles floues semble facile, par contre il est un des problèmes posés dans l'implémentation des algorithmes flous, des nombreuses méthodes d'extraction des règles ont été proposées pour résoudre ce problème, ses méthodes sont groupées selon leur principe en deux catégories :

a. Méthodes d'extraction naturelle

a .1. En se référant à un savoir-faire d'un expert : Cette méthode concerne essentiellement les applications de contrôle où les règles sont gagnées auprès d'un expert ou un opérateur qualifié. L'enquête peut être effectuée, soit par des interviews ou par des interrogations utilisant des questionnaires soigneusement élaborés. Le problème avec cette méthode, est que l'expert ou l'opérateur ne peuvent pas toujours exprimer les règles qui leurs permettent naturellement du formuler ou déterminer une action.

a.2 En modélisant les actions d'un opérateur : En utilisant cette stratégie, les règles floues sont érigées en observant l'opérateur en train de manipuler le système, éventuellement on peut lui poser des questions sur, par exemple, ces actions ou le type d'informations qu'il les utilise dans sa décision.

a .3 Utilisant le modèle flou d'un processus : On entend par le modèle flou d'un processus, la description linguistique des caractéristiques dynamiques du processus. En se basant sur ce modèle, deux méthodes peuvent être utilisées pour créer une base de règle. La première est une méthode heuristique dans laquelle les règles floues sont établies pour compenser les indésirables du système. La seconde détermine la structure et les paramètres des règles floues de telle sorte que le système flou et le modèle flou satisfait les exigences spécifiées.

b. Méthode d'extraction automatique

Plutôt que de compter sur un expert pour fournir un jeu de règles et de s'engager ensuite dans un long processus d'ajustement de ces règles ainsi que les fonctions d'appartenance, plusieurs méthodes auto-adaptatives ont été proposées récemment pour automatiser cette procédure. Ces méthodes sont très utiles dans le cas où aucun expert n'est disponible' et où l'on dispose suffisamment de grandes quantités de données concernant le problème à traiter.

Le premier système auto-adaptatif a été proposé par *Mandani* et *Procyk* qui l'on appelé Self-Organizing Controller (SOC). Ce SOC possède une structure hiérarchique constituée de deux bases de règles : la première est une base de règles générales du système flou, et la deuxième est une base générale de règles en se basant sur les performances désirées du système [25].

L'approche neuronale a été aussi utilisée pour raffiner les règles du système flou en les traitants comme synapses. L'utilisateur fournit le premier jeu de règles, que le réseau perfectionne sur des certaines de millier de cycles, modifiant chaque fois légèrement les fonctions d'appartenance pour minimiser l'erreur à la sortie [25].

En s'inspirant du mécanisme d'évolution biologique, les algorithmes génétiques ont prouvé leur capacité d'optimiser la base des règles et même les fonctions d'appartenance.

IV.3. Réseaux de Neurones

IV.3.1 Principe du neurone artificiel

Chaque neurone artificiel est un processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones en amont ou des capteurs composant la machine dont il fait partie. A chacune de ses entrées est associé un poids représentatif de la force de la connexion. Chaque processeur élémentaire est doté d'une sortie unique, qui se ramifie ensuite pour alimenter un nombre variable de neurones en aval. A chaque connexion est associé un poids.

La première version de ce dernier est celle de Mc Culloch et W. Pitts et date de 1943. S'inspirant de leurs travaux sur les neurones biologiques, ils ont proposé le modèle du neurone formel qui se voit comme un opérateur effectuant une somme pondérée de ses entrées suivie d'une fonction d'activation (ou de transfert) comme indiqué par la figure IV.3.

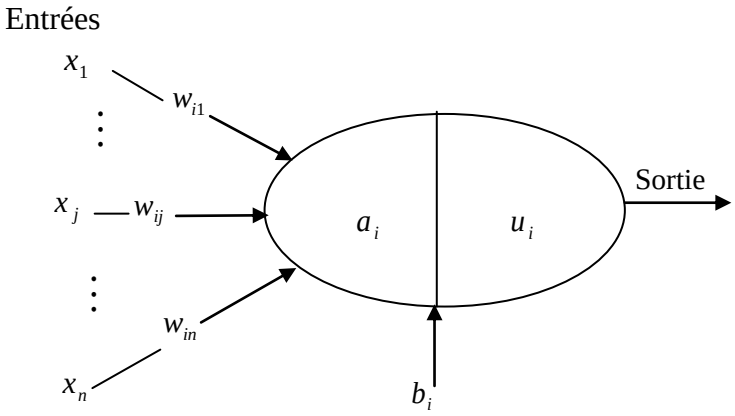


Figure. IV.3 : Modèle de base d'un neurone formel

a_i : représente la somme pondérée des entrées du neurone, elle est donnée par :

$$a_i = \sum_j w_{ij}.x_j + b_i \quad \text{(IV.10)}$$

x_j : représente l'entrée j connectée au neurone i. b_i : le seuil interne du neurone. w_{ij} : désigne le poids de la connexion reliant l'entrée j au neurone i.

$u_i = f(a_i)$ Est la sortie du neurone et f sa fonction d'activation.

La fonction d'activation de chaque neurone détermine ses propres caractéristiques. Par conséquent, le type du neurone est caractérisé par sa fonction d'activation. Conformément au neurone biologique, les fonctions d'activation sont généralement croissantes et continues. Les fonctions les plus utilisées sont la fonction linéaire et la fonction sigmoïde. Leur choix revêt une importance capitale et dépend souvent du type de l'application et du domaine de variation des variables d'entrée/sortie [21].

IV.3.2. Modélisation des réseaux de neurones artificiels

Un réseau de neurones peut être considéré comme un modèle mathématique de traitement réparti, composé de plusieurs éléments de calcul non linéaire (neurones), opérant en parallèle et connectés entre eux par des poids. Les réseaux de neurones artificiels sont des réseaux

fortement connectés de processeurs élémentaires fonctionnant en parallèle. Chaque processeur élémentaire calcule une sortie unique sur la base des informations qu'il reçoit. Les neurones artificiels sont souvent utilisés sous forme de réseaux qui diffèrent selon le type de connections entre les neurones, une cinquantaine de types peut être dénombrée. En guise d'exemples nous citons : le perceptron de Rosemblat, les réseaux de Hopfield etc.…..
Ces derniers sont les plus utilisés dans le domaine de la modélisation et de la commande des procédés. Ils sont constitués d'un nombre fini de neurones qui sont arrangés sous forme de couches. Les neurones de deux couches adjacentes sont interconnectés par des poids. L'information dans le réseau se propage d'une couche à l'autre, on dit qu'ils sont de type « feed-forward ». Nous distinguons trois types de couches :

Couche d'entrée : les neurones de cette couche reçoivent les valeurs d'entrée du réseau et les transmettent aux neurones cachés. Chaque neurone reçoit une valeur, il ne fait pas donc de sommation.

Couches cachées : chaque neurone de cette couche reçoit l'information de plusieurs couches précédentes, effectue la sommation pondérée par les poids, puis la transforme selon sa fonction d'activation qui est en général une fonction sigmoïde. Par la suite, il envoie cette réponse aux neurones de la couche suivante.

Couche de sortie : elle joue le même rôle que les couches cachées, la seule différence entre ces deux types de couches est que la sortie des neurones de la couche de sortie n'est liée à aucun autre neurone.

IV.3.3. Architecture des réseaux de neurones

On distingue deux structures de réseau, en fonction du graphe de leurs connexions, c'est-à-dire du graphe dont les nœuds sont les neurones et les arêtes les «connexions» entre ceux-ci :

- Les réseaux de neurones statiques (ou acycliques, ou non bouclés).
- Les réseaux de neurones dynamiques (ou récurrents, ou bouclés).

IV.3.3.1. Réseaux de neurones non boulées

Dans ce type de structure dite 'feedforward', la propagation de l'information se fait uniquement de l'entrée vers la sortie. Les neurones de la même couche peuvent se connecter uniquement avec les neurones de la couche suivante. L'architecture la plus utilisée est le perceptron multicouche. Les neurones composant ce réseau s'organisent en N couches successives $(N \geq 3)$.
Il existe deux types de réseaux de neurones : les réseaux complètement connectés et les réseaux à couche. Le réseau perceptron multicouche est un cas particulier de ce dernier type.

1. Réseaux de neurones complètement connectés : Dans un réseau complètement connecté, les entrées puis les neurones (cachés et de sortie) sont numérotés et pour chaque neurone ses entrées sont toutes les entrées du réseau ainsi que les sorties des neurones de numéro inférieur et sa sortie est connectée aux entrées de tous les neurones de numéro supérieur.
2 . R***éseaux de neurones à couches :*** Dans une architecture de réseaux à couches, les neurones cachés sont organisés en couches, les neurones d'une même couche n'étant pas connectés entre eux. De plus les connexions entre deux couches de neurones non

consécutives sont éliminées. Une telle architecture est historiquement très utilisée, surtout en raison de sa pertinence en classification.

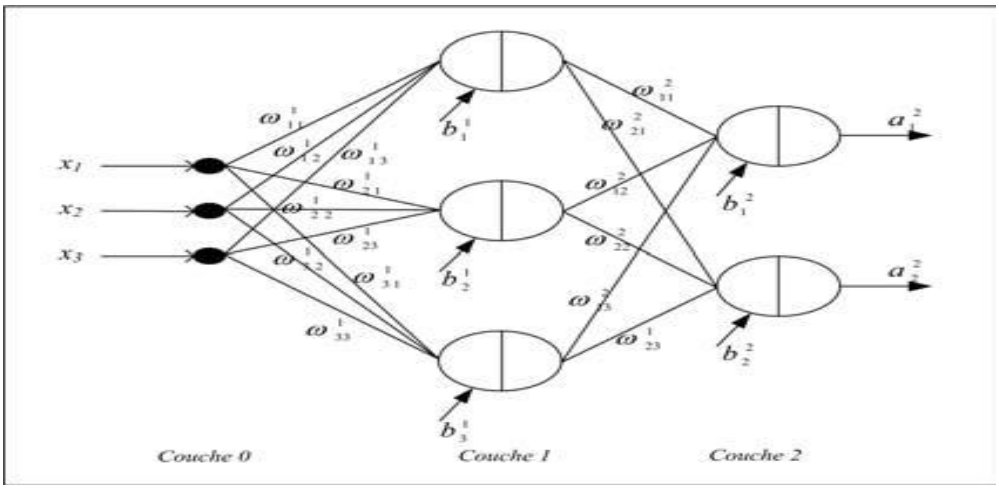


Figure. IV.4. Perceptron à une couche cachée.

Dans l'exemple suivant (figure IV.4), nous présentons un perceptron à trois couches. Les neurones de la première couche, nommée couche d'entrée, envoient leur activation forcée à la valeur d'entrée. La dernière couche est appelée couche de sortie. Elle regroupe les neurones dont les fonctions d'activation sont généralement de type linéaire. Les couches intermédiaires sont appelées couches cachées. Elles constituent le cœur du réseau.

Sur la figure IV.4, les termes b_i^l et w_{ij}^l désignent respectivement le biais du neurone i de la couche l et le poids de connexion entre le neurone j de la couche l-1 et le neurone i de la couche l. Tenant compte de ces notations, la sortie du neurone i dans la couche l est peut être donnée par :

$$a_i^l = \sum_j^{N_{l-1}} w_{ij}^l \times u_j^{l-1} + b_i^l \quad \text{(IV.11)}$$

$$u_i^l = f^l\left(a_i^l\right) \quad \text{(IV.12)}$$

$f^l(.)$ est la fonction d'activation des neurones de la couche l.

On peut réécrire les équations ci-dessus sous forme matricielle comme suit :

$$a^l = W^l \times u^{l-1} + \underline{b^l} \quad \text{(IV.13)}$$

$$\underline{u^l} = \underline{f^l}\left(\underline{a^l}\right) \quad \text{(IV.14)}$$

Avec $\underline{a^l} = \left(a_1^l, a_2^l, \ldots, a_{N_l}^l\right)^T$, $\underline{u^l} = \left(u_1^l, u_2^l, \ldots, u_{N_l}^l\right)^T$, $\underline{b^l} = \left(b_1^l, b_2^l, \ldots, b_{N_l}^l\right)^T$ et

$$W^l = \begin{pmatrix} w_{11}^l & w_{12}^l & \cdots & w_{1N_{l-1}}^l \\ w_{21}^l & w_{22}^l & \cdots & w_{N_l N_{l-1}}^l \\ \vdots & \vdots & \ddots & \cdots \\ w_{N_l 1}^l & w_{N_l 2}^l & \cdots & w_{N_l N_{l-1}}^l \end{pmatrix}$$

Le perceptron multicouche présente une alternative prometteuse pour la modélisation des systèmes complexes. Avec une seule couche cachée, il constitue un approximateur universel [22 ,23].

IV.3.3.2. Réseaux de neurones bouclés

Un réseau dynamique ou récurrent possède la même structure qu'un réseau multicouche muni de rétroactions. Les connexions rétroactives peuvent exister entre tous les neurones du réseau sans distinction, ou seulement entre certains neurones (les neurones de la couche de sortie et les neurones de la couche d'entrée ou les neurones de la même couche par exemple). La figure IV.3 montre deux exemples de réseaux récurrents. Le premier est un simple multicouche qui utilise un vecteur d'entrée qui contient les copies des activations de la couche de sortie du réseau et le deuxième est un réseau à mémoire se distingue du premier par la présence des unités mémoires [24].

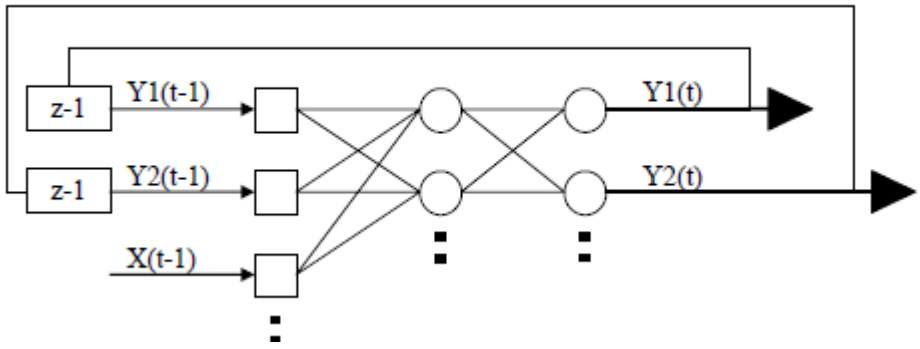


Figure. IV.5. Réseau de neurone bouclé

IV.3.4. L'apprentissage

L'apprentissage d'un réseau de neurones peut être défini comme la phase durant laquelle les divers paramètres le caractérisant sont remis à jour jusqu'à ce qu'ils permettent au réseau d'approximer au mieux la fonction qu'il a à réaliser. Selon l'application dans laquelle le réseau va être intégré, la fonction à approcher peut être connue ou inconnue analytiquement. En effet, le phénomène physique, chimique, économique, biologique, etc. qui nécessite l'emploi d'un réseau de neurones peut être parfaitement modélisé par l'intermédiaire d'équations analytiques ou simplement observables, mettant en jeu un nombre fini de valeurs expérimentales de sorties fonction de valeurs d'entrées bien choisies, sans que les relations qui les relient ne soient connues [25].

Dans la majorité des algorithmes actuels, les variables modifiées pendant l'apprentissage sont les poids des connexions. L'apprentissage est la modification des poids du réseau dans l'optique d'accorder la réponse du réseau aux exemples et à l'expérience. Les poids sont initialisés avec des valeurs aléatoires. Puis des exemples expérimentaux représentatifs du fonctionnement du procédé dans un domaine donné, sont présentés au réseau de neurones. Ces exemples sont constitués de couples expérimentaux de vecteurs d'entrée et de sortie. Une méthode d'optimisation modifie les poids au fur et à mesure des itérations pendant lesquelles on présente la totalité des exemples, afin de minimiser l'écart entre les sorties calculées et les sorties expérimentales. Afin d'éviter les problèmes de surapprentissage, la base d'exemples est divisée en deux parties : la base d'apprentissage et la base de test. L'optimisation des poids se fait sur la base d'apprentissage, mais les poids retenus sont ceux pour lesquels l'erreur obtenue sur la base de test est la plus faible. En effet, si les poids sont optimisés sur tous les exemples de l'apprentissage, on obtient une précision très satisfaisante sur ces

exemples mais on risque de ne pas pouvoir généraliser le modèle à des données nouvelles. A partir d'un certain nombre d'itérations, le réseau ne cherche plus l'allure générale de la relation entre les entrées et les sorties du système, mais s'approche trop prés des points et « apprend » le bruit [25].

Surapprentissage : Il arrive qu'à faire apprendre un réseau de neurones toujours sur le même échantillon, celui-ci devient inapte à reconnaître autre chose que les éléments présents dans l'échantillon. Le réseau ne cherche plus l'allure générale de la relation entre les entrées et les sorties du système, mais cherche à reproduire les allures de l'échantillon. On parle alors de surapprentissage : le réseau est devenu trop spécialisé et ne généralise plus correctement.

IV.3.4.1. Type d'apprentissage

Il existe de nombreux types de règles d'apprentissage qui peuvent être regroupées en trois catégories : les règles d'apprentissage non supervisé, renforcé et supervisé. Mais l'objectif fondamental de l'apprentissage reste le même : soit la classification, l'approximation de fonction ou encore la prévision.

- **Apprentissage non-supervisé :** Ce type d'apprentissage est choisi lorsqu'il n'y pas de connaissances à priori des sorties désirés pour des entrées données. En fait, le réseau est laissé libre de converger vers n'importe quel état final lorsqu'on lui présente des entrées.

- **Apprentissage par renforcement :** L'apprentissage renforcé est une technique similaire à l'apprentissage supervisé à la différence qu'au lieu de fournir des résultats désirés au réseau, on lui accorde plutôt un grade (ou score) qui est une mesure du degré de performance du réseau après quelques itérations. Les algorithmes utilisant la procédure d'apprentissage renforcé sont surtout utilisés dans le domaine des systèmes de contrôle.

- **Apprentissage supervisé** : Un apprentissage est dit supervisé lorsque l'on force le réseau à converger vers un état final précis, en même temps qu'on lui présente un motif. Ce genre d'apprentissage est réalisé à l'aide d'une base d'apprentissage, constituée de plusieurs exemples de type entrées-sorties (les entrées du réseau et les sorties désirées ou encore les solutions souhaitées pour l'ensemble des sorties du réseau). La procédure usuelle dans le cadre de la prévision est l'apprentissage supervisé (ou à partir d'exemples) qui consiste à associer une réponse spécifique désirée à chaque signal d'entrée. La modification des poids s'effectue progressivement jusqu'à ce que l'erreur (ou l'écart) entre les sorties du réseau (ou résultats calculés) et les résultats désirés soient minimisés.

IV.3.4.2. Algorithme d'apprentissage

L'algorithme d'apprentissage est la méthode mathématique qui va modifier les poids de connexions afin de converger vers une solution qui permettra au réseau d'accomplir la tâche désirée. L'apprentissage est une méthode d'identification paramétrique qui permet d'optimiser les valeurs des poids du réseau. Les algorithmes les plus utilisées pour l'apprentissage sont :

- *Loi de Hebb* : Cette règle très simple émet l'hypothèse que lorsqu'un neurone « A » est excité par un neurone « B » de façon répétitive ou persistante, l'efficacité (ou le poids) de l'axone reliant ces deux neurones devrait alors être augmentée sinon il nécessiterait être affaiblie.

- *Loi de Hopfield* : Cette loi se base sur la même hypothèse que la loi de Hebb mais ajoute une variable supplémentaire pour contrôler le taux de variation du poids entre les neurones avec une constante d'apprentissage qui assure à la fois la vitesse de convergence et la stabilité du RNA.

- *Loi Delta* : Cette loi est aussi une version modifiée de la loi de Hebb. Les poids des liens entre les neurones sont continuellement modifiés de façon à réduire la différence (le delta) entre la sortie désirée et la valeur calculée de la sortie du neurone. Les poids sont modifiés de façon à minimiser l'erreur quadratique à la sortie du RNA. L'erreur est alors propagée des neurones de sortie vers les neurones des couches inférieures, une couche à la fois.

- *Règle de rétropropagation* **:** L'algorithme de rétropropagation (backpropagation) est l'un des algorithmes supervisé les plus utilisés pour l'apprentissage des réseaux de neurones. C'est d'ailleurs à sa découverte au début des années 80 (Rumelhart et McClelland, 1986) que l'on doit le renouveau d'intérêt pour les réseaux de neurones. L'objectif de cet algorithme est de modifier les poids du réseau dans le sens contraire du gradient du critère de performance. Dans ce qui suit, nous allons présenter les équations constituant l'algorithme en utilisant un réseau multicouches. Une mise sous forme matricielle sera aussi faite afin de faciliter l'implantation de l'algorithme sous un logiciel bien adapté aux calculs matriciels. Considérons le réseau multicouche décrit précédemment. Pour alléger l'exposé, on suppose que l'apprentissage se fait à chaque présentation d'un couple entrée/sortie de l'ensemble d'apprentissage. Le critère de performance à minimiser peut être alors exprimé

$$J(t)=\frac{1}{2}\sum_{i=1}^{N_L}\left(u_i^L(t)-d_i(t)\right)^2 \quad \text{(IV.15)}$$

Avec $J(t)$ est la valeur du critère à l'instant t.

$d_i(t)$ est la ième sortie désirée à l'instant t.

Les paramètres du réseau sont modifiés suivant la règle du gradient comme suit:

$$w_{ij}^l(t+1)=w_{ij}^l(t)-\eta\frac{\partial J(t)}{\partial w_{ij}^l(t)} \quad \text{(IV.16)}$$

$$b_i^l(t+1)=b_i^l(t)-\eta\frac{\partial J(t)}{\partial b_i^l(t)} \quad \text{(IV.17)}$$

avec η est une constante positive appelée taux d'apprentissage.

Le calcul des quantités $\frac{\partial J}{\partial w}$ et $\frac{\partial J}{\partial b}$ fait intervenir les décompositions ci-dessous:

$$\frac{\partial J(t)}{\partial w_{ij}^l(t)}=\frac{\partial J(t)}{\partial a_i^l(t)}\times\frac{\partial a_i^l(t)}{\partial w_{ij}^l(t)} \quad \text{(IV.18)}$$

$$\frac{\partial J(t)}{\partial b_i^l(t)}=\frac{\partial J(t)}{\partial a_i^l(t)}\times\frac{\partial a_i^l(t)}{\partial b_i^l(t)} \quad \text{(IV.19)}$$

De l'équation (IV.11) on déduit que :

$$\frac{\partial a_i^l(t)}{\partial w_{ij}^l(t)}=u_j^{l-1} \quad \text{(IV.20)}$$

$$\frac{\partial a_i^l(t)}{\partial b_i^l(t)}=1 \quad \text{(IV.21)}$$

En posant, $\delta_i^l(t) = \frac{\partial J(t)}{\partial a_i^l(t)}$ on obtient

$$\frac{\partial J(t)}{\partial w_{ij}^l(t)} = \delta_i^l(t) \times u_j^{l-1} \quad \text{(IV.22)}$$

$$\frac{\partial J(t)}{\partial b_i^l(t)} = \delta_i^l(t) \quad \text{(IV.23)}$$

La quantité $\delta_i^l(t)$ exprime la sensibilité du critère de performance aux changements du potentiel a_i^l du neurone i de la couche l. Dans le cas où i est l'indice d'un neurone de sortie (l = L), on obtient :

$$\delta_i^L(t) = \frac{\partial J(t)}{\partial a_i^L(t)} = \frac{\partial J(t)}{\partial u_i^L(t)} \times \frac{\partial u_i^L(t)}{\partial a_i^L(t)} = \left(u_i^L(t) - d_i(t)\right) \times \dot{g}^l\left(a_i^L(t)\right) \quad \text{(IV.24)}$$

Avec $\dot{g}^L\left(a_i^L(t)\right) = \frac{dg^L\left(a_i^L(t)\right)}{da_i^L(t)}$

Dans le cas ou l'indice d'un neurone caché $(1 \prec l \prec L-1)$

$$\underline{\delta}^l = \dot{G}^l\left(\underline{a}^l\right) \times \left(W^{l+1}\right)^T \times \underline{\delta}^{l+1} \quad \text{(IV.25)}$$

$$\text{Ou } \dot{G}^l\left(\underline{a}^l\right) = \begin{pmatrix} \dot{g}^l\left(a_1^l(t)\right) & 0 & \cdots & 0 \\ 0 & \dot{g}^l\left(a_2^l(t)\right) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \dot{g}^l\left(a_{N^l}^l(t)\right) \end{pmatrix} \quad \text{(IV.26)}$$

Pour résumer, l'algorithme de mise à jour des paramètres du réseau se déroule comme suit : premièrement le vecteur d'entrée $\left(x_1, x_2, \ldots\ldots, x_{N_0}\right)^T$ est propagé vers la sortie en utilisant les équations (IV.11) et (IV.12). Ensuite, on calcule les fonctions de sensibilités par rétropropagation de l'erreur de sortie à l'aide des équations (IV.24) et (IV.25). Finalement on modifie les poids et les biais en utilisant les équations (IV.22), (IV.23), (IV.16) et (IV.17).Cet algorithme, présenté ici dans sa version la plus simple, possède de nombreuses variantes. Elles correspondent pour la plupart à l'adaptation du coefficient d'apprentissage [25], ou à l'utilisation de méthodes du deuxième ordre pour le calcul du gradient [26].

IV.3.5. Conception d'un réseau de neurones

Les réseaux de neurones réalisent des fonctions non linéaires paramétrées. Leurs mises en œuvre nécessitent :

- La détermination des entrées et des sorties pertinentes, c'est à dire les grandeurs qui ont une influence significative sur le phénomène que l'on cherche à modéliser.
- La collecte des données nécessaires à l'apprentissage et à l'évaluation des performances du réseau de neurones.
- La détermination du nombre de neurones cachés nécessaires pour obtenir une approximation satisfaisante.
- La réalisation de l'apprentissage
- L'évaluation des performances du réseau de neurones à l'issue de l'apprentissage.

IV.3 .5.1. Détermination des entrées/sorties du réseau de neurones

Pour toute conception de modèle, la sélection des entrées doit prendre en compte deux points essentiels :

- Premièrement, la dimension intrinsèque du vecteur des entrées doit être aussi petite que possible, en d'autre terme, la représentation des entrées doit être la plus compacte possible, tout en conservant pour l'essentiel la même quantité d'information, et en gardant à l'esprit que les différentes entrées doivent être indépendantes.

- En second lieu, toutes les informations présentées dans les entrées doivent être pertinentes pour la grandeur que l'on cherche à modéliser : elles doivent donc avoir une influence réelle sur la valeur de la sortie.

IV.3 .5.2. Choix et préparation des échantillons

Le processus d'élaboration d'un réseau de neurones commence toujours par le choix et la préparation des échantillons de données. La façon dont se présente l'échantillon conditionne le type de réseau, le nombre de cellules d'entrée, le nombre de cellules de sortie et la façon dont il faudra mener l'apprentissage, les tests et la validation. Il faut donc déterminer les grandeurs qui ont une influence significative sur le phénomène que l'on cherche à modéliser. Lorsque la grandeur que l'on veut modéliser dépend de nombreux facteurs, c'est-à-dire lorsque le modèle possède de nombreuses entrées, il n'est pas possible de réaliser un « pavage » régulier dans tout le domaine de variation des entrées : il faut donc trouver une méthode permettant de réaliser uniquement des expériences qui apportent une information significative pour l'apprentissage du modèle. Cet objectif peut être obtenu en mettant en œuvre un plan d'expériences. Pour les modèles linéaires, l'élaboration de plans d'expériences est bien maîtrisée, par ailleurs, ce n'est pas le cas pour les modèles non linéaires.
Afin de développer une application à base de réseaux de neurones, il est nécessaire de disposer de deux bases de données, une pour effectuer l'apprentissage et l'autre pour tester le réseau obtenu et déterminer ses performances.

IV.3 .5.3. Elaboration de la structure du réseau

La structure du réseau dépend étroitement du type des échantillons. Il faut d'abord choisir le type de réseau : un perceptron standard, un réseau de Hopfield, un réseau à décalage temporel (TDNN), un réseau de Kohonen, etc...
Dans le cas du perceptron multicouches, il faudra aussi bien choisir le nombre de couches cachées que le nombre de neurones dans cette couche.

- ***Nombre de couches cachées :*** Mis à part les couches d'entrée et de sortie, il faut décider du nombre de couches intermédiaires ou cachées. Sans couche cachée, le réseau n'offre que de faibles possibilités d'adaptation.

- ***Nombre de neurones cachés :*** Chaque neurone peut prendre en compte des profils spécifiques de neurones d'entrée. Un nombre plus important permet donc de mieux ″coller″ aux données présentées mais diminue la capacité de généralisation du réseau. Il faut alors trouver le nombre adéquat de neurones cachés nécessaire pour obtenir une approximation satisfaisante.

IV.3 .5.4. Apprentissage

L'apprentissage est un problème numérique d'optimisation. Il consiste à calculer les pondérations optimales des différentes liaisons, en utilisant un échantillon. La méthode la plus

utilisée est la rétropropagation, qui est généralement plus économe que les autres en terme de nombres d'opérations arithmétiques à effectuer pour évaluer le gradient.

IV.3 .5.5. Validation et Tests

Alors que les tests concernent la vérification des performances d'un réseau de neurones hors échantillon et sa capacité de généralisation, la validation est parfois utilisée lors de l'apprentissage. Une fois le réseau de neurones développé, des tests s'imposent afin de vérifier la qualité des prévisions du modèle neuronal. Cette dernière étape doit permettre d'estimer la qualité du réseau obtenu en lui présentant des exemples qui ne font pas partie de l'ensemble d'apprentissage. Une validation rigoureuse du modèle développé se traduit par une proportion importante de prédictions exactes sur l'ensemble de la validation. Si les performances du réseau ne sont pas satisfaisantes, il faudra, soit modifier l'architecture du réseau, soit modifier la base d'apprentissage.

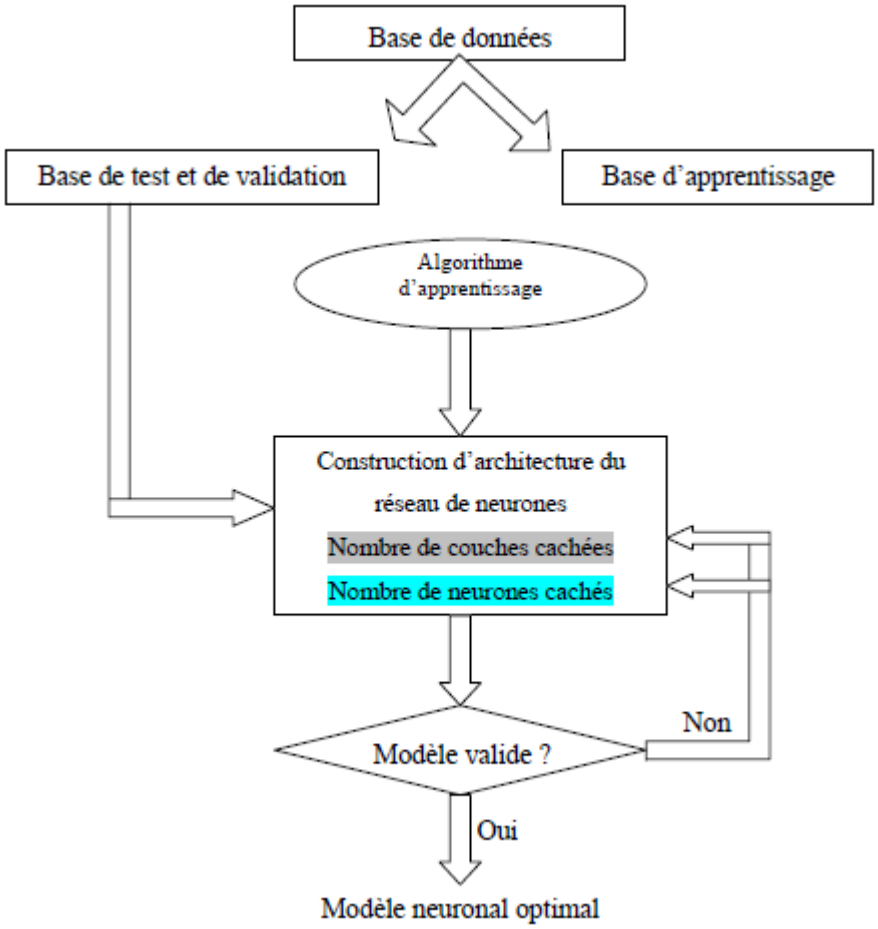


Figure. IV.6. Organigramme de conception d'un réseau de neurones.

IV.4. Algorithmes génétiques

L'AG est une métaphore biologique qui prend son inspiration des mécanismes de sélection naturelle et de génétique de l'évolution, et utilisées comme outil d'optimisation. Il est capable de localiser rapidement des solutions satisfaisantes pour des problèmes difficiles et complexes et sa structure est telle qu'on peut aisément mettre en œuvre sur l'ordinateur. Il utilise à la fois les principes de la survie des structures les mieux adaptées, et les échanges de l'information [27].

Les AG's présentent l'avantage d'explorer l'espace des solutions réalisables à partir d'un ensemble de solutions créées de manière aléatoire. A l'aide d'opérations spécifiques, l'algorithme génétique va générer de nouvelles solutions à partir de l'ensemble de solutions de départ, connu sous le nom de *population*. L'exploration de l'espace des solutions s'articule sur des mécanismes adaptés du domaine de la génétique. Ces algorithmes se basent sur le principe d'évolution des espèces, d'adaptation et de sélection naturelle. Les solutions réalisables ou acceptables sont assimilées à des individus qui vont résister et s'adapter à leur environnement. Seuls les meilleurs individus survivront. Par analogie, les meilleures solutions seront celles ayant une plus forte probabilité d'être choisies tout au long du processus de recherche de solutions. Les premiers travaux ont été menés par (Holland, 1975) dans l'ouvrage « *Adaptation of Natural and Artificial System* » qui formalise les algorithmes génétiques dans le cadre de l'optimisation mathématique. L'indisponibilité d'ordinateurs puissants, empêche l'implémentation de ces algorithmes sur des problèmes de taille réelle. Il faut attendre les travaux de (Goldberg, 1994) permettent de vulgariser l'utilisation des algorithmes génétiques et leur utilisation dans des problèmes d'optimisation concrets.
Le mot *gène* est un terme général définissant, l'unité héréditaire ou l'entité physique qui détermine le développement d'un caractère particulier. Les gènes, dans les algorithmes génétiques, sont tous simplement des caractères dont les valeurs sont appelées des *allèles*. Ils sont juxtaposés pour former ce qu'on appelle le *chromosome***.** Une *population* est un groupe *d'individus* (chromosome) qui se reproduisent entre eux, cette opération a pour but la transmission des gènes d'une génération à une autre.

IV. 4. 1. Description des AG's :

Un algorithme génétique implémente une version très simplifiée et très schématique des mécanismes de l'évolution biologique. A partir d'un problème qu'on cherche à résoudre, un algorithme génétique est un algorithme itératif de recherche globale dont le but est d'optimiser une fonction définie par l'utilisateur (le critère, la fonction de coût ou de profit) appelée fonction objectif (ou d'adaptation), pour atteindre cet objectif, on considère une population d'individus générée aléatoirement, distribués dans entièreté de l'espace de recherche. Chaque individu est décodé pour donnée un phénotype [28]**.**

Ce dernier est utilisé pour évaluer les individus en calculant la valeur de la fonction objectif qui normalement indépendante des autres individus de la population. Chaque individu ou chromosome est constitué d'un ensemble de gènes, pouvant prendre plusieurs valeurs appartenant a un alphabet non nécessairement numérique. Dans l'algorithme de base, les allèles possibles sont 0 et 1, un chromosome est une chaîne binaire. Le but est donc de rechercher une combinaison optimale de ces éléments qui donne lieu au maximum de la fonction objectif. A chaque itération, appelée génération, est créée une nouvelle population avec le même nombre d'individus. Cette nouvelle génération consiste généralement en des individus les mieux 'adaptés' à l'environnement tel qu'il est représenté par la fonction objectif. Au fur est à mesure des générations, les individus vont tendre en général vers

l'optimum de la fonction objectif. La génération d'une nouvelle population à partir de la précédente s'effectue en trois étapes :

Evaluation: l'algorithme génétique évalue la fonction objectif de chaque individu I de la l'ancienne population.

Sélection : on sélectionne le groupe de producteurs pour la génération suivante en utilisant une roue de loterie, cette est dite sélection par roue loterie, elle consiste a donné ou individus de fonction objectif plus grand une probabilité plus élevée d'être contribuer à la génération suivante. Cette opération est bien entendue une version artificielle de la sélection naturelle, la survie de darwinienne des chromosomes plus adaptés (qui en une fonction adaptation plus élevée). Dans les populations naturelles, l'adaptation est déterminée par la capacité d'une créature à survivre aux prédateurs, aux maladies, et aux autres obstacles à franchir pour atteindre l'âge adulte et la période de reproduction, dans notre environnement artificiel, la fonction objectif est l'arbitre final décidant de la vie ou la mort de chaque chromosome créature [28].

Reproduction avec croisement et mutation : l'algorithme génétique combine les individus sélectionnés au moyen d'opérateurs génétiques tels que la mutation et le croisement, qui d'un point de vue algorithmique, peuvent être considérés comme des mécanismes servant à changer localement les solutions représentées par les parents. La mutation agit en modifiant aléatoirement un ou plusieurs gènes d'un chromosome, pour un codage binaire la mutation revient à changer le 1 à 0 ou vice versa, tandis que le croisement échange certains gènes d'un parent avec ceux de l'autre. Ces deux opérations sont appliquées avec des probabilités fixées (pc pour le croisement, et pm pour la mutation) sur chacun des individus de la présente population. Plus précisément, pour le croisement, on choisira au hasard 2 parents dans la présente population, soit un entier k, représentant une position sur les chromosomes ($parent_1$et $parent_2$), est choisi aléatoirement entre 1et la longueur du chromosome moins 1 $[1, l-1]$, par exemple, considérons deux chromosomes C_1, C_2, supposons qu'en choisissant au hasard un nombre k=5

$parent_1$= xxxxxxxx

$parent_2$= yyyyyyyy

↑ Position du croissement

Le croisement des deux chromosomes parents à la position k=5 conduit à la production de deux nouveaux chromosomes dans la nouvelle génération :

$Enfant_1$= xxxxxyyy

$Enfant_2$= yyyyyxxx

Les nouveaux chromosomes constituent la nouvelle génération. Et le cycle continue : évaluation, sélection, reproduction, évaluation, etc. Le critère d'arrêt peut être un nombre de générations spécifiques, soit sur la variation de la fonction à optimiser.

Il est clair que l'opérateur croissement a un grand rôle dans la recherche globale de l'espace par combinaison de 'blocs de construction', l'opérateur de mutation à un rôle théoriquement plus marginale : il est la pour éviter les pertes.

Il y a donc trois paramètres de base pour le fonctionnement d'un algorithme génétique : le nombre d'individus dans la population (n). Les taux de mutation pm et de croissement pc.

Trouver les bonnes valeurs de ces paramètres est un problème parfois délicat. La valeur de n dépend fort du problème, tandis que des valeurs de bonne pratique pour pm=0.005 et pc=1.

IV. 4. 2. Codage et population initiale

Premièrement, il faut représenter les différents états possibles de la variable dont on cherche la valeur optimale sous forme utilisable pour un AG: c'est le codage. Cela permet d'établir une connexion entre la valeur de la variable et les individus de la population, de manière à imiter la transcription génotype-phénotype qui existe dans le monde vivant. Il existe principalement deux types de codage : le codage binaire, le codage réel.

- **Codage binaire :** Ou chaque phénotype est représenté par une chaîne de caractère d'alphabet {0,1}. Trois types de codage binaire sont utilisés: codage à base de 2, codage Gray, et le codage en virgule flottante si les paramètres à optimiser sont réels. Ces codages donnent une certaine flexibilité, mais ils ne constituent pas une méthode pratique pour le codage des problèmes auxquels on est confrontés dans les sciences, les affaires et surtout l'ingénierie. Une méthode de codage des problèmes d'optimisation multiparamétriques qui a été appliquée avec succès est celle du codage concaténé, multiparamétriques et à borne fixe [9].
- **Codage en nombre réel:** Des nombreuses applications des algorithmes génétiques ne se basent plus sur un codage binaire des paramètres à optimiser, mais travaillent directement sur les paramètres eux-mêmes [30], [31], [32]. Ces versions des algorithmes génétiques, qui sont appelées les algorithmes génétiques codés-réels, offrent généralement l'avantage d'être mieux adaptés aux problèmes d'optimisation numérique continus, d'accélérer la recherche et de rendre plus aisé le développement de méthodes hybrides avec les méthodes classiques. Ces versions sont sans doute plus proches des besoins et habitudes des praticiens industriels, pour la résolution de problèmes réels [33], c'est pourquoi elles sont plus répandues dans le milieu industriel que les algorithmes génétiques à codage binaire. Cependant les algorithmes génétiques codés-réels nécessitent le développement des opérateurs (croissement, mutation, sélection).

IV. 4. 3 Opérateur de croissement

Le croisement est le principal opérateur agissant sur la population des parents. Il permet de créer de nouveaux individus par l'échange d'information entre les chromosomes par leur biais de leur combinaison. Les différents opérateurs de croissement utiliser dans les algorithmes génétiques, certains d'entre eux sont utilisés pour un codage binaire et d'autres pour un codage réel.

- **Croissement binaire**

1. Croissement un point : est le cas le plus simple.
2. Croissement à multiple points : la version la plus simple de ce croissement est à deux points. Le croisement en deux points est le même qu'un point sauf au lieu de choisir un seul point de croissement on choisit deux points et on échange l'ensemble de caractères existant entre ces deux points [34].
3. Croissement uniforme: On associe à chaque bit de parents1 une probabilité d'être échanger avec le bit qui correspond dans le parent2, la probabilité de croissement doit être assez grande pour assurer la différence entre la nouvelle et précédente génération [35].

- **Croissement réel**

1. Croissement simple : chaque nombre réel de vecteur des paramètres est traité comme un bit de codage binaire, il peut être échangé sauf avec l'élément qui lui correspond dans l'autre parent. Comme le cas de croissement binaire, le croissement simple peut être à un point, à deux points ou à multiple point [38,37].

2. Croissement aléatoire : croisement aléatoire combine deux parents selon une chaîne binaire générée aléatoirement. Chaque nombre réel de parent1 échangé avec le nombre réel qui le correspond dans parent2 si le caractère de la chaîne aléatoire est « 1 », si le caractère

Parent1 :xxxxxxxxxxxxxxxxxxx

Parent2 :yyyyyyyyyyyyyyyyyy

La chaîne aléatoire:00111110010100000001

Enfant1:xxyyyyyyxxyxyxxxxxxy

Enfant2:yyyxxxxxyyyxyxyyyyyy

Figure IV.6. L'opérateur de croissement aléatoire

est «0 » aucun changement va se produire (figure IV. 6).

3. Croissement arithmétique : dans les nouveaux chromosomes (enfant1 et enfant2) sont obtenus par la combinaison linéaire des vecteurs parent 1 et parent2

$$enfant_1 = C_1 parent_1 + C_2 parent2$$
$$enfant_2 = C_2 parent_1 + C_1 parent2 \qquad \text{(IV.27)}$$

Les coefficients C_1 et C_2 peuvent être constants ou générer aléatoirement à chaque croissement. La condition $C_1 + C_2 = 1$ doit être respectée. Il possible d'imaginer des autres types de croisement: Par exemple, le croisement 'biaisé' qui produirait un enfant du meilleur des 2 parents, ou encore un croisement effectue une opération de type 'moyenne' sur les gènes choisis au hasard des deux parents.

Enfin, pour terminer ces descriptions, nous pouvons imaginer d'autres cas de combinaisons : par exemple, au lieu de prendre deux parents, on peut prendre beaucoup plus que deux parents. Cela permet bien sur d'accélérer la recherche, car on utilise pour la reproduction des mécanismes de croisement forts axés sur l'exploitation locale de l'espace de recherche à la différence des règles traditionnelles de croisement qui font une grande part de hasard [36].

IV.4.4 Opérateur de mutation

Le rôle de cet opérateur est de modifier aléatoirement la valeur d'un gène dans un chromosome. Dans le cas du codage binaire, chaque bit $a_i \in \{0,1\}$ est remplacé par son complémentaire $a_i = 1 - a_i$.

Dans le cas d'un codage réel, on utilise principalement deux opérateurs de mutation: la mutation uniforme et la mutation non uniforme. En supposant fixée la probabilité de mutation p_m, un tirage au sort pour chaque gène x_k d'un chromosome permet de décider si ce gène doit être ou non modifié. Nous supposons que le gène prend ses valeurs dans un intervalle $\left[x_k^{\min}, x_k^{\max}\right]$.

Pour la mutation uniforme, qui est une simple extension de la mutation binaire, on remplace le gène x_k sélectionné par une valeur quelconque x'_k tirée aléatoirement dans l'intervalle $\left[x_k^{\min}, x_k^{\max}\right]$.

Pour la mutation non uniforme, le calcul de la nouvelle valeur d'un gène est un peu plus complexe. Le gène x_k subit des modifications importantes durant les premières générations, puis graduellement décroissantes au fur et à mesure que l'on progresse dans le processus d'optimisation. Pour une génération t, on tire au sort une valeur binaire qui décidera si le changement doit être positif ou négatif. La nouvelle valeur x'_k du gène x_k est donnée par :

$$x'_k = \begin{cases} x_k = x_k + \Delta\left(t, x_k^{\max} - x_k\right) \; si \;\; rand = 1 \\ x_k = x_k + \Delta\left(t, x_k - x_k^{\min}\right) \; si \;\; rand = 0 \end{cases} \quad \text{(IV.28)}$$

Où $\Delta(t, y)$ est une fonction qui définit l'écart entre la nouvelle valeur et la valeur initiale. Les auteurs proposent d'utiliser une fonction $\Delta(t, y)$ correspondante à une décroissance exponentielle de l'écart à travers les générations. Cette fonction est définie par :

$$\Delta(t, y) = y \times \left(1 - r^{(1-t/T)^{\beta}}\right) \quad \text{(IV.29)}$$

T est le nombre de génération, β est un paramètre de l'opérateur de mutation (souvent $\beta = 5$), r est un nombre produit aléatoirement dans l'intervalle [0,1] et t le numéro de la génération.

IV.4.5. Fonction d'évaluation

Chaque chromosome apporte une solution potentielle au problème à résoudre. Néanmoins, ces solutions n'ont pas toutes le même degré de pertinence. C'est à la fonction de performance (*fitness*) de mesurer cette efficacité pour permettre à l'AG de faire évoluer la population dans un sens bénéfique pour la recherche de la meilleure solution. Autrement dit, la fonction de performance, $f(.)$, doit pouvoir attribuer à chaque individu un indicateur positif représentant sa pertinence pour le problème qu'on cherche à résoudre. Nous la nommerons fonction d'adaptation où objective $f(.)$ **(**fitness function).

IV.4.6. Opérateur de sélection

La sélection est le premier opérateur génétique appliqué à une population d'individus en vue de la renouveler. Cet opérateur base la constitution de la nouvelle population sur le Fitness des individus de la population qui la précède. Le principe de la sélection est tel que les individus les mieux adaptés fournissent la descendance la plus nombreuse. La sélection peut être réalisée selon différentes méthodes, dont nous citons principalement:

Méthode de la roulette : Cette méthode est ainsi dénommée car elle consiste à attribuer à chaque individu, un secteur de la roue de loterie proportionnel à son fitness relatif. La sélection de N individus, N étant la taille de la nouvelle population, est réalisée en effectuant

N tirages de la roue biaisée. En pratique, ceci revient à calculer pour chaque individu, une probabilité p_i de survie proportionnelle à sa performance et calculée comme suit :

$$p_i = \frac{fitness(ind_i)}{\sum_{i=1}^{N} fitness(ind_i)} \qquad 0 < p_i < 1 \quad \text{(IV.30)}$$

On effectue alors un calcul d'une probabilité de sélection q_i tel que $q_i = \sum_{j=1}^{i-1} p_j$ puis on génère aléatoirement un nombre **r** sur l'intervalle $[0,1]$, N fois de suite. Un individu ind_i est sélectionné lorsque $q_{i-1} < r < q_i$

Méthode du rang : Cette méthode a été proposée par J.BAKER en 1985 afin de pallier aux problèmes soulevés par la méthode de la roue de loterie. La méthode du rang consiste principalement à ordonner les individus en fonction de leur performance, et ce du meilleur, de rang 1, jusqu'au moins performant, de rang N. La probabilité de sélection d'un individu de rang i est calculée comme :

$$Ps(i) = D_{\max} - (D_{\max} - D_{\min}) \times \frac{(i-1)}{(N-1)} \quad \text{(IV.31)}$$

$Ps(i)$: Probabilité de sélection de l'individu de rang i

$D_{\max}$: Nombre maximal de descendants par individu

$D_{\min}$: Nombre minimal de descendants par individu

Avec : $D_{\min} = 2 - D_{\max}$, $0 \le D_{\max} \le 2$

IV. 4. 7. Performances d'un algorithme génétique

Pour définir complètement la simulation d'un algorithme génétique, on doit choisir un ensemble de paramètres qui sont la probabilité de croissement Pc , la probabilité de mutation Pm, le nombre de population et la taille des chromosomes, De Jong a effectué une série d'analyses des paramètres pour l'optimisation de la fonction objectif, cette étude suggéré que pour avoir de bonnes performances, il faut choisir une probabilité de croissement élevée, une probabilité de mutation faible(inversement proportionnelle à la taille de population) et population de taille modérée. Schaffer et Al ont développé une équation qui donne approximativement les paramètres optimaux d'un algorithme génétique pour différents problèmes d'optimisation.

$$\ln(n) + 0.93\ln(p_m) + 0.456\ln(l) = 0.56 \qquad \text{(IV.32)}$$

Ou n est la taille de population ; Pm est la probabilité de mutation, et l est la longueur de chromosome.

Cette équation peut être approximée par :

$$n * p_m * \sqrt{l} = 1.7 \qquad \text{(IV.33)}$$

Cette formule indique que la probabilité de mutation est inversement proportionnelle la taille de population. Fogarty a trouvé que la probabilité de mutation diminuée avec le nombre de générations. Mais Muhlenbein a démontré que la probabilité de mutation qui donne les meilleurs éléments diminués avec la longueur de chromosome, et la probabilité de croissement qui produit les meilleurs individus augments avec le nombre la longueur de chromosome.

Paramètre	Code	Niveau(-)	Niveau(+)
Taille de population Maximum de génération Type de croissement Probabilité de croissement Probabilité de mutation	Pop Gen pt Pc Pm	50 200 A deux points 0.5 0.04	150 800 Uniforme 0.85 0.08

Tableau IV. 1. Les paramètres optimaux pour un AG

IV. 4. 8. Contraintes

Nous avons jusqu'à maintenant considéré l'utilisation des algorithmes génétiques pour l'exploration de fonctions non-contraintes. Cependant la plus part des problèmes réels contiennent une ou plusieurs contraintes qui doivent aussi satisfaites.

Les contraintes sont habituellement exprimées sous formes d'équations ou d'inéquations. Puisque les contraintes d'égalité peuvent être inclues dans le modèle du système. Mais pour les contraintes inégalité on procède comme suit :

Pour chaque nouvel individu, on évalue la fonction à optimiser, et on vérifie qu'aucune des contraintes n'est violée. Si tel est le cas, la fonction à optimiser gardera sa valeur. Si certaines contraintes sont violées, la solution n'est pas acceptable, et la fonction à optimiser n'a pas de valeur.

En général ces contraintes sont prises en compte pour des problèmes très contraints, mais trouver un point acceptable est presque difficile que trouver le meilleure, pour cela on fait appel aux méthodes de pénalisation qui transforment un problème d'optimisation contrainte en un

IV.4.9. Algorithmes hybrides

Problème non contraint en associant un coût d'évaluation de la fonction objectif [35]. Tentant à présent d'améliorer la procédure d'inversion en concevant une méthode d'optimisation plus efficace, à savoir une méthode offrant à la fois une meilleure fiabilité et une meilleure précision tout en nécessitant un temps de calcul raisonnable. L'idée consiste à combiner deux méthodes d'optimisation dans le but de marier leurs avantages et de réduire leurs inconvénients. Les méthodes d'hybridation les plus utilisées sont:

La méthode d'hybridation séquentielle **:** Dans cette méthode il suffit de lancer l'algorithme génétique à un niveau conséquent de la convergence; ensuite, on Permet à la procédure d'optimisation locale de prendre le relais, en prenant un pourcentage de la dernière génération, les individus générés par cette approche vont donner une certaine diversité dans la population de l'algorithme génétique, permettant ainsi à des sous-populations stables de se former à différents pics dans le domaine de la fonction .

La méthode d'hybridation d'implantation parallèle **:** Dans cette méthode on envisage de disposer de nombreux processeurs en parallèles ayant la capacité de traitement satisfaisante pour que l'évaluation de fonction soit réalisée simultanément pour différents individus dans la génération. De cette façon, les procédures parallèles peuvent être utilisées pour évaluer les valeurs de la fonction objective.

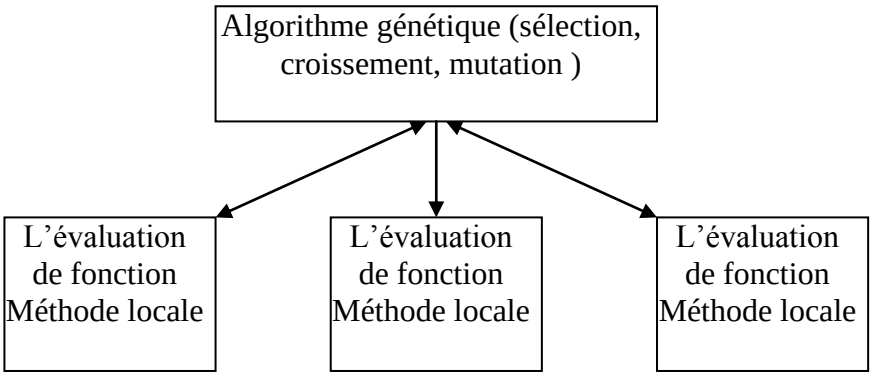


Figure IV. 7. Schéma d'un algorithme génétique hybride parallèle

Une méthode fondamentale d'exploration locale qui peut donner lieu à une hybridation avec l'algorithme génétique est l'amélioration par G-bit (amélioration bit par bit de type Gradient). Elle comprend les étapes suivantes :

1. Choisir au moins un des meilleurs individus de la population actuelle.
2. Balayer les individus bit par bit, en réalisant à chaque fois un changement de bit, tout en gardant les meilleurs de ces deux opérations.
3. A fin de ce balayage, insérer les meilleures structures dans la population et continuer l'exploration génétique normale.

Conception de prédicteurs neuronaux par algorithmes génétiques réels

V. 1. Introduction

L'étude des réseaux de neurones artificiels (RNA) et l'étude des algorithmes génétiques (AG's) se sont développés en parallèle, ils ont souvent été en interaction durant cette dernière décennie. On distingue deux types d'interactions : l'utilisation conjointe des RNA et des AG's pour la résolution d'un même problème, ou l'emploi des AG's pour le développement de RNA. Nous nous arrêtons dans cette présentation sur ce deuxième aspect.

Les difficultés rencontrées dans la conception des prédicteurs neuronaux (PN) ont guidé les chercheurs à s'orienter vers l'utilisation des algorithmes génétiques (AG's) à cause de leur caractéristique d'exploration globale dans un environnement complexe. Le prédicteur neuronal est défini par un nombre assez important de paramètres et tous ces paramètres peuvent faire l'objet d'une optimisation par les AG's.

1. Les poids des connexions constituent le plus souvent des variables dont les valeurs sont optimisées par les AG; l'intervalle de variation des poids synaptiques des couches cachée et de couche de sortie est l'un des paramètres de contrôle de la complexité du réseau son optimisation devrait permettre d'améliorer la capacité de généralisation du réseau.
2. La structure de prédicteur neuronal: le nombre de couches, les nombres de neurones par couche, la forme de la fonction d'activation utilisée par couches, la présence ou l'absence de biais, le nombre d'entrée de prédicteur et le retards (lag) sont des donnés que nous pouvons l'optimiser par les AG's.
3. les AG's peuvent être utilisée pour l'optimisation des paramètres de contrôle de prédicteur neuronal ; il peut s'agir du coefficient d'apprentissage, coefficient de l'inertie, nombre de pas d'apprentissage.

Dans notre travail, on s'intéresse aux trois stratégies simultané, qui s'avère être la plus efficace car en toute logique, une méthode traite globalement les différents paramètres d'un prédicteur neuronal devrait donner de meilleures solutions.

V. 2. Technique employée

V.2.1 Représentation de chromosome

Les opérateurs des AG's agissent directement sur une représentation des individus (RNA) à optimiser, alors que l'évaluation de ces individus est effectuée à partir de leur comportement.

Le choix de la représentation est indissociable du choix des opérateurs génétiques(OG) et le couple <représentation, opérateurs> conditionne le résultat de l'application de l'AG. Idéalement, le choix effectué devrait assurer la complétude de la recherche et la validité de tout individu obtenu et la présence d'une métrique convenable sur l'espace des représentations. Dans la pratique, plusieurs problèmes se posent :

La recherche est complète dans la mesure où toute solution possible peut être trouvée, quelque soit la population initiale. Pour les RNA, l'espace de recherche est extrêmement vaste ; il est donc indispensable de restreindre l'espace de recherche aux seules solutions potentiellement

utiles. Toute la difficulté est de définir ce nouvel espace et de s'assurer qu'il contient la solution du problème à résoudre. En tenant compte des résultats connus concernant les propriétés d'approximation universelle de différentes architectures de RNA ou nous pouvons réduire significativement l'espace de recherche sans diminuer sa puissance de modélisation. Les représentations peuvent alors être plus compactes mais il faut définir les OG attentivement afin de garantir une recherche complète.
Pour que l'AG fonctionne normalement il est nécessaire que toute représentation obtenue par application des OG's corresponde à un RNA de l'espace de recherche (la représentation est valide). D'une part, on réduit l'espace des représentations pour améliorer l'efficacité de la recherche, D'autre part, il est difficile de définir des OG's capables d'assurer non seulement la complétude de la recherche, mais aussi la validité des solutions obtenues.
L'optimisation simultanée des poids synaptique et la topologie de réseaux ainsi que les paramètres de contrôle représentent une tâche plus motivante pour les algorithmes génétiques. La difficulté dans cette tâche réside dans le choix de la représentation de chromosome convenable. Il existe trois types de représentation :
Codage direct, codage grammaire et codage indirect

- ***Le codage binaire direct :*** est souvent mis en œuvre par la représentation des connexions entre les neurones à l'aide d'une matrice de connexion C de taille $N \times N$ Où N est le nombre maximal de neurones dans le réseau. Le codage des connexions entre les neurones exige que la présence de connexion entre neurone i et d'autre j qui est représentée par $c_{i,j} = 1$ et l'absence est représentée $c_{i,j} = 0$. Le codage binaire direct utilise un chromosome de taille très grand ce qui diminue l'efficacité des AG.

- ***Le codage de grammaire matriciel :*** est la représentation de chromosome qui semble avoir les meilleures propriétés d'adaptabilité par rapport au premier codage. Le codage de grammaire matriciel, dans lequel les chromosomes codent une grammaire de génération de graphique. Une règle de grammaire récrit un $l \times l$ la matrice dans un 2×2 la matrice, et cela jusqu'à ce qu'il produise un $2^k \times 2^k$ la matrice, tel que $2^k < N$, le nombre maximal de neurones dans le réseau. Une matrice $N \times N$ est extraite du $2^k \times 2^k$ la matrice. Chaque gène est présenté par 0 ou 1, pour produire la matrice de connectivité du réseau. Malheureusement, ce codage n'est efficace que pour une petite topologie de réseau.

- ***Le codage indirect ou paramétrique*** : utilise une représentation de la topologie des réseaux. Cela en présentant le nombre de couches et le nombre de neurones dans chaque couche au lieu de la matrice de connexion. Cependant, ce type de codage peut être seulement représenté d'architecture.

Dans ce travail, on propose d'utilisation des algorithmes génétiques qui ne compte pas sur la longueur d'un chromosome prédéterminé. Utilise un codage réel de l'intervalle de variation des poids synaptique du réseau de neurones, la topologie ainsi que les paramètres de contrôle de réseaux de neurones. Ce codage est utilisée avec une fonction objectif qui mesure l'efficacité de réseau cela en calculant une erreur quadratique moyenne entre la sortie de réseau et la sortie réelle.
Nous considérons une représentation spécifique des paramètres liés au poids synaptique, la structure de prédicteur neural ainsi que les paramètres de contrôle; le chromosome est constitué de 17 gènes : Le nombre de couches cachées (NHL), les nombres de neurones dans chaque couche l (Nhl) et d'autres gènes représentant la forme de la fonction d'activation des

couches cachées et de couche de sortie (f^{l}, f^{out}), le nombre de neurones la couche d'entrée (N), la présence ou l'absence de biais (b), le temps entre deux entrée successif retard (δ), le coefficient d'apprentissage (η), coefficient d'inertie (α), l'intervalle de variation des poids synaptique pour les couches cachées $[low_1, high_1]$, $[low_2, high_2]$ et la couche de sortie $[low_3, high_3]$.

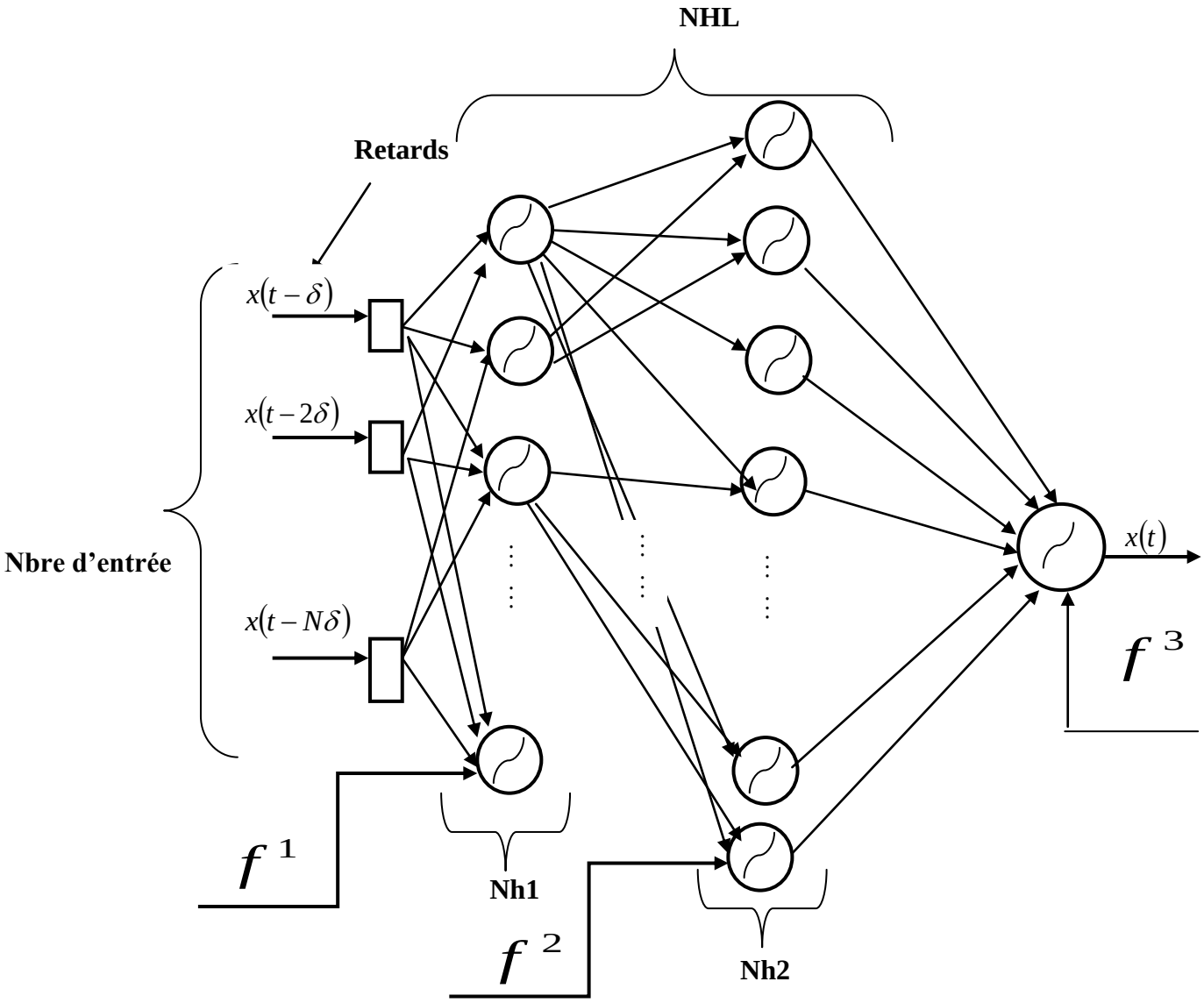


Figure V.1 Représentation des paramètres à optimiser

V.2.2. Structure du chromosome

Dans le paragraphe précédent, nous avons présenté le codage des paramètres permettant l'optimisation de topologie, les paramètres de contrôle, les intervalles initiales des poids des couches cachées et couche de sortie de réseau de neurones. Chaque chromosome de l'AG Constitué un réseau de neurones. Chaque allèle dans le chromosome est défini dans un sous-ensemble $S_i, \mathrm{i} = \{1,.....,7\}$ présenté dans le tableau V.1.

	$NHL\ \eta\ \alpha\ b\ N\ Nh1\ Nh2\ \delta\ f^1\ f^2\ f^{out}\ high_1\ low_1\ high_2\ low_2\ high_3\ low_3$
Set	$S_1\ S_2\ S_2\ S_3\ S_4\ S_5\ S_5\ S_6\ S_7\ S_7\ S_7\ S_2\ S_2\ S_2\ S_2\ S_2\ S_2$

Tableau V.1. Le sous-ensemble de chaque gène

Puisque c'est suffisant dans la plupart des applications des réseaux de neurone d'utiliser un petit nombre de couches cachées, la valeur de (NHL) est prise entre 1 à 2 tandis que la valeur maximale pour (Nhl) est calculé par d'équation :

$$\max\_Nhl = \frac{(N+1)}{2} + \sqrt{n\_data} \quad \text{(V.1)}$$

n_data Nombre des données utilisé dans la phase l'apprentissage. Tous les allèles sont définis dans un sous-ensemble.

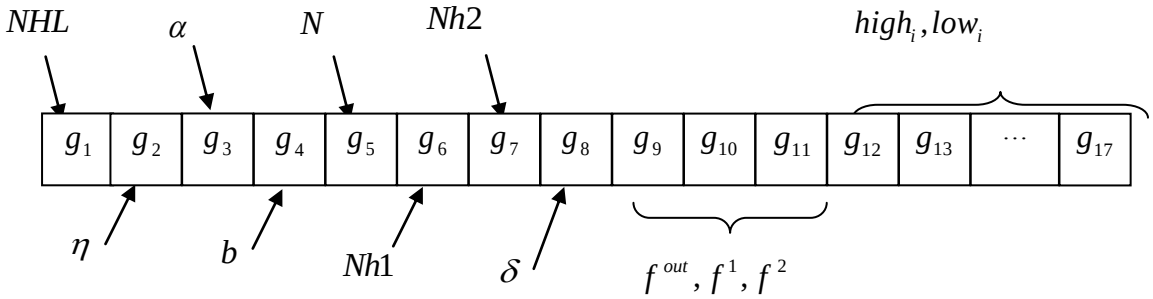


Figure V. 2 Représentation de chromosome

$S_1 = \{1,2\}$, $S_2 = [0,1], S_3 = \{0,1\}, S_4 = \{2,3,...,30\}, S_5 = \{3,4,....,40\}, S_6 = \{2,3,....,10\}, S_7 = \{1,2,3\}$

S_7 Signifie que la fonction d'activation peut prendre la valeur $1 \equiv f_1$ pour la forme sigmoïde, $2 \equiv f_2$ pour la forme tangente hyperbolique, $3 \equiv f_3$ pour le pur linéaire. S_3 Signifie $0 \equiv$ l'absence de biais, $1 \equiv$ la présence de biais. $[low_i, high_i]$ Présentent l'intervalle de variation des poids synaptique dans les deux couches cachées et la couche de production.

V.2.3. Opérateurs Génétiques

Rappelons d'abord que le choix des OG à utiliser est indissociable du choix de la représentation des RNA. Nous pouvons distinguer entre deux grandes familles d'opérateurs : les opérateurs qui agissent sur les représentations individuelles et ceux qui opèrent sur plusieurs représentations pour produire une nouvelle. Nous appellerons ici les premiers des

mutations et les seconds des recombinaisons. La sélection des individus peut être regardée aussi comme un opérateur ; Au sens le plus général, une mutation opère sur une seule représentation existante pour en produire une nouvelle. La modification effectuée peut changer ou non le nombre d'éléments de la représentation. Dans les AG classiques, l'opérateur de mutation agit sur les valeurs binaires ou réelles de ces éléments avec une probabilité constante ou qui diminue dans le temps ; cette probabilité reste cependant indépendante de l'identité des éléments d'une représentation et de l'identité des individus de la population. Plusieurs variations ont été proposées dans ce travail la mutation utilisée est :

1. Sélection

Sélection d'une paire d'individus de la population. Il a plusieurs type de sélection dans ce travail nous avons utilisé ''*remainder selection''*.

2. Croissement

Croisement est le principal opérateur agissant sur deux chromosomes. Il permet de créer de nouveaux individus par l'échange d'information entre les chromosomes par leur biais de leur combinaison. Dans notre cas, on 'a utilisé le croissement heuristique dans l'équation :

$$child = parent_2 + R'\left(parent_1 - parent_2\right) \quad \text{(V.2)}$$

Ou R=1.2

2. Mutation

Le rôle de cet opérateur est de modifier aléatoirement la valeur d'un gène dans un chromosome. Dans le cas d'un codage réel, on utilise la mutation uniforme. En supposant la probabilité de mutation $\boldsymbol{p_m}$, un tirage au sort d'un gène x de chromosome permet de décider si ce gène doit être ou non modifié. Nous supposons que le gène prend ces valeurs $[x_{\min}, x_{\max}]$ pour la mutation uniforme, qui est une simple extension de la mutation binaire, on remplace le gène x sélectionné par une valeur quelconque x' tirée aléatoirement dans l'intervalle $[x_{\min}, x_{\max}]$.

V.2.4. Initialisation des chromosomes

Le chromosome dans la population initiale est généré aléatoirement. Les chromosomes de la première population sont initialisés avec des valeurs aléatoires ou avec des valeurs qui dérivent le savoir-faire des experts pour accélérer la convergence de l'algorithme d'optimisation. Nous avons utilisé une combinaison de ces deux méthodes, ainsi nous initialisons les paramètres de topologie, contrôle, les bornes des intervalles des poids initiales des couches cachée et couches de sortie par des valeurs aléatoires prises dans leur un sous ensemble $S_i, \mathrm{i} = \{1,.....,7\}$.

V.2.5. Critères de performance

En générale, l'objectif de prédiction des séries temporelles est de minimiser l'écarte entre la valeur prédite et a valeur réelle de série temporelle. Il existe plusieurs critères numériques permettant de mesurer la qualité de prédiction. Parmi ces critères on peut citer **:**

Erreur quadratique Moyenne Normalisée (NMSE)

$$NMSE = \frac{\sum_i \left(Forecasted_i - Measured_i\right)^2}{\sum_i \left(Forecasted_i - \overline{Measured}_i\right)} \quad \text{(V. 3)}$$

Erreur Absolue (MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N} \left|Forecasted_i - Measured_i\right| \quad \text{(V. 4)}$$

Erreur biais moyenne(MBE)

$$MBE = \frac{1}{N}\sum_{i=1}^{N} \left(Forecasted_i - Measured_i\right) \quad \text{(V. 5)}$$

Erreur quadratique Moyenne :

$$MSE = \frac{1}{N}\sum_{i=1}^{N} \left(Forecasted_i - Measured_i\right)^2 \quad \text{(V. 6)}$$

V.2.6. Paramètres de l'AG

L'algorithme génétique exploitant le codage des paramètres décrits précédemment doit permettre d'optimiser simultanément la topologie, paramètres de contrôle, l'intervalle des poids initiaux des couches cachées et couche de sortie. Pour se faire, on doit choisir soigneusement les valeurs des paramètres régissant l'évolution de la population traitée par cet algorithme génétique: taille de la population, probabilités de croisement et de mutation. Dans ce travail, après une série de tests, nous avons opté pour les paramètres du tableau V.2.

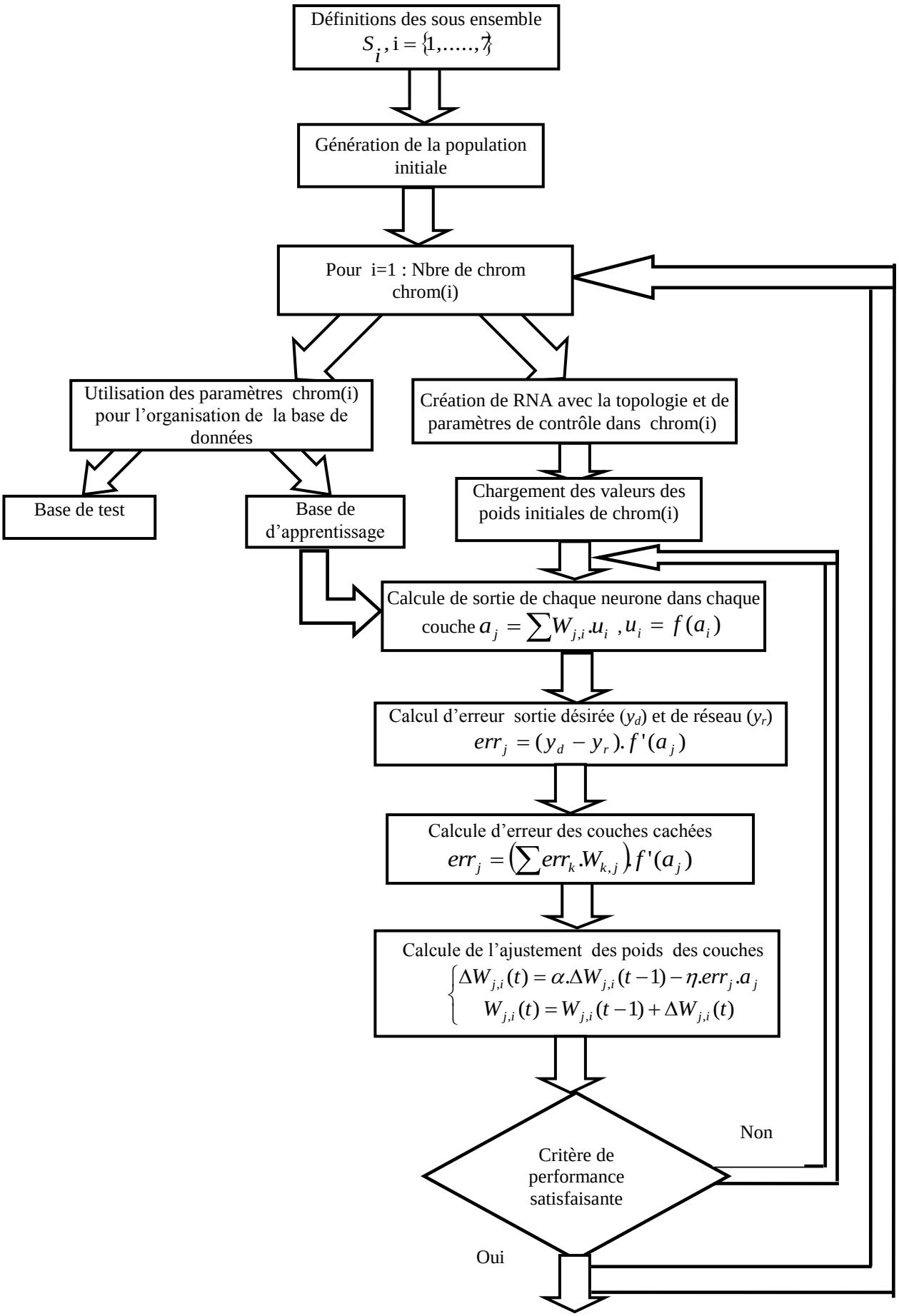
Représentation	Réelle
Taille de chromosome	17
Probabilité de croissement	1
Probabilité de mutation	0.1
Taille de population	50
Nombre de génération	50-300

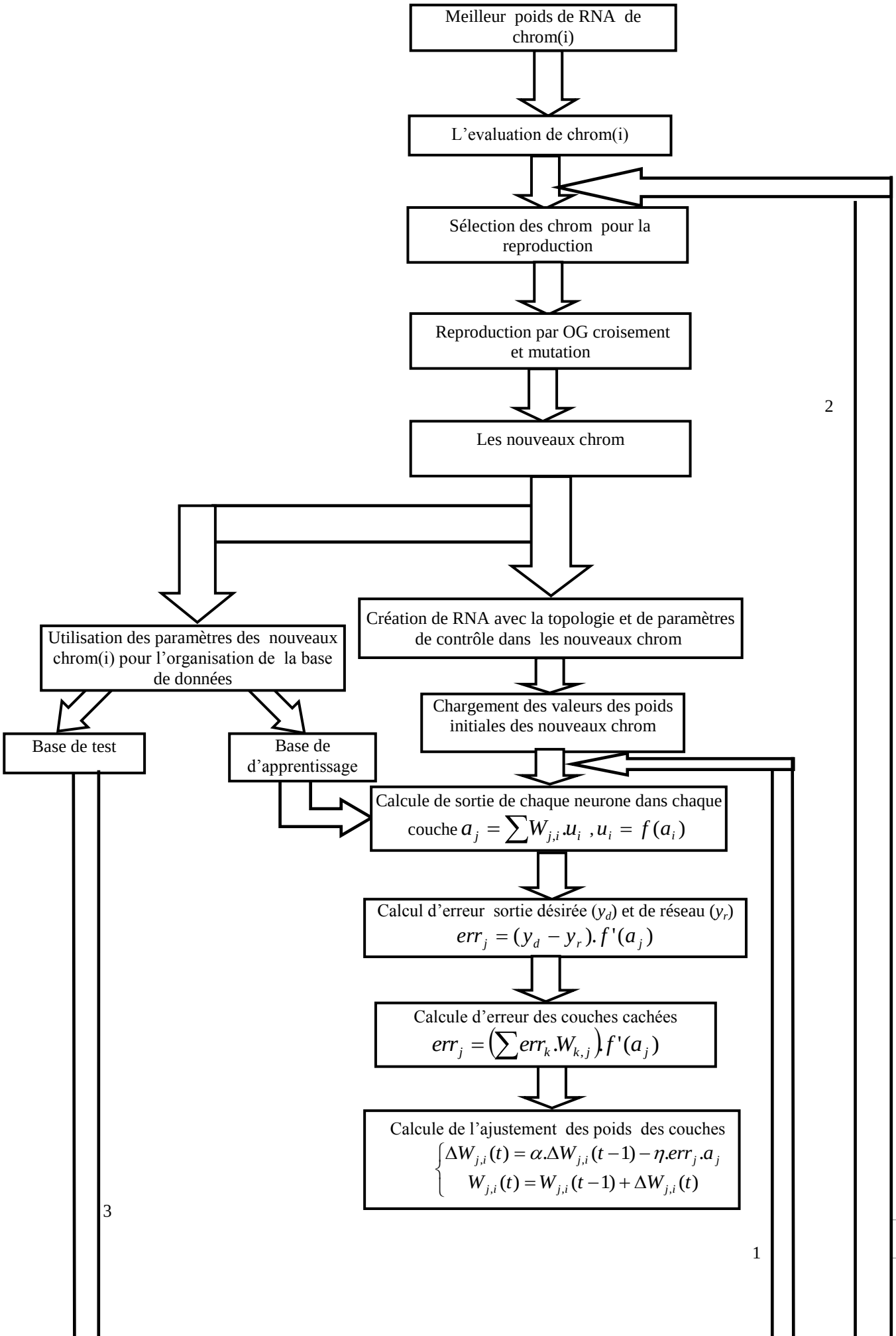
Tableau V.2 Paramètres de l'AG

V. 2 .7 Fonction Objectif

On fixe comme objectif, la minimisation de l'erreur e(t) entre la valeur actuelle de séries temporelle et valeur prédite. Cet objectif peut être défini par plusieurs indices numériques (NMSE, MAE, MBE, MSE), Dans notre travail, nous avons adopté pour la minimisation de l'erreur quadratique moyenne MSE.

$$fitness = MSE \quad \text{(V.7)}$$


Définitions des sous ensemble
S_i, i = {1,.....,7}
Génération de la population initiale
Pour i=1 : Nbre de chrom
chrom(i)
Utilisation des paramètres chrom(i) pour l'organisation de la base de données
Création de RNA avec la topologie et de paramètres de contrôle dans chrom(i)
Base de test
Base de d'apprentissage
Chargement des valeurs des poids initiales de chrom(i)
Calcule de sortie de chaque neurone dans chaque couche $a_j = \sum W_{j,i}.u_i$, $u_i = f(a_i)$
Calcul d'erreur sortie désirée (y_d) et de réseau (y_r)
$err_j = (y_d - y_r).f'(a_j)$
Calcule d'erreur des couches cachées
$err_j = \left(\sum err_k.W_{k,j}\right).f'(a_j)$
Calcule de l'ajustement des poids des couches
$\Delta W_{j,i}(t) = \alpha.\Delta W_{j,i}(t-1) - \eta.err_j.a_j$
$W_{j,i}(t) = W_{j,i}(t-1) + \Delta W_{j,i}(t)$
Critère de performance satisfaisante
Non
Oui


Meilleur poids de RNA de chrom(i)
L'evaluation de chrom(i)
Sélection des chrom pour la reproduction
Reproduction par OG croisement et mutation
Les nouveaux chrom
2
Création de RNA avec la topologie et de paramètres de contrôle dans les nouveaux chrom
Utilisation des paramètres des nouveaux chrom(i) pour l'organisation de la base de données
Chargement des valeurs des poids initiales des nouveaux chrom
Base de test
Base de d'apprentissage
Calcule de sortie de chaque neurone dans chaque couche $a_j = \sum W_{j,i}.u_i \ , u_i = f(a_i)$
Calcul d'erreur sortie désirée (y_d) et de réseau (y_r) $err_j = (y_d - y_r).f'(a_j)$
Calcule d'erreur des couches cachées $err_j = \left(\sum err_k.W_{k,j}\right).f'(a_j)$
Calcule de l'ajustement des poids des couches $\begin{cases} \Delta W_{j,i}(t) = \alpha.\Delta W_{j,i}(t-1) - \eta.err_j.a_j \\ W_{j,i}(t) = W_{j,i}(t-1) + \Delta W_{j,i}(t) \end{cases}$
3
1

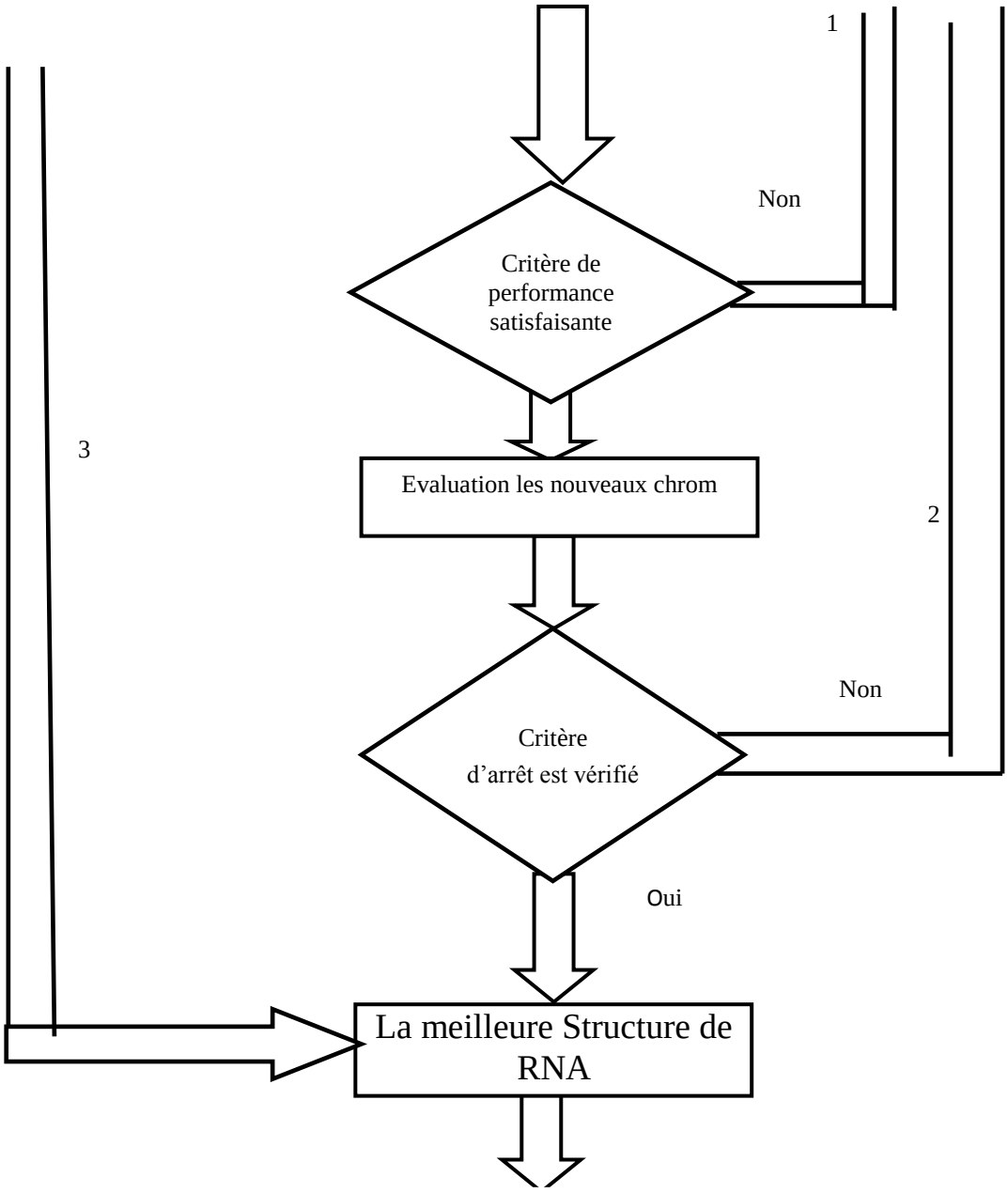


Figure V.3 L'organigramme de prédicteurs hybride RNA/AG

V .3. Application au domaine d'économie : la prédiction des Taux de change

V.3.1.Les apports de l'intelligence artificielle à l'amélioration du processus de décision financière et économique

Depuis le début de la décennie, 1990, les applications de l'intelligence artificielle en finance se sont multipliées. Elles concernent trois principaux domaines : la prévision, l'optimisation et la classification. La finalité d'utilisation de ces nouvelles et de fournir une information fiable au moment opportun et au moindre coût. En effet, la valeur d'une information est déterminée par son utilisation, c'est-à-dire est fonction du résultat de la décision dans laquelle elle est utilisée. De plus, certains chercheurs et praticiens insistent sur le fait que la valeur de l'information s'accroît avec son actualité, son exhaustivité, son exactitude et sa fiabilité. Or, ce type de méthodes pourrait parfaitement satisfaire ces conditions.

La revue de la littérature fait ressortir différentes finalités d'utilisation de techniques modernes par opposition aux méthodes statistiques traditionnelles. Les outils intelligents représentent une source importante de solutions pour des problèmes variés, qui touchent essentiellement à des aspects stratégiques de la vie de l'organisation. Les réseaux de neurones ont été utilisés à des fins de prévision en exploitants le passé d'une variable en vue d'extraire des relations permettant de prédire sa valeur future. En outre, les réseaux de neurones sont capables de découvrir la forme de ces relations (linéaire, non linéaire). Ainsi, les réseaux de neurones ont été utilisés pour prévoir la volatilité des indices boursiers, le taux de change, le taux d'inflation dans une économie donnée (Paquet, 1997).

En particulier, une étude (Tir & Abbas, 2004) montre comment les prévisions d'indices boursiers peuvent être améliorées en utilisant le réseau neuronal. En fait, Les intervenants sur le marché boursier recourent, généralement, aux méthodes statistiques pour prévoir l'évolution des cours. Cependant, la pertinence de ces méthodes est, de plus en plus contestée. Cela est du essentiellement aux hypothèses simplificatrices, aux anomalies et à l'inefficience du marché en général. De plus, supposer la linéarité des distributions des cours, par exemple, n'a plus de sens eu égard aux résultats médiocres obtenus (Gradojevic & Young, 2000 ; Medeiros et al. 2000).

Quant aux algorithmes génétiques, ce domaine (la prévision) constitue une voie de recherche à grand potentiel. Les AGs peuvent rechercher une forme fonctionnelle, des valeurs des coefficients de régression etc. La prévision du taux de change en est l'exemple (Drake & Marks, 2002). Selon (vallée & Yildizoglu, 2001), les AGs ont été adaptés aux problèmes de séries temporelles (type de modélisation, valeur des coefficients etc.). On peut rajouter une piste de recherche, qui est proche du Data mining. Celle-ci consiste à prévoir des événements rares. La prévision de tels événements est très importante pour certaines activités de banque et d'assurance, par exemple : carte de crédit en utilisant un historique d'achats, comportements inhabituels sur un marché financier, les comportements frauduleux en assurance etc.).Enfin, la logique floue a permis la prévision de différents phénomènes à travers ce qu'on appelle variable linguistique. Elle reste l'instrument privilégié quand il s'agira de données imprécises ou incomplètes.

Les AGs sont, généralement, connus pour leur puissance en matière d'optimisation. Ils ont été exploités pour identifier les réseaux de neurones qui présentent une meilleure performance. Cela consiste à évaluer le nombre de neurones que comprend la couche cachée d'un réseau neuronal jugé performant. Plusieurs études ont traité cette question (Montagno et al, 2002). Vallée & Yildizoglu (2001) citent trois principaux succès des AGs dans le domaine de l'optimisation :

• Dans le cadre du management passif, les AGs contribuent largement à l'identification de meilleurs portefeuilles (composés d'actions, d'obligations etc.), ou à choisir un portefeuille optimal.
• Une application, c'est la capacité des AGs à chercher des règles optimales de prévision (sous forme : Si - Alors), afin de prévoir les performances futures d'un portefeuille donné. Autrement dit, l'algorithme génétique, à partir de l'évolution passée d'une valeur mobilière, pourrait prédire l'évolution future de son cours. En outre, les chercheurs recommandent l'utilisation de modèles mixtes qui améliorent nettement l'ensemble des résultats (utilisation d'un réseau de neurones avec un algorithme génétique d'optimisation, ou un système neuro-flou).
• Enfin, la découverte de règles optimale d'échanges (Trading Rules), peut être réalisée par les AGs, sur les marchés d'actions et de ch ange (Drake & Marks, 2000). Selon ces auteurs, les AGs ont donné des résultats prometteurs. En conséquence, les stratégies d'échanges élaborées ont généré un gain satisfaisant.

V.3.2 Les données de prédiction

Pour la validation de la méthodologie nous avons procédé la prédiction des taux de change quotidiens de dollar des États-Unis (USD), contre quatre autres monnaies : le Dollar canadien (CAD), le Livre britannique (GBP), le Yen japonais (JPY) et l'Euro d'une période de 8 ans à partir de 02/01/2003 jusqu'a 30/06/2011 prise de site Web: http://pacific.commerce.ubc.ca/xr [39]. Un total de 8 et six mois ans des données de taux de change, pour USD/CAD, USD/GBP, USD/EURO, USD/JPY.

V.3. 3 Description des variables des prédicteurs neuronaux

Comme tous les systèmes neuronaux mis en œuvre aujourd'hui, on utilise (N) entrées qui ne sont rien d'autre que les valeurs de taux de change retardées $x(t-\delta)$, $x(t-2.\delta)$,……., $x(t-N.\delta)$, dont (δ) est le retard. Ce prédicteur fournit à la sortie la valeur de taux de change au temps t : $x(t)$.

Les taux de change utilisés sont : USD/CAD, USD/GBP, USD/EURO, USD/JPY, chaque taux de change correspondant à un prédicteur. Les réseaux de neurones utilisés sont les perceptrons multicouches avec l'algorithme d'apprentissage retro-propagation.

V.3.4 Résultats de simulation

4 prédicteurs à base de réseaux de neurones ou tout simplement prédicteur neuronal sont utilisée pour la prédiction des quatre taux de change USD/CAD, USD/GBP, USD/EURO, USD/JPY, dans cette première application. Chaque prédicteur est constitué principalement de 3 ou 4 couches (couche d'entrée, 1ou 2 couches cachées et couche de sortie). Les prédicteurs neuronaux sont développé par le biais d'un algorithme génétique réel et cela en travaillant à minimiser l'erreur de la prédiction. L'algorithme génétique sert à optimiser l'architecture, paramètres de contrôle et l'intervalle de variation de poids des 4 prédicteurs neuronaux. Les paramètres des prédicteurs sélectionnées génétiquement sont: Le nombre de couche cachée (NHL), le nombre de neurones dans chaque couche cachées l (Nhl) et d'autres gènes représentant la forme de la fonction d'activation des couches cachées et de couche de sortie $(f^{l,} f^{out})$, le nombre d'entrée (N), la présence ou l'absence de biais (b), le temps entre deux

entrée successif retard (δ), le coefficient d'apprentissage (η), coefficient d'inertie (α), l'intervalle de variation des poids synaptique pour les couches cachées $[low_1, high_1]$, $[low_2, high_2]$ et la couche de sortie $[low_3, high_3]$. Ce travail est répété indépendamment pour les 4 prédicteurs. Pour la phase d'apprentissage on a utilisé les 2008 valeurs première de s taux de change quotidien; pour la phase de teste ou aussi dite phase de validation on a utilisée les 126 valeurs suivante. Le nombre d'itération d'apprentissage utilisée est de 1000 à 10000 itérations.

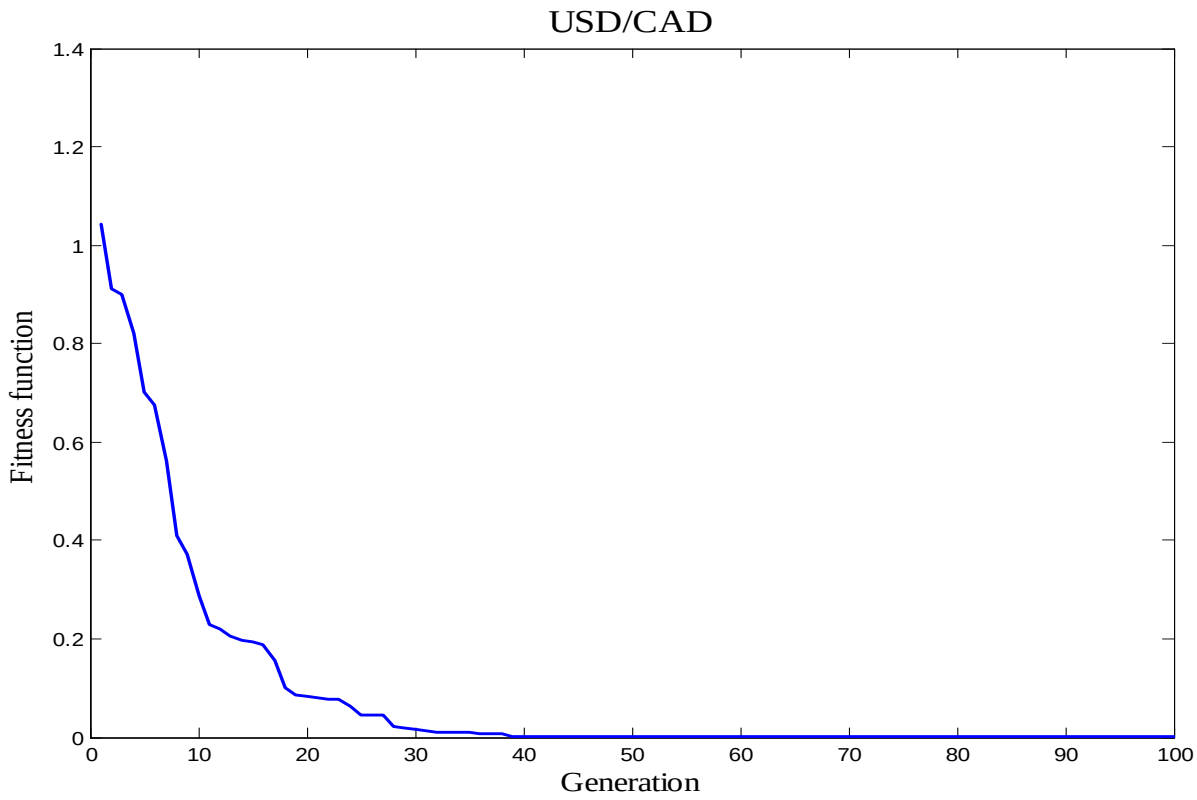


Figure V.4 Evolution de la fonction objectif à travers les générations (USD/CAD)

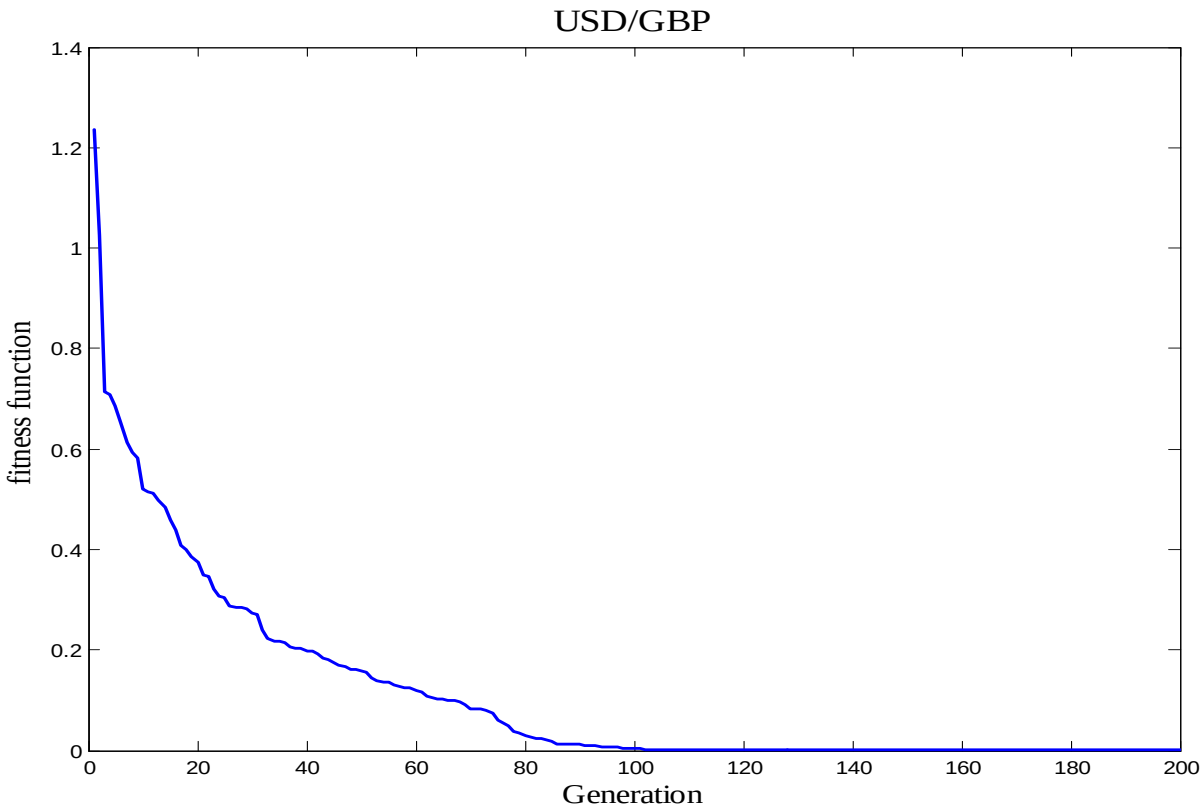


Figure V. 5 Evolution de la fonction objectif à travers les générations (USD/GBP)

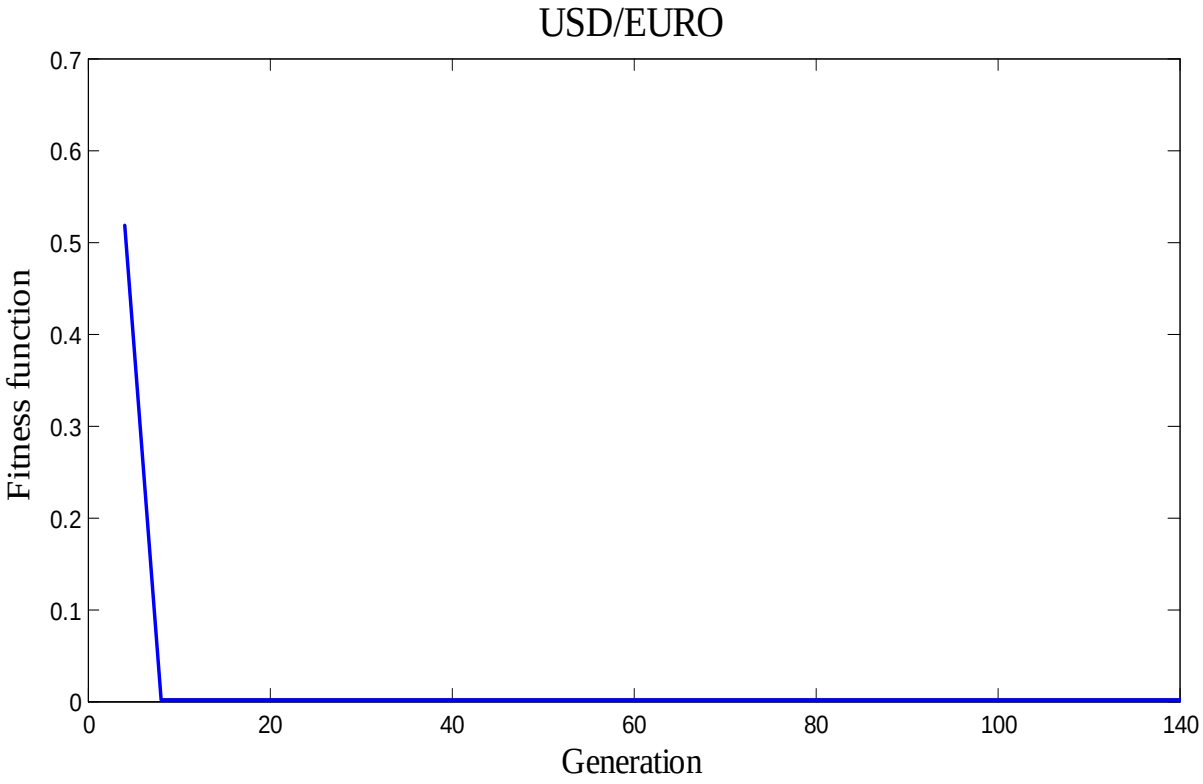


Figure V.6 Evolution de la fonction objectif à travers les générations (USD/EURO)

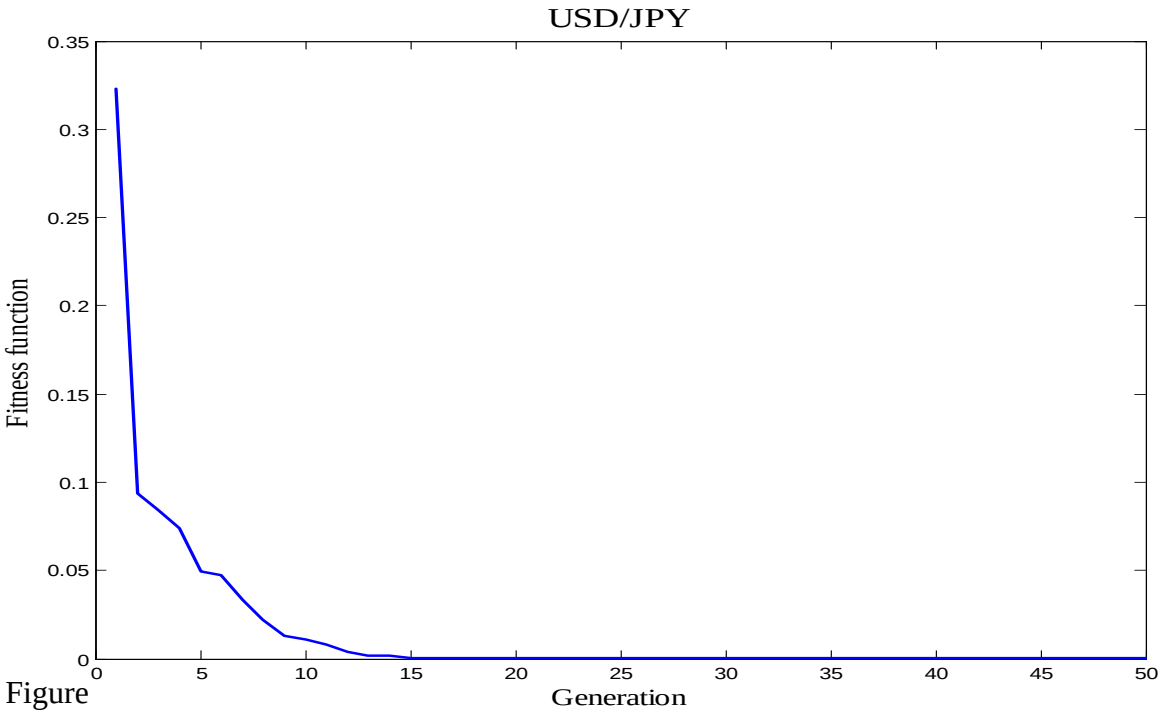


Figure V.7 Evolution de la fonction objectif à travers les générations (USD/JPY)

Les résultats obtenus pendant exécution des AG 's sont portés sur les figures V. 4, 5, 6 et 7 – *Evolution de la fonction objective à travers les générations*

Nous remarquons que les résultats diffèrent de génération en génération. La valeur du critère de performance ***fitness*** est rapidement amélioré au cours des premières générations, et est ramenée de 1.0429 à 3.487×10^{-29} au bout de 50 générations seulement pour la prédiction des taux de change USD/CAD, et de 1.2361 à 2.5739×10^{-29} au bout de 120 générations seulement pour la prédiction des taux de change USD/GBP, et de 0.5179 à 8.2092×10^{-30} au bout de 15 générations seulement pour la prédiction des taux de change USD/EURO , et de 0.3231 à 5.4972×10^{-28} au bout de 20 générations seulement pour la prédiction des taux de change USD/JPY.

L'efficacité de l'AG est ressentie dès les premières générations du processus d'optimisation, à titre d'exemple la fonction objective ***fitness*** pour le meilleure chromosome de la 58^{eme} génération pour la prédiction des taux de change USD/GBP est ramené à 0.1247 alors qu'il était initialement (première génération) de 1.2361, soit une diminution d'un rapport de 10. Les résultats sont améliorés au cours des générations et ce n'est qu'à partir de la génération 120^{ieme} qu'on atteint des performances très appréciables. Les tableaux V .3, 4, 5, 6 donnent Les nouveaux paramètres de structure des prédicteurs neuronaux correspondants aux meilleurs chromosomes obtenus à la fin d'optimisations des 4 AG's.

Paramètre de prédicteur neuronal	USD/CAD
Nombre de couche cachée	1
Coefficient d'apprentissage	0.9307
coefficient d'inertie	0.5457
la présence ou l'absence de biais	Présence (1)
le nombre d'entrée	10
Retard	4
le nombre de neurone dans couche cachée1	12
Forme de fonction d'activation de couche cachées 1	Semi- linear 3
Forme de fonction d'activation de couche de sortie	Semi- linear 3
int_{out}	[0.1634, 0.4703]
int1	[0.06360, 0.8137]

Tableau V.3 : la structure optimal reçue d'AG pour le taux de change USD/CAD

Paramètre de prédicteur neuronal	USD /GBP
Nombre de couche cachée	2
Coefficient d'apprentissage	0.3524
coefficient d'inertie	0.8858
la présence ou l'absence de biais	absence (0)
le nombre d'entrée	17
retard	1
le nombre de neurone dans couche cachée1	29
le nombre de neurone dans couche cachée2	8
Forme de fonction d'activation de couche cachées 1	Semi- linéaire 3
Forme de fonction d'activation de couche cachées 2	sigmoïdal
Forme de fonction d'activation de couche de sortie	Semi- linéaire 3
int_{out}	[0.0901, 0.3240]
int1	[0.0326, 0.563]
Int2	[0.0773, 0.6963]

Tableau V.4 : la structure optimal reçue à la fin d'AG pour le taux de change USD/GBP

Paramètre de prédicteur neuronal	USD/JPY
Nombre de couche cachée	1
Coefficient d'apprentissage	0.3475
coefficient d'inertie	0.4944
la présence ou l'absence de biais	Présence (1)
le nombre d'entrée	6
retard	6
le nombre de neurone dans couche cachée1	14
Forme de fonction d'activation de couche cachées 1	Semi- linéaire 3
Forme de fonction d'activation de couche de sortie	Semi- linéaire 3
int_{out}	[0.0240, 0.3264]
int1	[0.0097, 0.3710]

Tableau V.5 : la structure optimal reçue d'AG pour le taux de change USD/JPY

paramètres de prédicteur neuronal	USD/EURO
Nombre de couche cachée	2
Coefficient d'apprentissage	0.7578
coefficient d'inertie	0.8421
la présence ou l'absence de biais	Présence (1)
le nombre d'entrée	20
retard	1
le nombre de neurone dans couche cachée1	16
le nombre de neurone dans couche cachée2	16
Forme de fonction d'activation de couche cachées 1	Semi- linéaire 3
Forme de fonction d'activation de couche cachées 2	Semi- linéaire 3
Forme de fonction d'activation de couche de sortie	hyperbolique 2
int_{out}	0.0725, 0.2798]
int1	[0.0097, 0.3710]
Int2	[0.1106, 0.3346]

Tableau V.6 : la structure optimal reçue d'AG pour le taux de change USD/EURO

Les figures V. 8, 13, 18, 23 présentent L'évolution de MSE durant la phase d'apprentissage pour les 4 prédicteurs neuronaux. Nous remarquons que les résultats différents d'itération à autre. La valeur de MSE est amélioré au cours des premières itérations, et ramenée à 3.487×10^{-29} au bout de 4629 itérations pour la prédiction des taux de change USD/CAD, et à 2.5739×10^{-29} au bout de 4316 itérations pour la prédiction des taux de change USD/GBP, et à 8.2092×10^{-30} au bout de 8250 itérations pour la prédiction des taux de change USD/EURO, et à 5.4972×10^{-28} au bout de 1419 itérations pour la prédiction des taux de change USD/JPY.
Les figures V. 9, 14, 19, 24 présentent les taux de change quotidiens USD/CAD, USD/GBP, USD/EURO, USD/JPY actuels et prédits généré à la fin de phase d'apprentissage.
Les figures V. 10, 15, 20, 25 présentent l'erreur de prédiction générée à la fin de phase de d'apprentissage pour les 4 prédicteurs neuronaux. Nous remarquons que l'erreur de prédiction est inférieur à 3×10^{-14} pour le prédicteur neuronal USD/CAD, et inférieur à 4.5×10^{-13} pour le prédicteur neuronal USD/GBP, et inférieur à 2.5×10^{-14} pour le prédicteur neuronal USD/EURO, et inférieur à 2.5×10^{-11} pour le prédicteur neuronal USD/JPY. On constate que les valeurs de l'erreur de prédiction sont très faibles est ces résultats sont très satisfaisantes.
Les figures V. 11, 16, 21, 26 présentent les taux de change quotidiens USD/CAD, USD/GBP, USD/EURO, USD/JPY actuels et prédits issue de la phase de test.
es figures V. 12, 17, 24, 27 présentent l'erreur de prédiction issue de la phase de test pour les 4 prédicteurs neuronaux. Nous remarquons que l'erreur de prédiction est inférieur à 5×10^{-12} pour le prédicteur neuronal USD/CAD, et inférieur à 3.5×10^{-14} pour le prédicteur neuronal USD/GBP, et inférieur à 3×10^{-15} pour le prédicteur neuronal USD/EURO, et inférieur à 3.5×10^{-13} pour le prédicteur neuronal USD/JPY. Comme notre objectif est d'obtenir un prédicteur neuronal avec une très faible erreur de prédiction les résultats de simulations ont confirmé l'intérêt de la méthode.

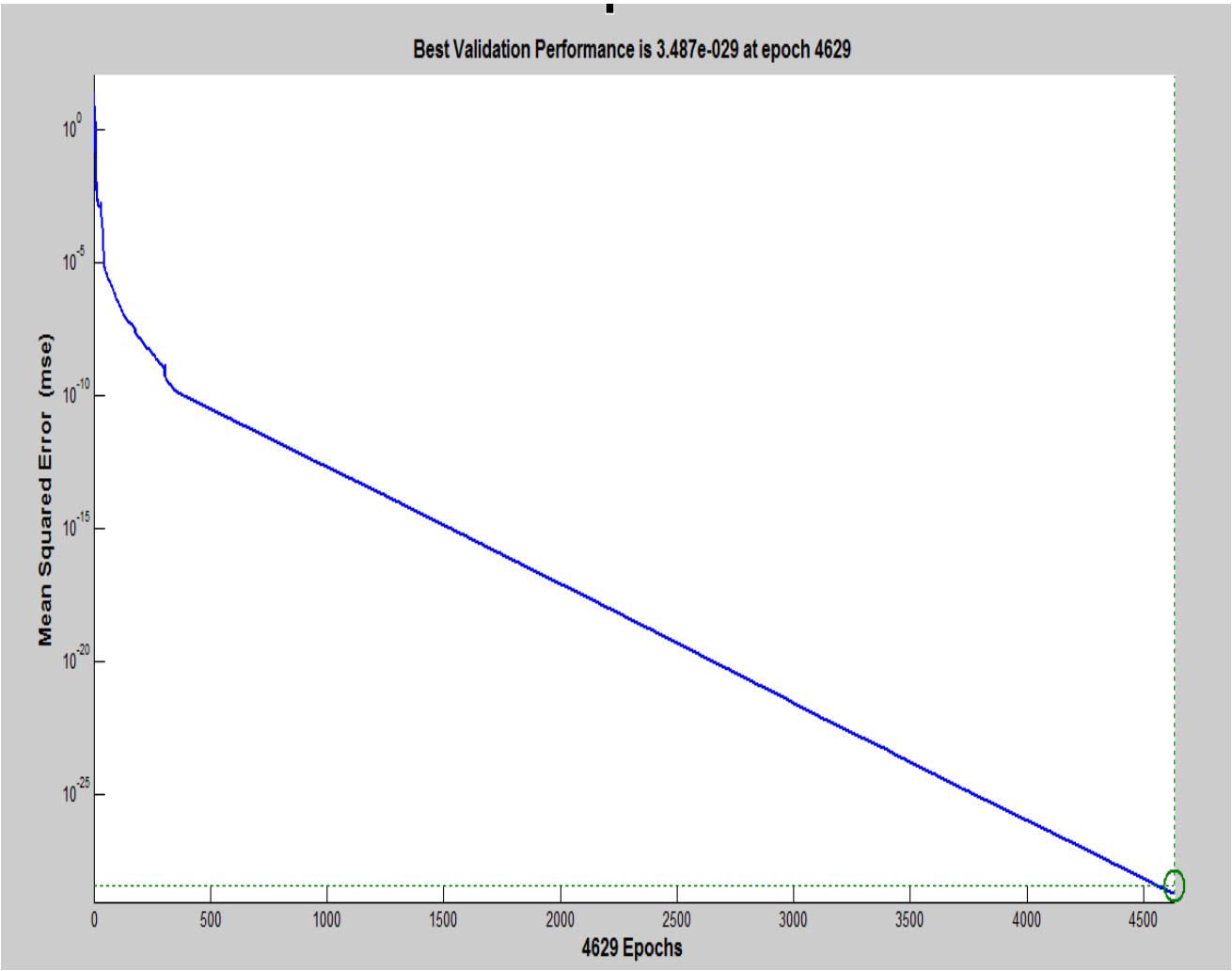


Figure V.8 Evolution de MSE durant la phase d'apprentissage (USD/CAD)

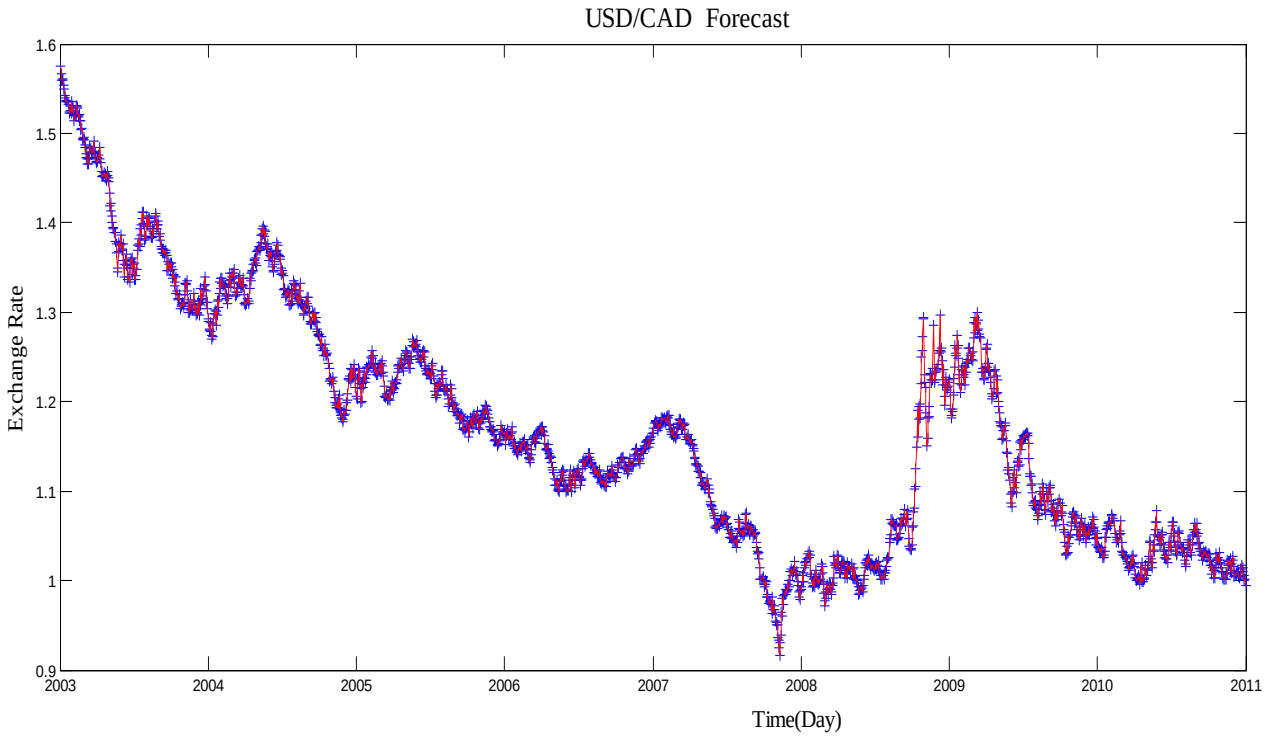


Figure V. 9 Le taux de change quotidien actuel et prédit généré à la fin de phase d'apprentissage (USD/CAD)

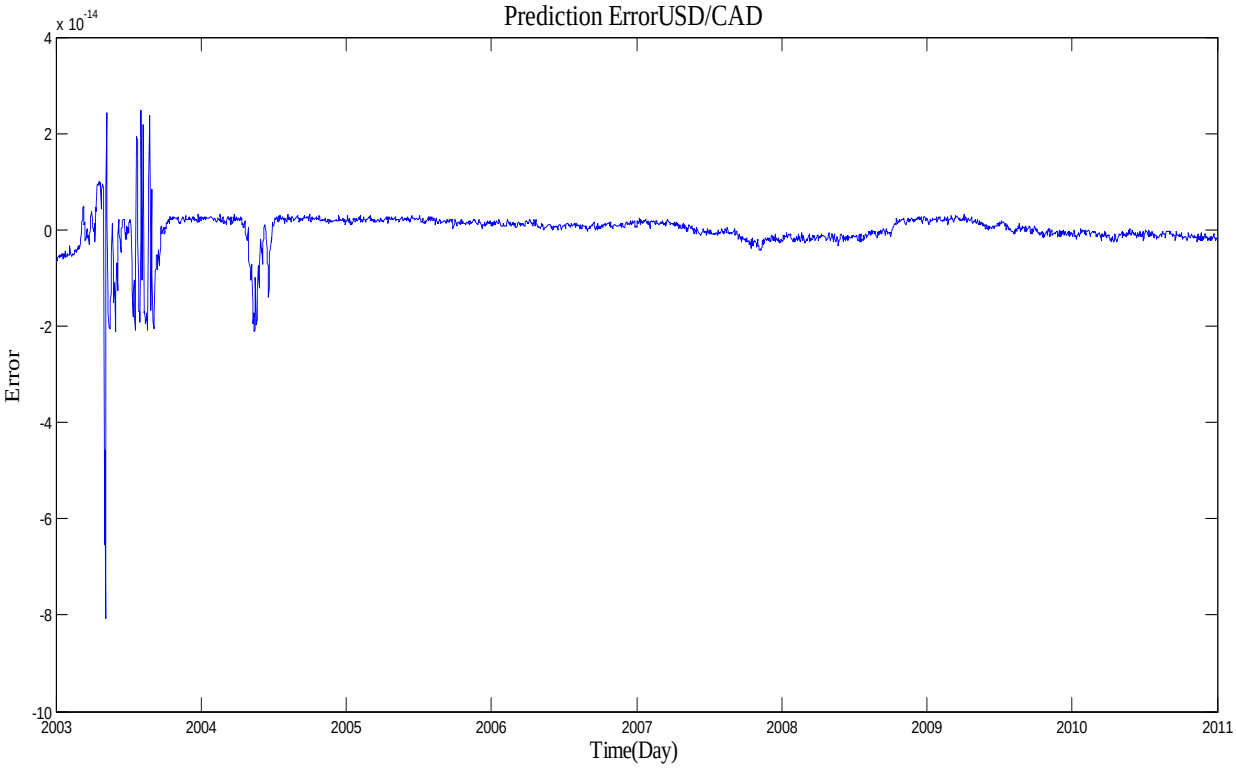


Figure V.10 Erreur de prédiction générée à la fin de phase de d'apprentissage (USD/CAD)

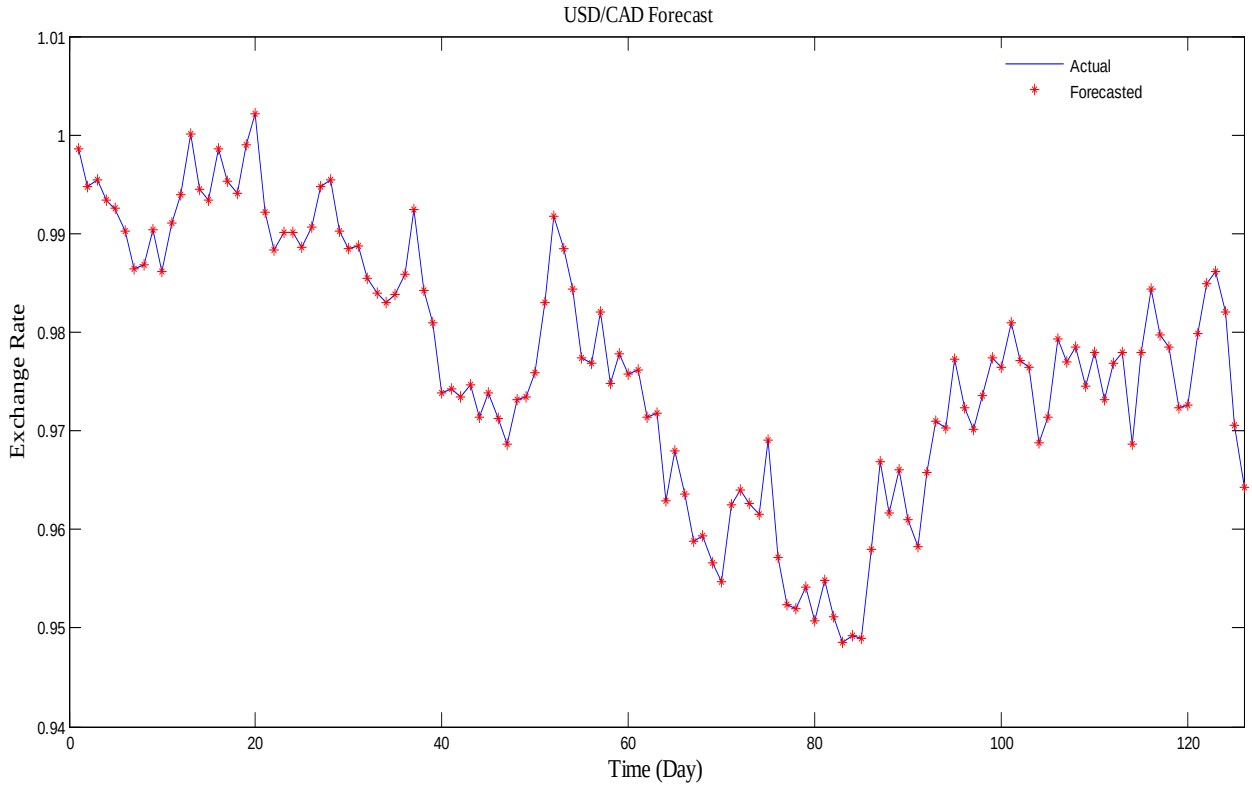


Figure V.11 Le taux de change quotidien actuel et prédit de la phase de test (USD/CAD)

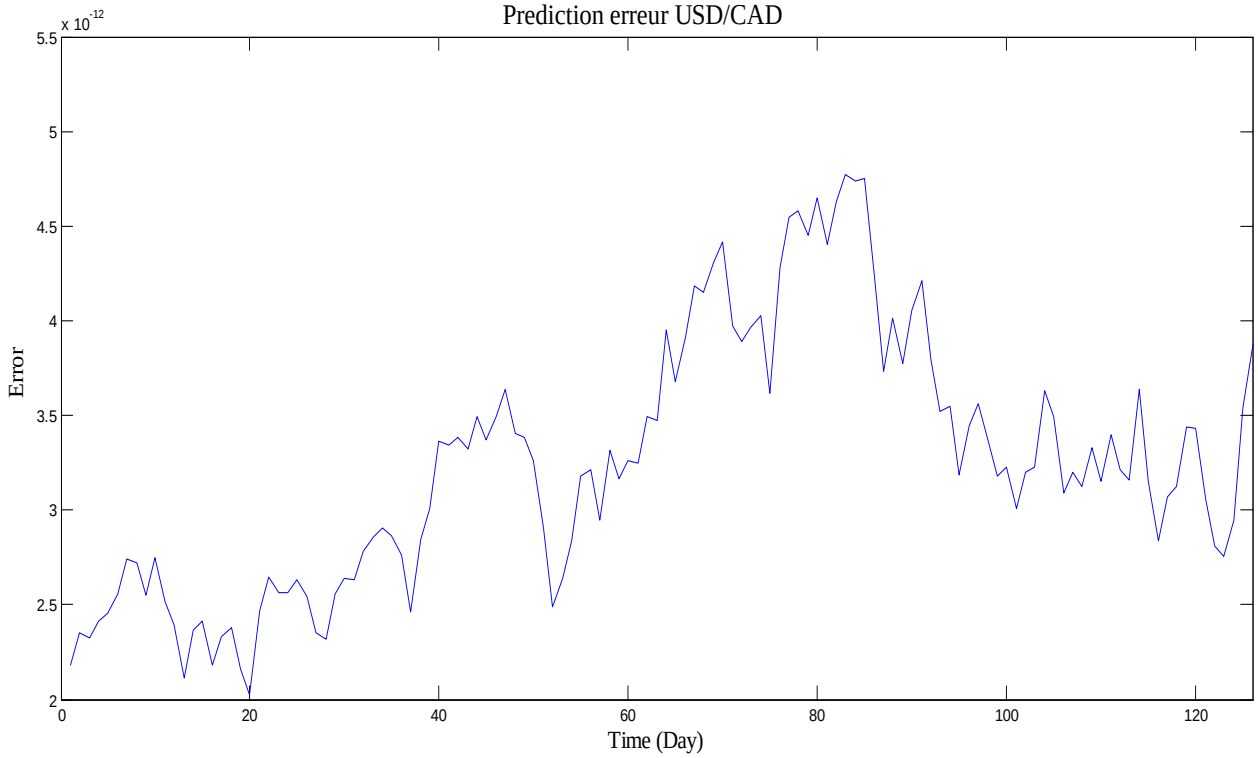


Figure V.12 Erreur de prédiction issue de la phase de test (USD/CAD)

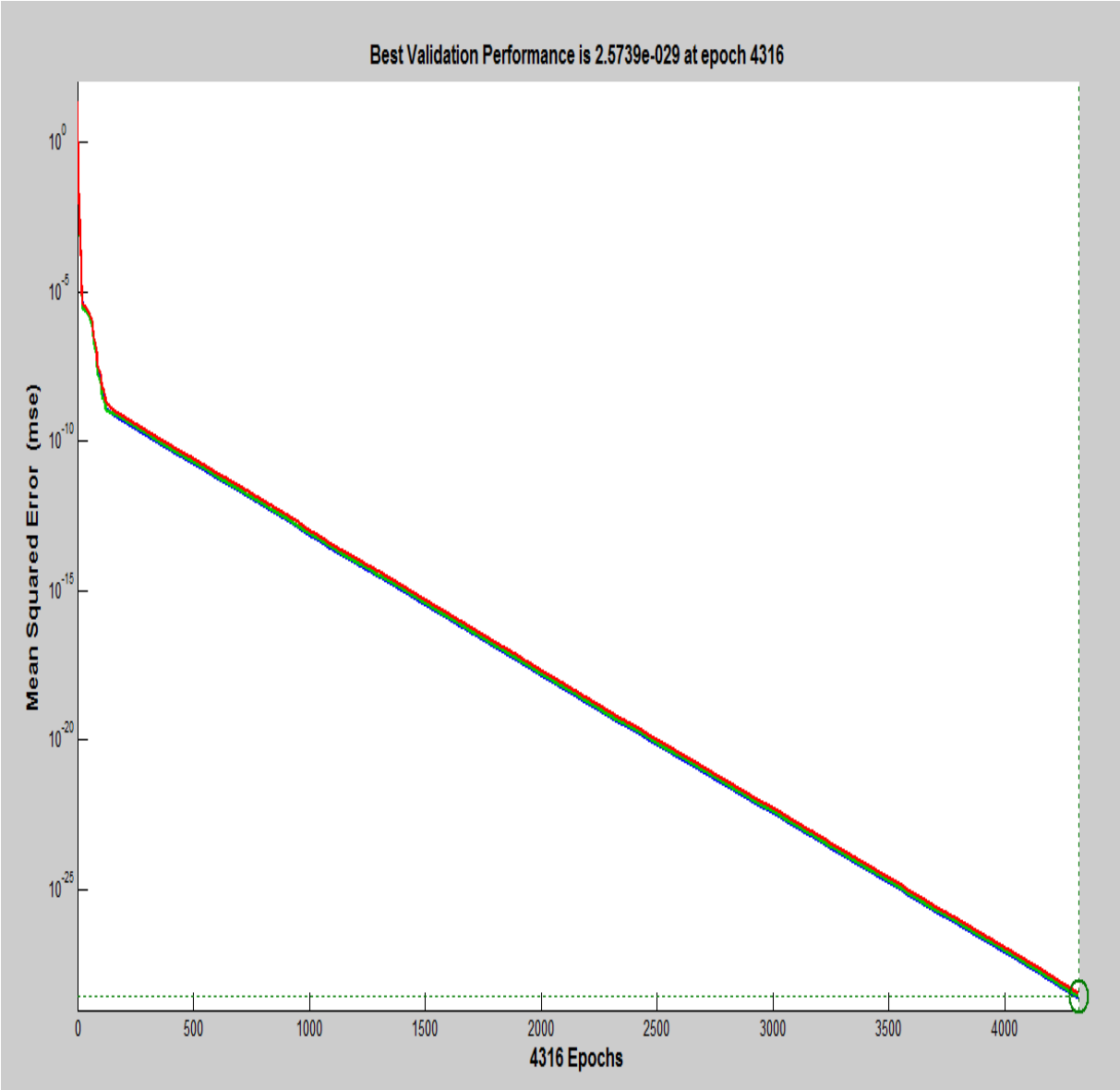


Figure V.13 Evolution de MSE durant la phase d'apprentissage (USD/GBP)

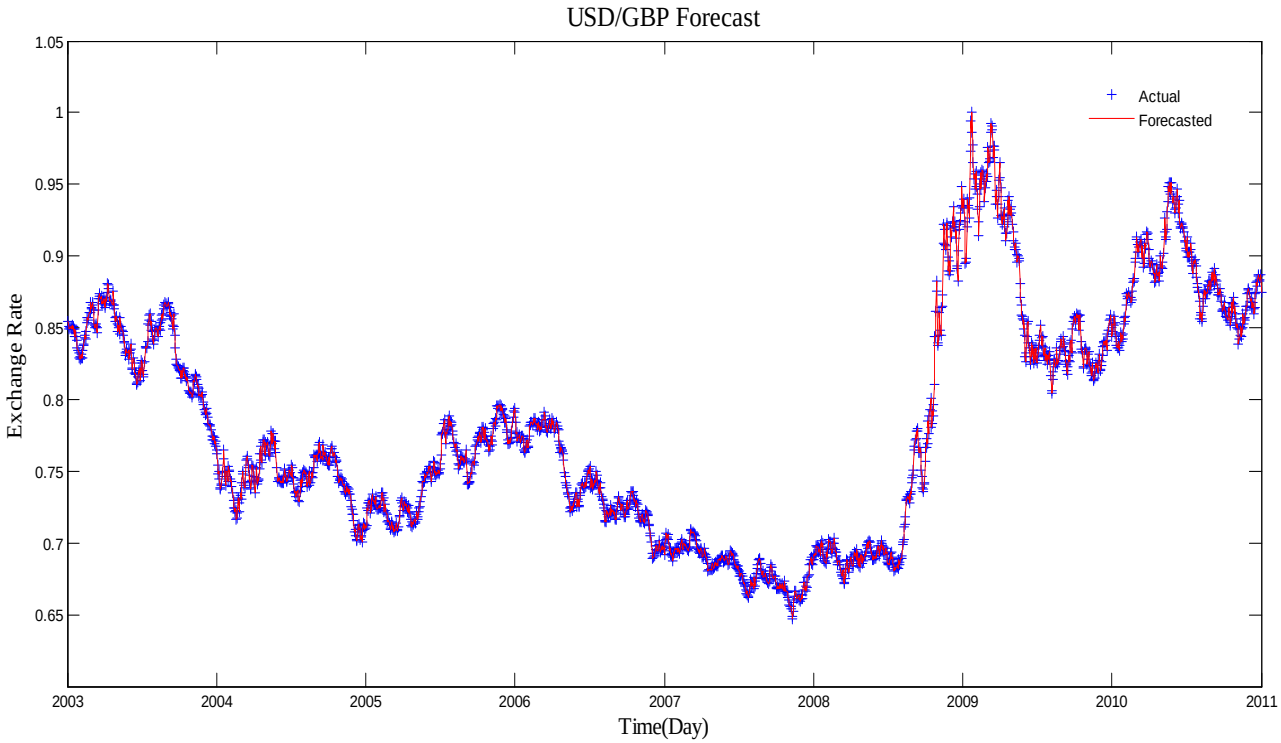


Figure V.14 Le taux de change quotidien actuel et prédit généré à la fin de phase d'apprentissage (USD/GBP)

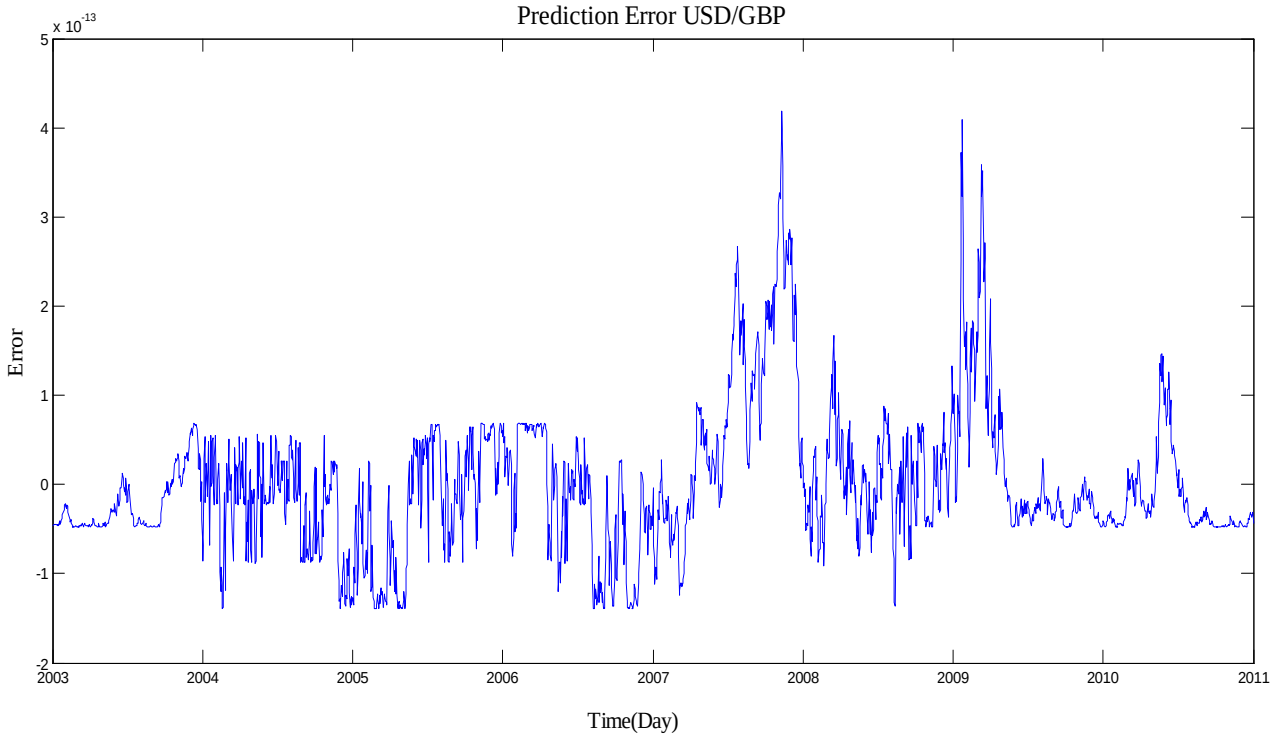


Figure V.15 Erreur de prédiction pour générée à la fin de phase de d'apprentissage (USD /GBP)

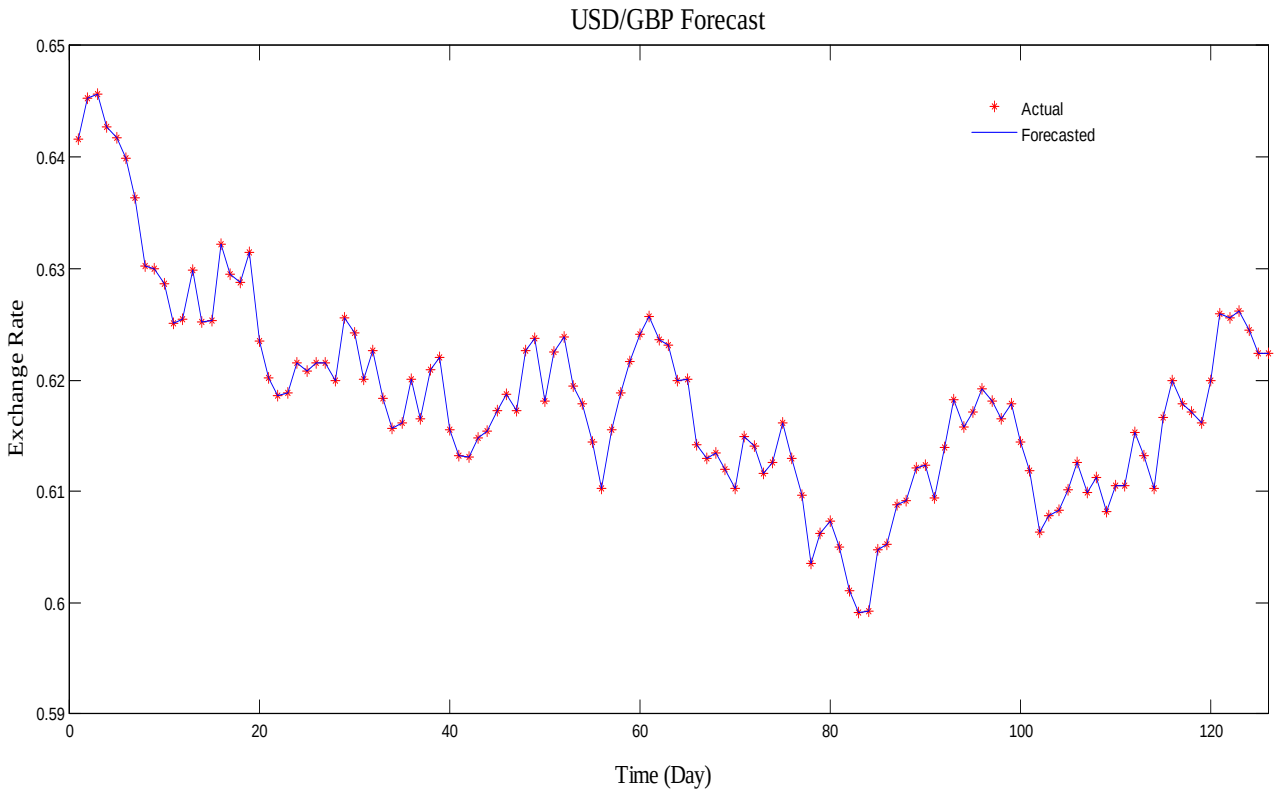


Figure V.16 Le taux du change quotidien issu de la phase de test (USD/GB)

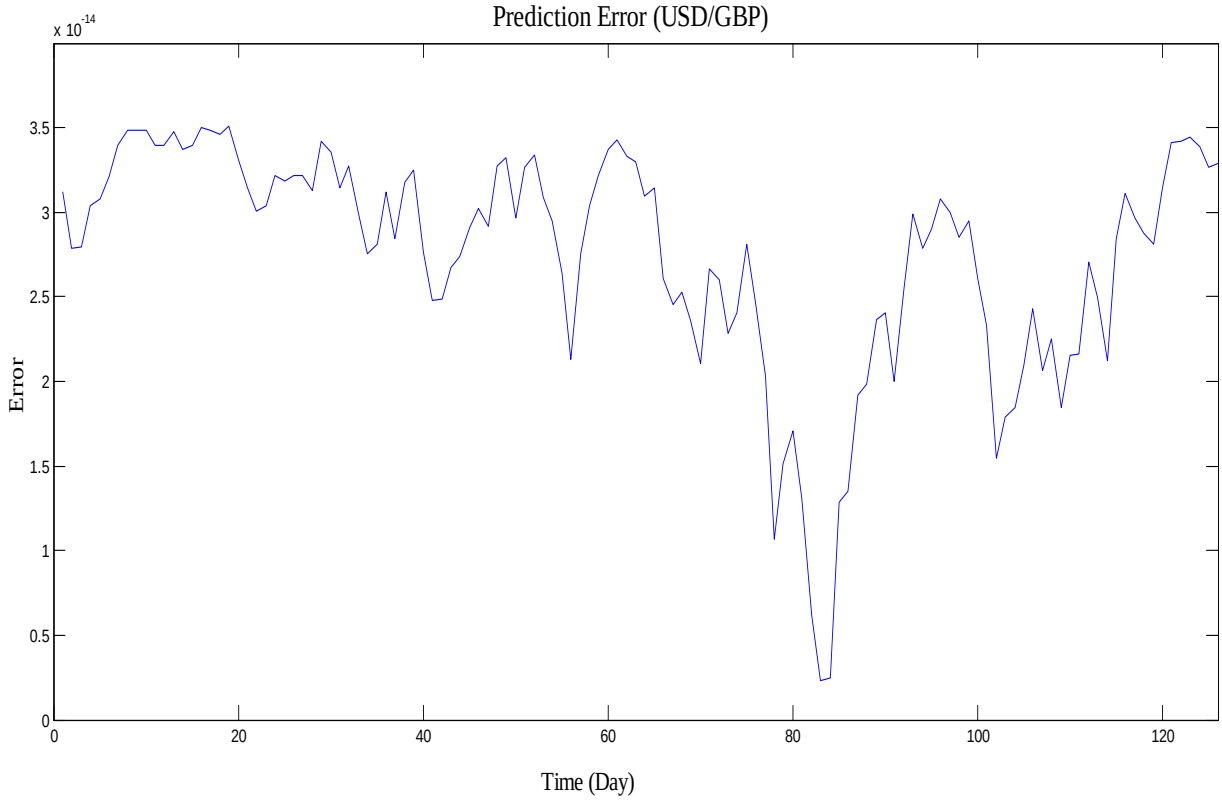


Figure V.17 Erreur de prédiction issue de la phase de test (USD/GBP)

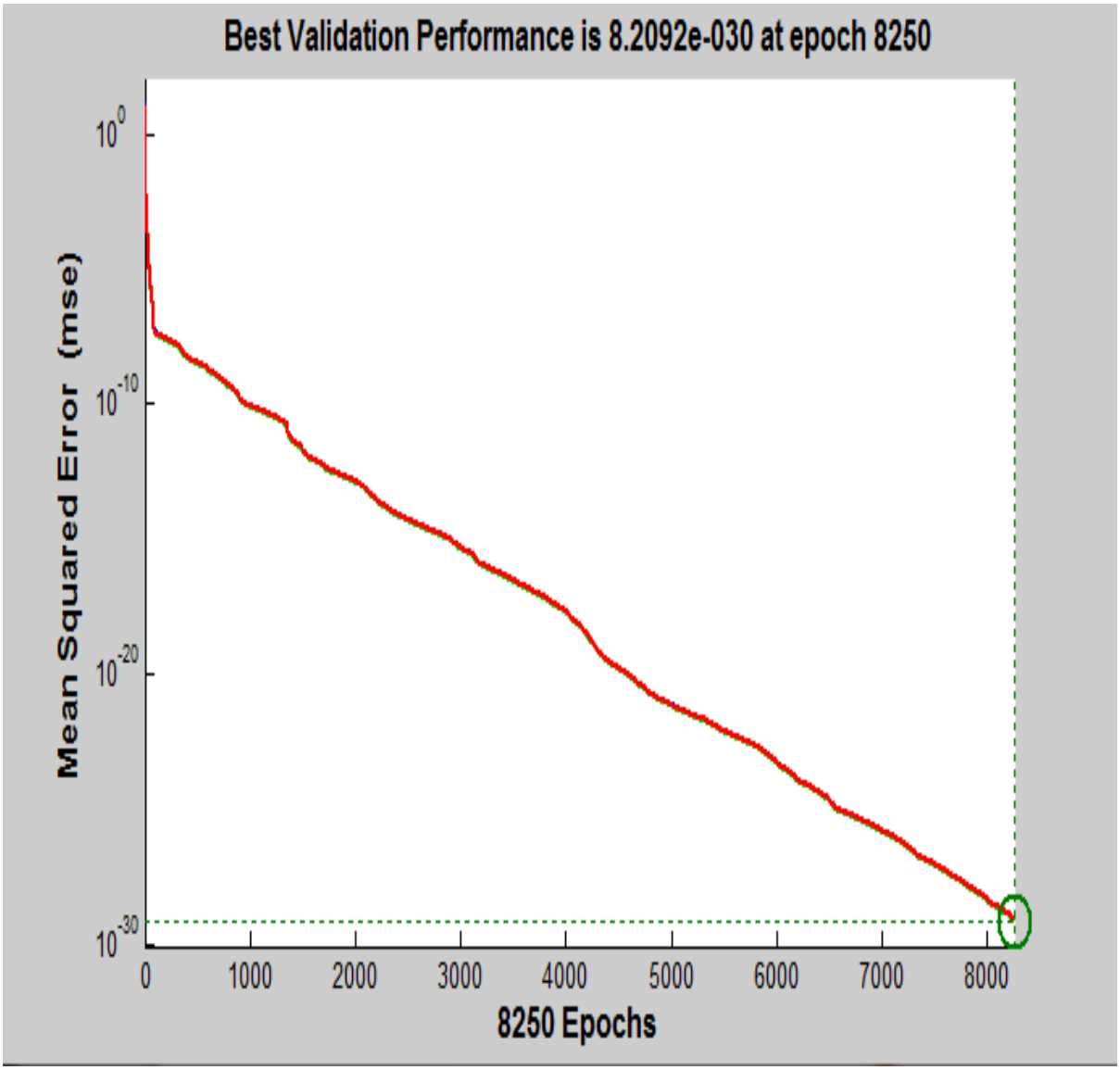


Figure V.18 L'évolution de MSE durant la phase d'apprentissage (USD/EURO)

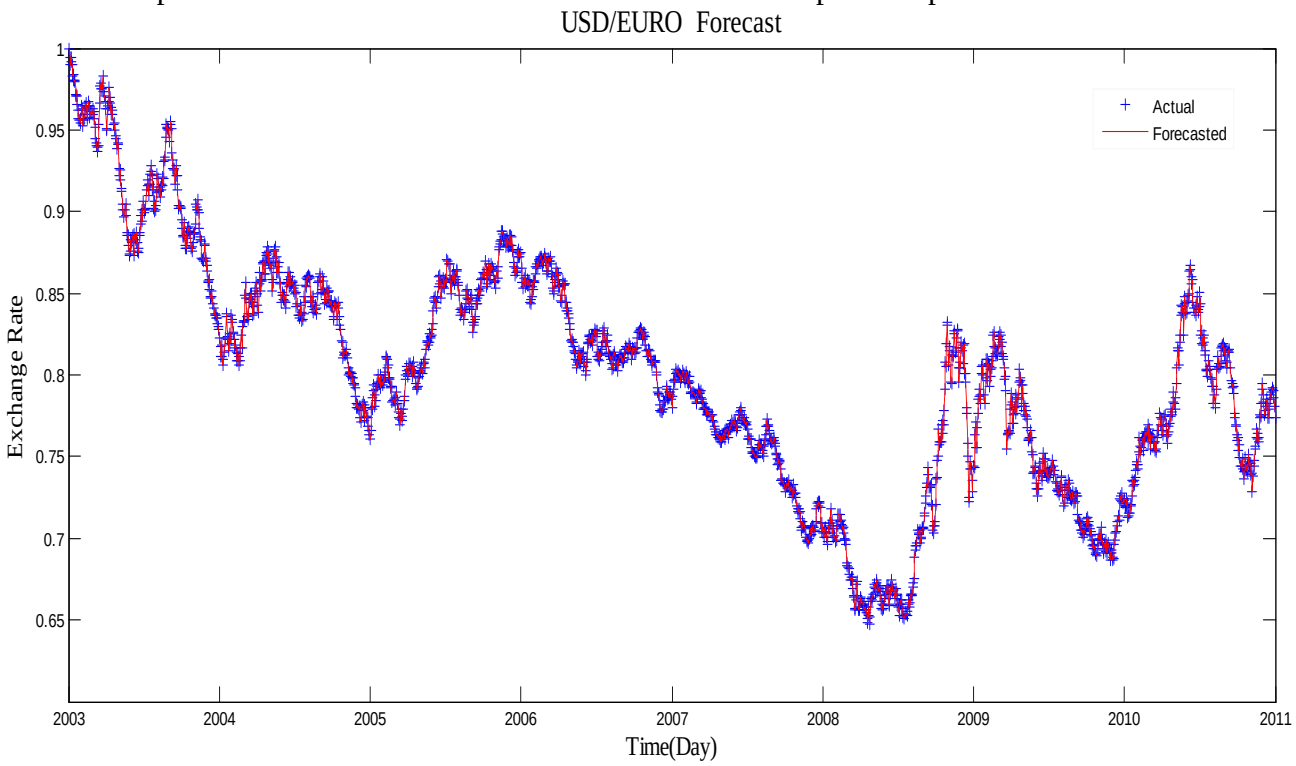


Figure V.19 Le taux de change quotidien actuel et prédit généré à la fin de phase d'apprentissage (USD/EURO)

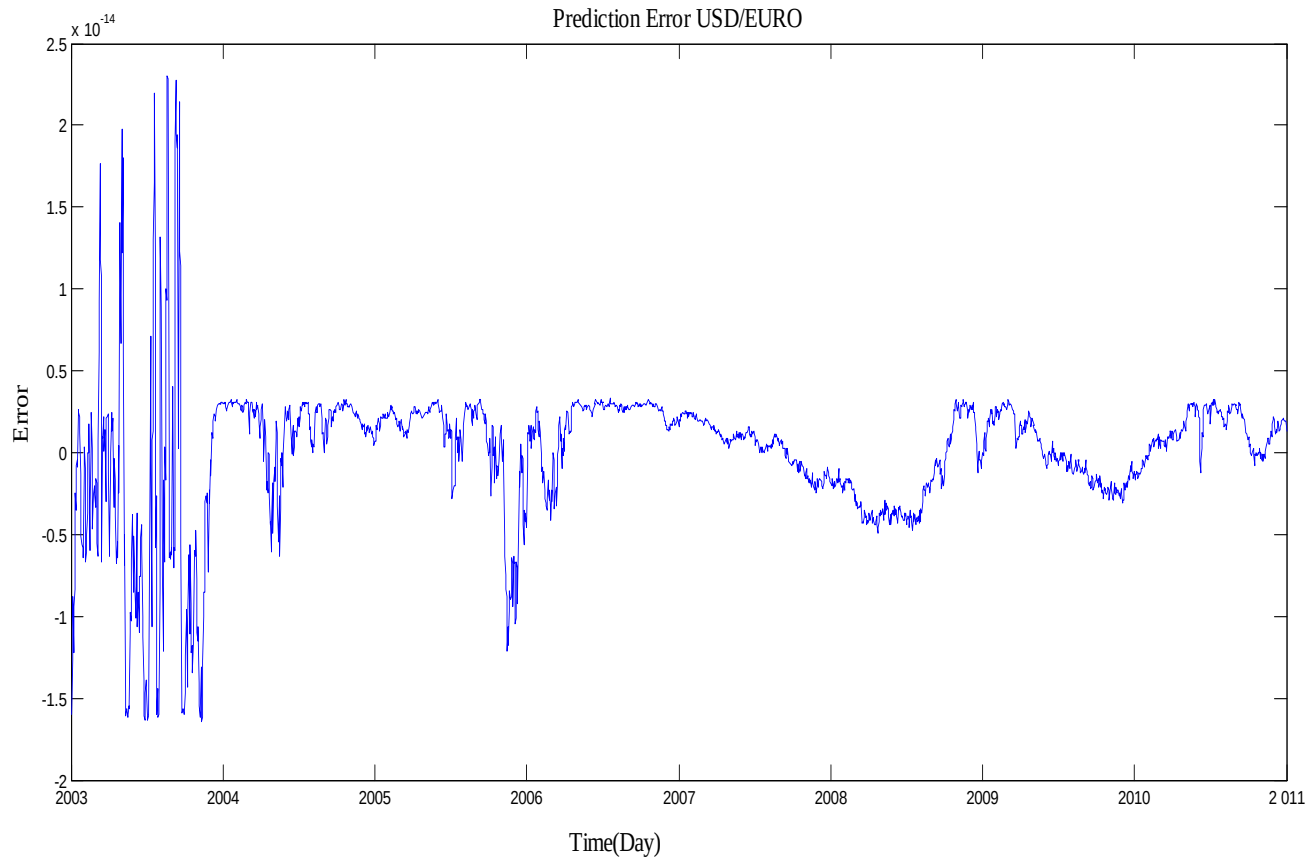


Figure V. 20 Erreur de prédiction générée a la fin de phase de d'apprentissage (USD/EURO)

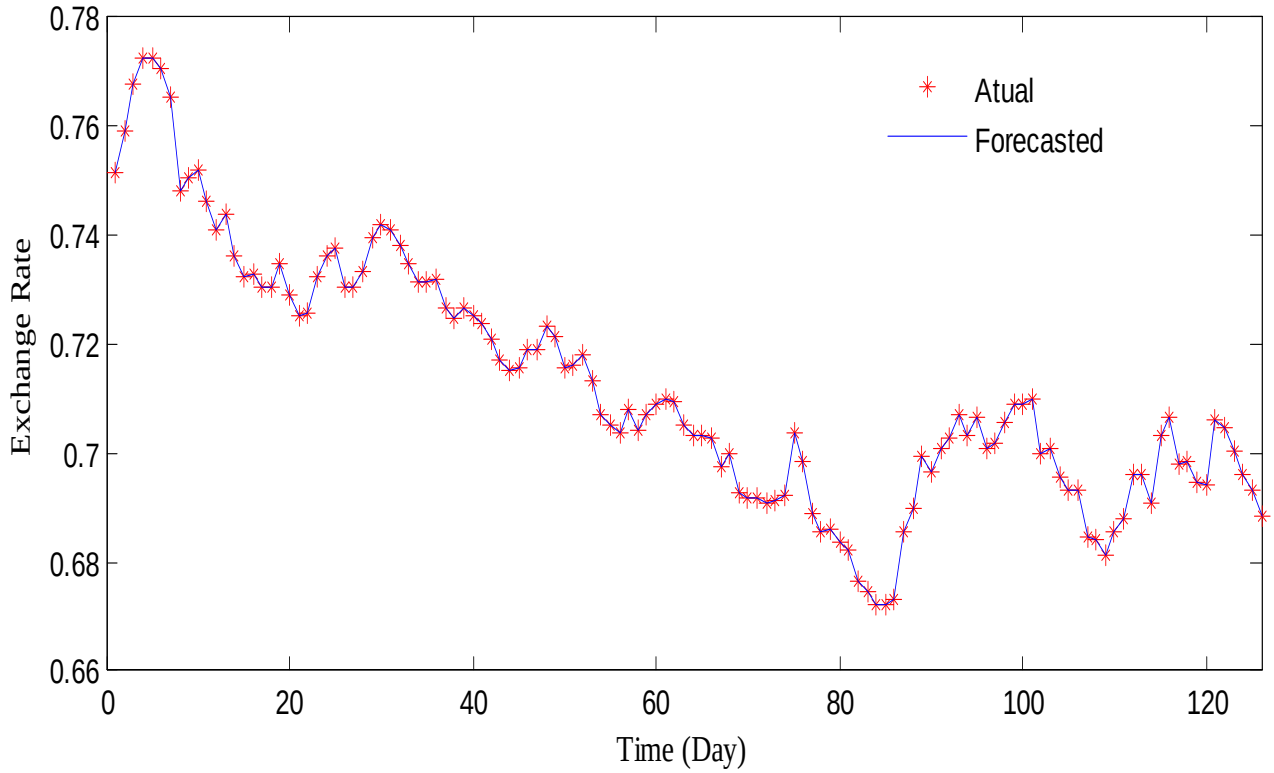


Figure V. 21 Le taux de change quotidien actuel et prédit généré de la phase de test (USD/EURO)

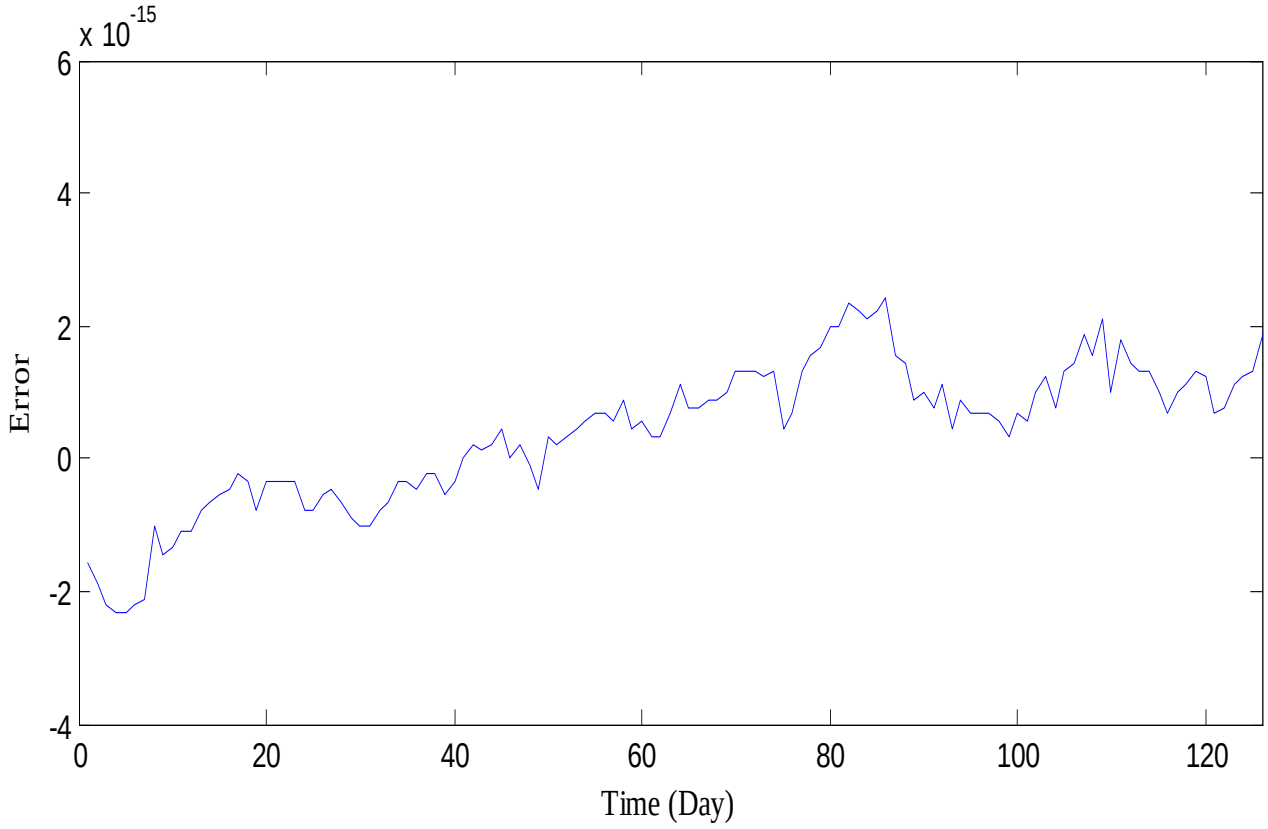


Figure V. 22 Erreur de prédiction issue de la phase de test (USD/EURO)

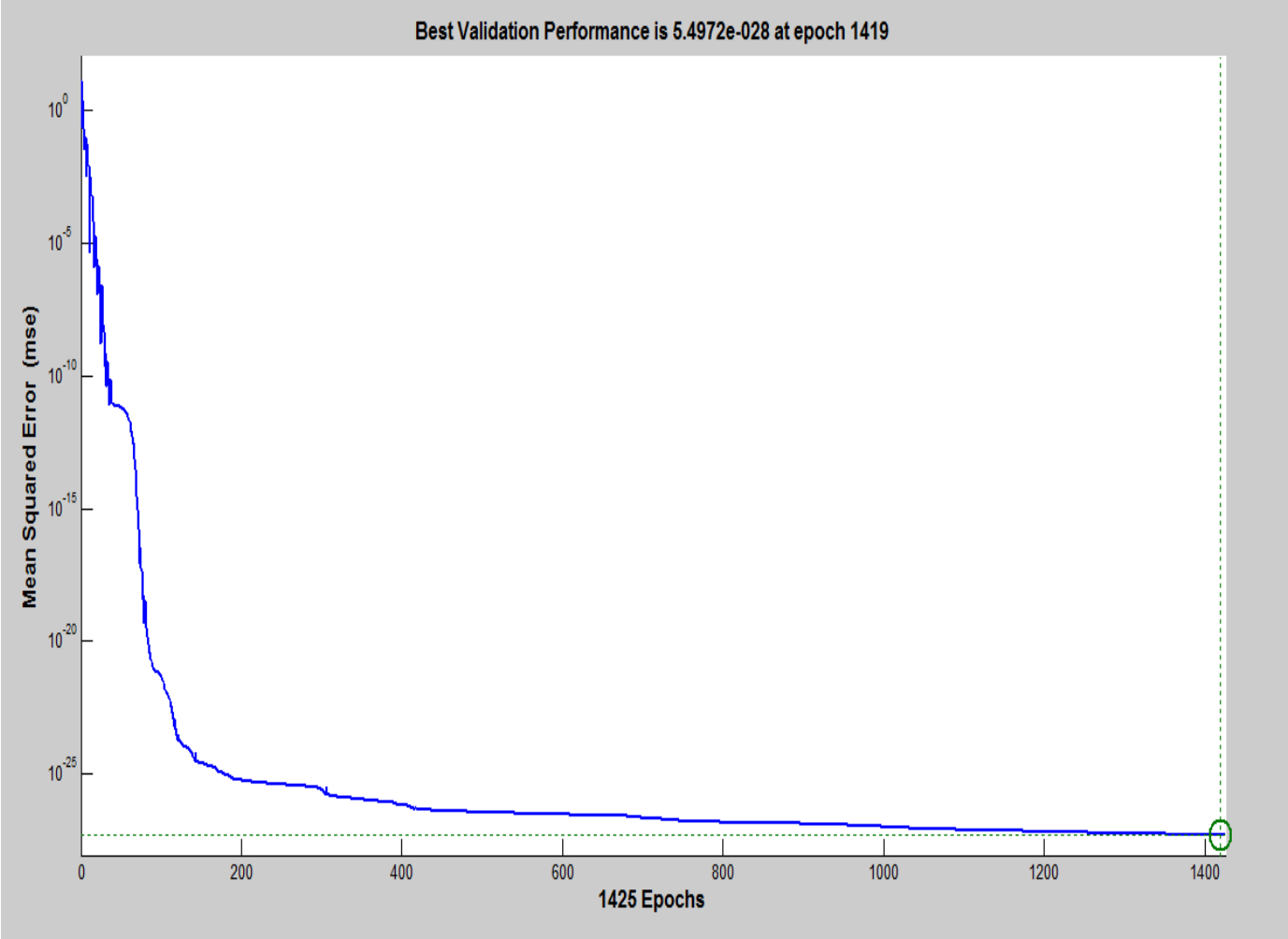


Figure V. 23 Evolution de MSE durant la phase d'apprentissage USD/JPY

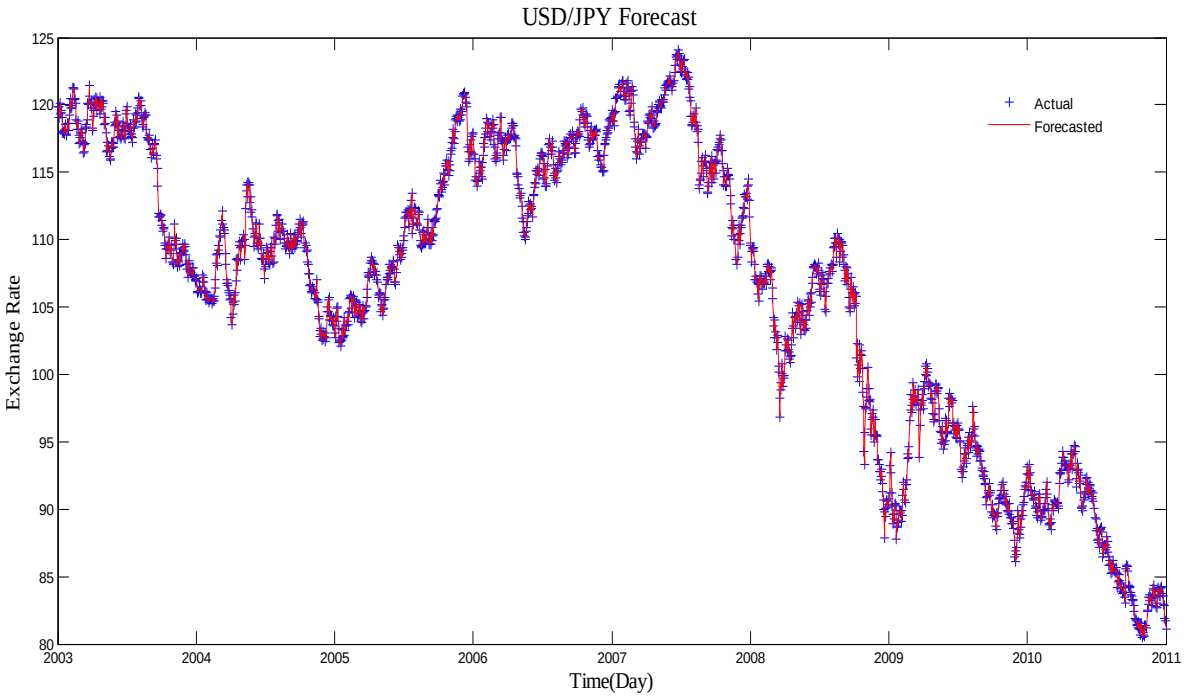


Figure V. 24 Le taux de change quotidien actuel et prédit généré à la fin de phase d'apprentissage USD /JPY

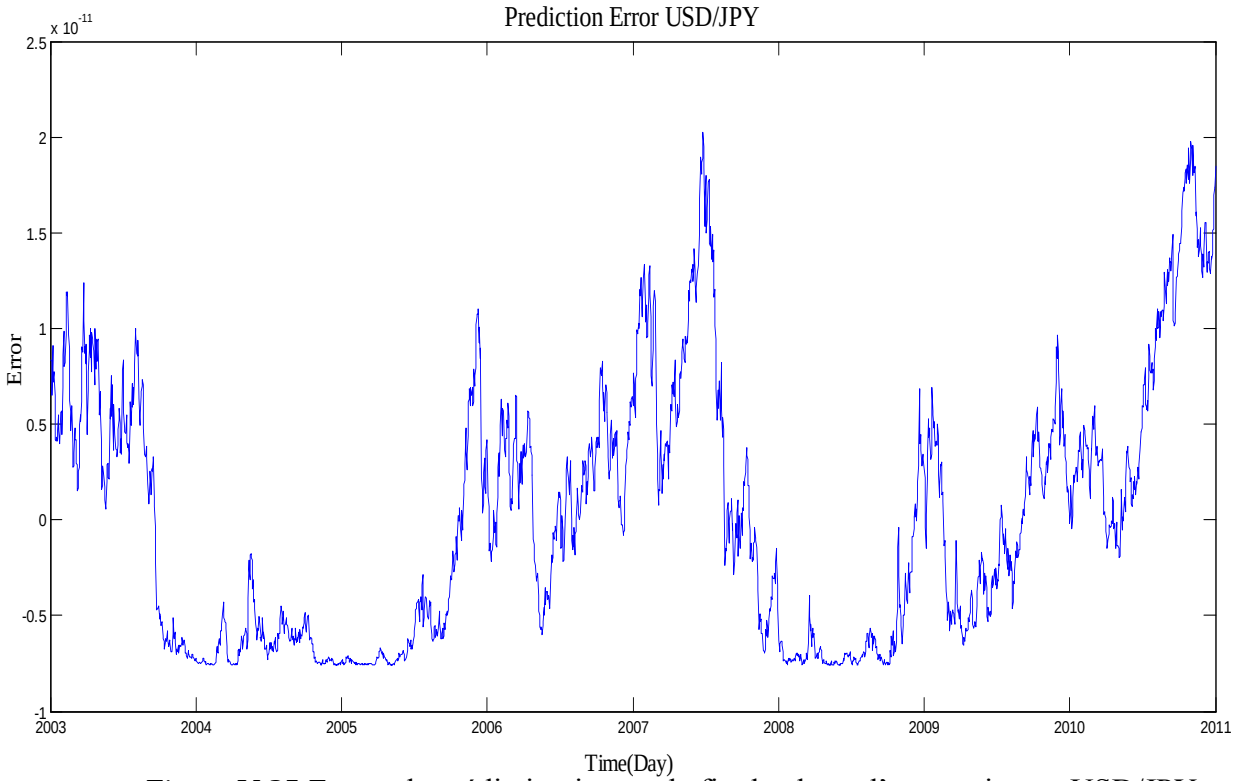


Figure V.25 Erreur de prédiction issue a la fin de phase d'apprentissage USD/JPY

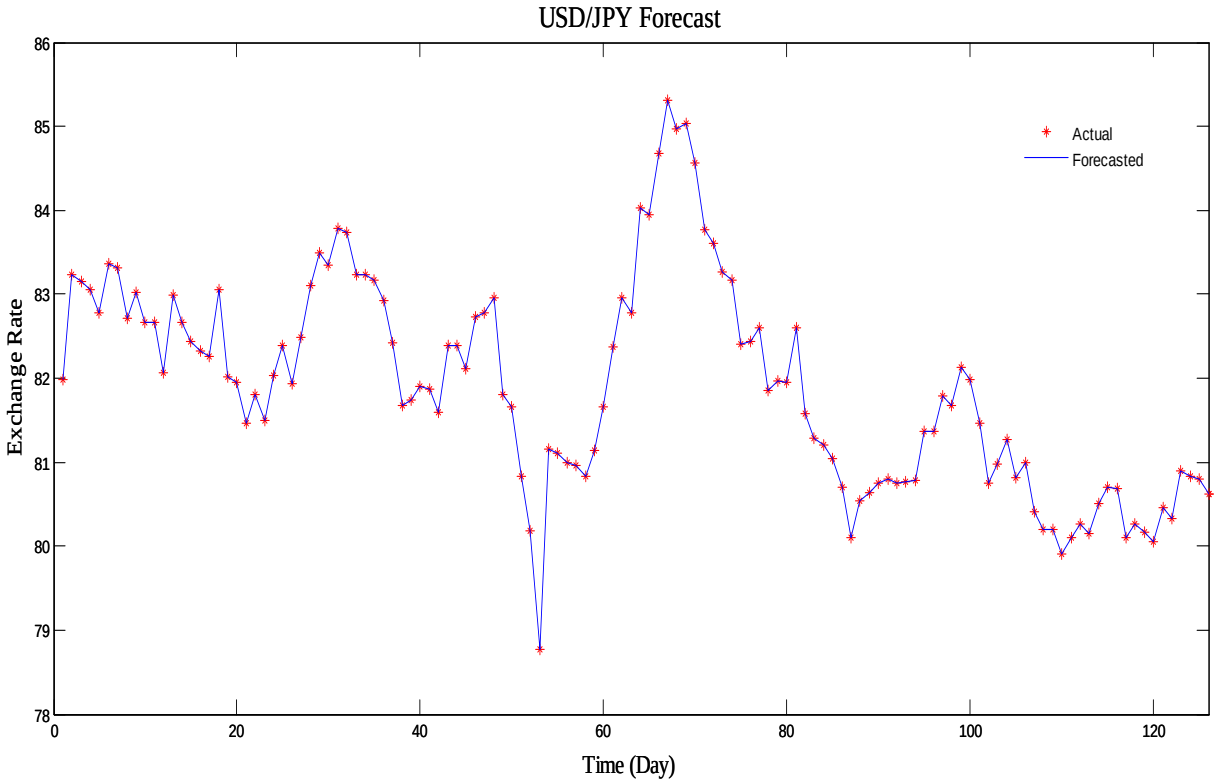


Figure V.26 Le taux de change quotidien actuel et prédit de la phase de test (USD/JPY)

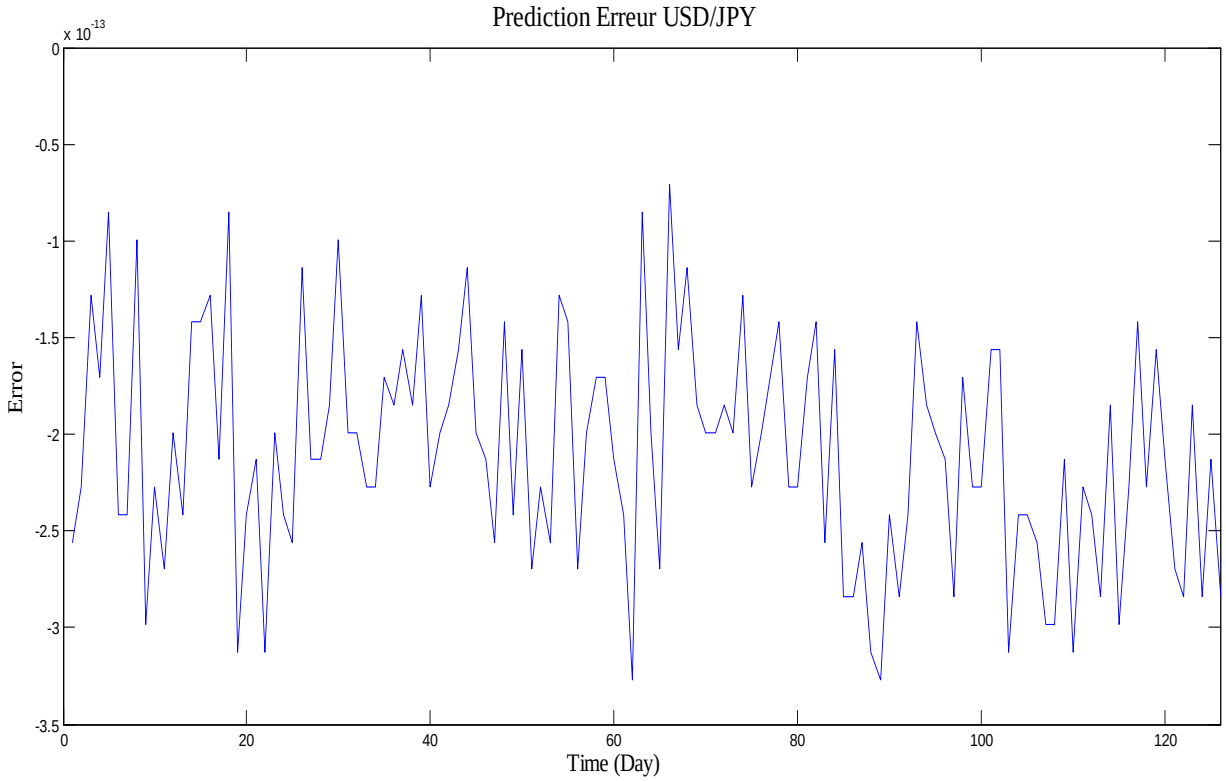


Figure V.27 Erreur de prédiction issue a la fin de phase d'apprentissage USD/JPY

On constate que les paramètres de structure de prédicteur neuronal sont tous a fait différents de ceux obtenus au début du processus d'optimisation. Cela se justifie par le rôle important que joue les opérateurs génétiques en explorant de nouvelles solutions dans l'espace de recherche.

V.3.4. Comparaison avec d'autres méthodes

Pour évaluer numériquement nos performances de prévision, nous avons utilisé deux indicateurs de performance. Il existe une règle simple pour savoir si le modèle développé est utile à la prévision. Il s'agit de comparer les résultats avec d'autre méthodes dans cette application avec les modèles héteroscedastique conditionnels (GARCH, GARCH-M, EGARCH, TGARCH/GJR et IGARCH)[52] et les réseaux de neurones [52] (Tableau V. 7, 8, 9, 10). D'après ces résultats on constate que le prédicteur hybride RNA/AG à des pouvoir prédictif supérieur à celle des modèles héteroscedastique conditionnels (GARCH, le GARCH-M, EGARCH, TGARCH/GJR et IGARCH) et des réseaux de neurones en terme de deux indicateurs de performances (NMSE, MAE) pour les 4 taux de change, l'indicateurs de performances NMSE de la méthode proposée vaut au minimum 10^4 fois NMSE des modèles héteroscedastique conditionnels ainsi que les réseaux de neurones. La raison de la qualité de prévision fournit par cette méthode est l'efficacité des AG's à chercher les meilleurs paramètres de réseaux de neurones: l'architecture, des paramètres de contrôle, et d'intervalle de variation de poids qui convient avec les valeurs de séries temporelles à prédire.

Model	NMSE	$MAE.10^2$
GARCH(1,1)	1.00011	0.27177
GARCH(1,1)-M	0.86999	0.26832
EGARCH(1,1)	0.89286	0.32988
TGARCH/GJR(1,1)	0.87995	0.26164
IGARCH(1,1)	0.80463	0.27175
Réseaux de neurones	0.56853	0.2274
Méthode proposé [40]	**$6.0868.10^{-4}$**	**$1.7504.10^{-5}$**

Tableau V.7 USD/GBP

Model	**NMSE**	**$MAE.10^2$**
GARCH(1,1)	0.99998	0.34930
GARCH(1,1)-M	0.93987	0.34918
EGARCH(1,1)	0.89833	0.30927
TGARCH/GJR(1,1)	0.89986	0.31414
IGARCH(1,1)	0.92064	0.34999
Réseaux de neurones	0.88442	0.38644

Méthode proposé [40]	**7.1053 10^{-5}**	**5.5898.10^{-6}**

Tableau V. 8 USD/EURO

Model	**NMSE**	**MAE.10^{2}**
GARCH(1,1)	0.89899	0.36216
GARCH(1,1)-M	0.84957	0.36010
EGARCH(1,1)	0.79654	0.32844
TGARCH/GJR(1,1)	0.80971	0.32207
IGARCH(1,1)	0.82374	0.36236
Réseaux de neurones	0.74021	0.37706
Méthode proposé [40]	**9.8296.10^{-4}**	**1.3284.10^{-5}**

Tableau V. 9: USD/ JPY

Model	**NMSE**	**MAE.10^{2}**
Méthode proposé [40]	**1.5483.10^{-3}**	**2.0568.10^{-4}**

Tableau V.10 USD/CAD

V.4. Application dans le domaine d'écologique : Pollution d'air

La prévision des situations de pollution atmosphérique est une nécessité de notre société. En effet, ces situations, sont d'année en année passées d'un stade accidentel à un problème fréquent de santé publique. A ces considérations de santé publique, il faut ajouter leurs influences sur le climat [66].

Les humains sont exposés à la pollution atmosphérique depuis des temps immémoriaux parce que le feu, une source de pollution de première importance, a été utilisé pratiquement partout dans le passé pour la cuisson et le chauffage. La majorité des polluants atmosphériques sont attribuables à la combustion de combustibles fossiles dans les véhicules à moteur, les usines, les centrales thermiques et les appareils des chauffages des maisons. Parmi les polluants atmosphériques, on distingue les polluants primaires et les polluants secondaires. Les polluants primaires sont rejetés directement dans l'air par des sources précises, comme l'industrie ou les véhicules à moteur. Certains polluants primaires peuvent être altérés par la lumière du soleil, la chaleur ou d'autres substances chimiques et forment alors des polluants secondaires. Les polluants de l'air peuvent être sous formes solide (particules et fibres), liquide ou gazeuse, ou encore sous forme de vapeur. Les principaux polluants comprennent des particules de matière (PM) et certains gaz. Les oxydes d'azote (NO_X) sont un groupe de gaz extrêmement réactifs formés lorsqu'on brûle du combustible à des températures très élevées. Les composés organiques volatils (COV) sont des composés ayant une pression de vapeur élevée et une pression d'eau basse. Il s'agit habituellement de solvants industriels. Les oxydes d'azote et les COV sont des composés organiques qui se transforment en vapeur ou en gaz sans réaction chimique et constituent des gaz précurseurs clés qui réagissent avec d'autres gaz en présence des rayons du soleil pour former l'ozone. L'ozone est un exemple de polluant secondaire et un des principaux éléments du smog qui, à son tour, est un type important de pollution atmosphérique[66].

Les polluants atmosphériques gazeux les plus étudiés sont l'ozone O_3, le dioxyde de soufre ou anhydride sulfureux SO_2, les oxydes d'azote (monoxyde d'azote NO et dioxyde d'azote NO_2 et le monoxyde de carbone CO. Parce que le O_3 et le NO_2 sont moins solubles que les autres gaz irritants, ils peuvent pénétrer plus profondément dans les poumons jusqu'à des endroits où ils causent, de l'inflammation et de l'œdème. Les polluants ayant des propriétés cancérogènes comprennent ozone, le benzène, le butadiène, le chloroforme, le chrome, d'autres métaux ……

Les différentes substances ont des durées de vie dans l'atmosphère qui sont excrément variables, ce qui explique que les problèmes de pollution se situent sur des différentes échelles de temps et d'espace très variables. Voici l'ordre de grandeur de la durée de vie dans l'atmosphère de quelques polluants [66] :

Polluants	Durée de vie
CH4	année
CO	mois
SO2	jours à mois
Ozone	qq jours

Tableau V.11 Durée de vie de certaines polluantes dans l'atmosphère.

V.4.1 Les données de prédiction

Dans cette deuxième application Nous avons précédé la simulation des séries temporelles pollution atmosphérique. La concentration mensuelle d'ozone et de dioxyde carbone pour une durée de 32 ans à partir de 1965 jusqu'a 1996 prise de **Hipel-McLeod Time Series Datasets Collection** [14].

V.4.2 Résultats de simulation

Dans cette deuxième application, 2 prédicteurs neuronaux sont utilisée pour prédire séparément La concentration mensuelle d'ozone et concentration mensuelle de dioxyde carbone. Les données sont divisées dans deux parties, dans lesquelles les premiers 25 ans équivalent d'un nombre de données de 300 sont utilisées pour la phase d'apprentissage, et les derniers 7 ans équivalent d'un nombre de données de 84 sont utilisé pour la phase de test. Le nombre d'itération d'apprentissage utilisée est de 200 à 500 itérations.

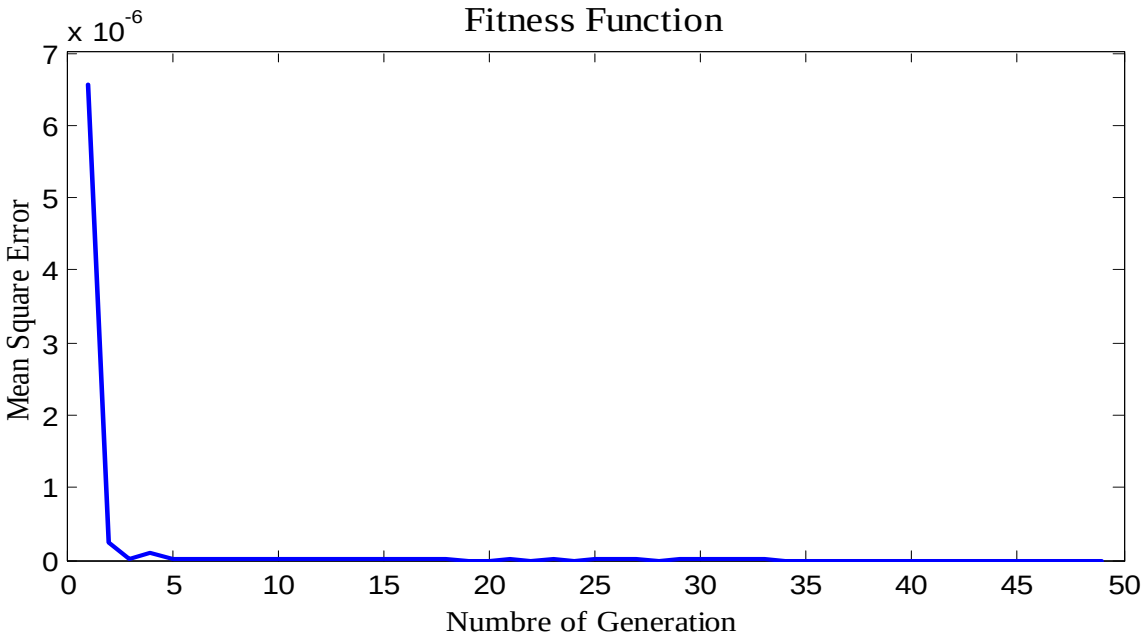


Figure V.28 Evolution de la fonction objectif à travers les générations (concentration d'ozone)

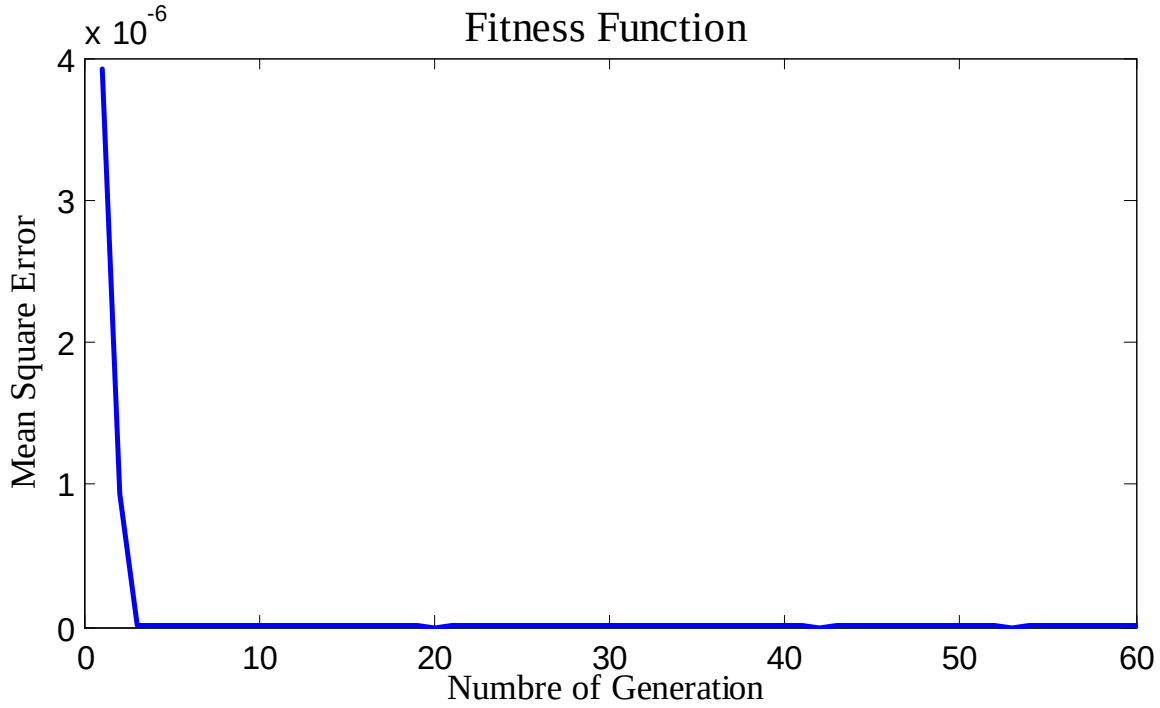


Figure V.29 Evolution de la fonction objectif à travers les générations (concentration de dioxyde carbone)

Les résultats obtenus pendant exécution des AG 's sont portés sur les figures V. 28, 29.
La valeur du critère de performance ***fitness*** est rapidement amélioré au cours des premières générations, et est ramenée de 7.10^{-6} à 3.3099×10^{-17} au bout de 124 itérations pour la prédiction d'ozone et 4.10^{-6} à 4.3098×10^{-17} au bout de 622 itérations pour la prédiction de dioxyde carbone.
Les figures V. 31,36 présentent La concentration d'ozone et de dioxyde carbone mensuelle actuel et prédit générée à la fin de la phase d'apprentissage.

Les figures V. 32, 37 présentent l'erreur de prédiction générée à la fin de phase de d'apprentissage pour les 2 prédicteurs neuronaux. Nous remarquons que l'erreur de prédiction est inférieur à 2.5×10^{-8} pour le prédicteur neuronal d'ozone, et inférieur à 1.5×10^{-8} pour le prédicteur neuronal de dioxyde carbone. On constate que les valeurs de l'erreur de prédiction sont très faibles est ces résultats sont très satisfaisantes.
Les figures V. 33, 38 présentent La concentration d'ozone et de dioxyde carbone mensuelle actuel et prédit de la phase de test.

Les figures V. 34, 39 présentent l'erreur de prédiction générée de la phase de test pour les 2 prédicteurs neuronaux. Nous remarquons que l'erreur de prédiction est inférieur à 1.5×10^{-8} pour le prédicteur neuronal d'ozone, et inférieur à 2×10^{-8} pour le prédicteur neuronal dioxyde carbone.

paramètres de prédicteur neuronal	Concentration d ’ozone
Nombre de couche cachée	1
Coefficient d’apprentissage	0.34112
coefficient d’inertie	0.53289
la présence ou l'absence de biais	Présence (1)
le nombre d’entrée	20
Retard	1
le nombre de neurone dans couche cachée1	8
Forme de fonction d’activation de couche cachées 1	Semi- linéaire 3
Forme de fonction d’activation de couche de sortie	Semi- linéaire 3
int_{out}	[0.16, 0.4559]
int1	[0.1404, 0.771]
Int2	[0.1572, 0.7688]

Tableau V.12 : la structure optimal reçue à la fin d’AG pour l’ozone

Paramètre de prédicteur neuronal	Concentration de dioxyde carbone
Nombre de couche cachée	2
Coefficient d’apprentissage	0.8594
coefficient d’inertie	0.3911
la présence ou l'absence de biais	Présence (1)
le nombre d’entrée	24
Retard	4
le nombre de neurone dans couche cachée1	10
le nombre de neurone dans couche cachée2	1
Forme de fonction d’activation de couche cachées 1	Semi- linéaire 3
Forme de fonction d’activation de couche cachées 2	Semi- linéaire 3
Forme de fonction d’activation de couche de sortie	Semi- linéaire 3
int_{out}	[0.2781, 0.4554]
int1	[0.5467, 0.98]
Int2	[0.4264, 0.82]

Tableau V.13 : la structure optimal reçue à la fin d’AG pour le dioxyde carbone

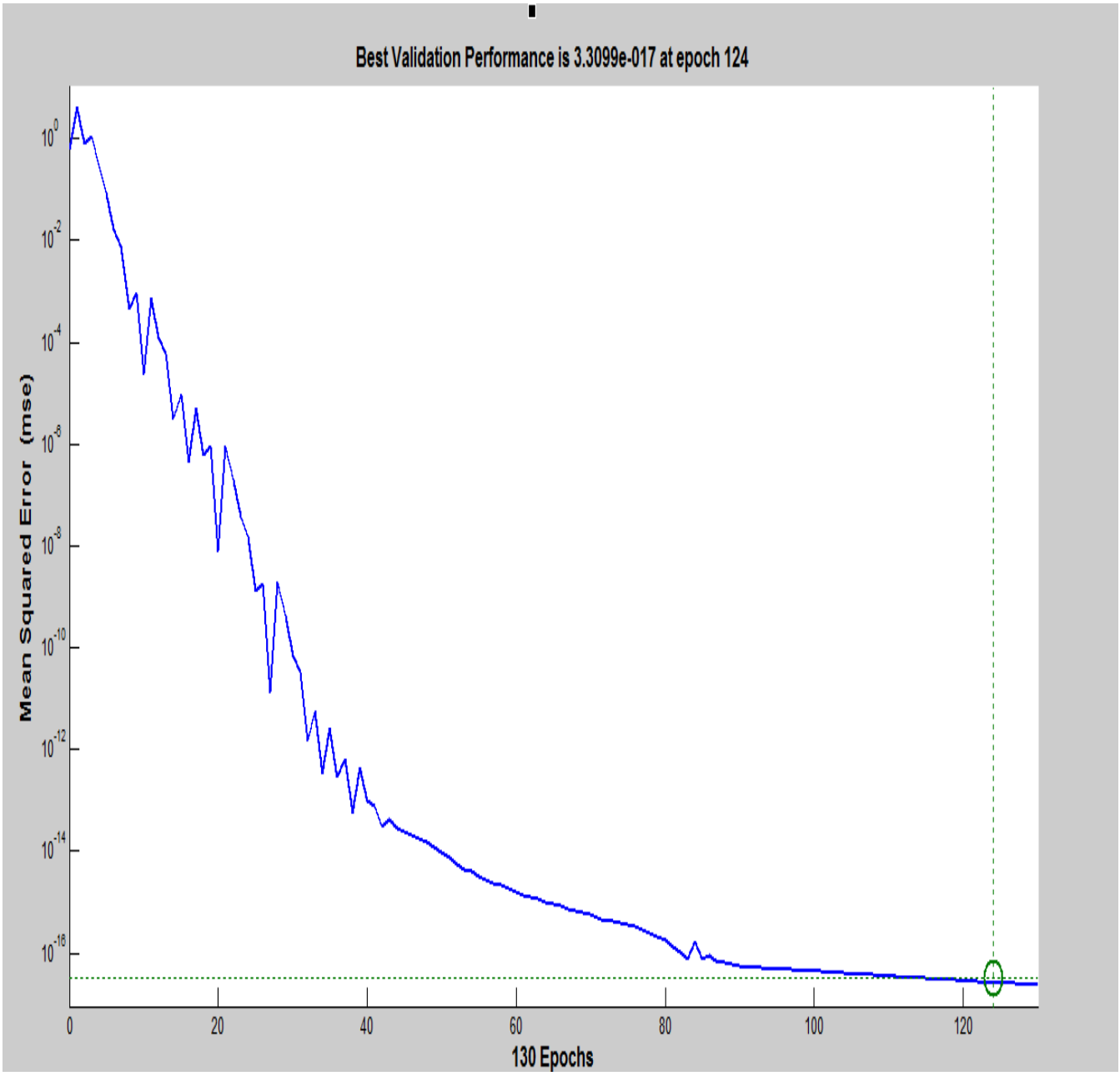


Figure V. 30 Evolution de MSE durant la phase d'apprentissage (ozone)

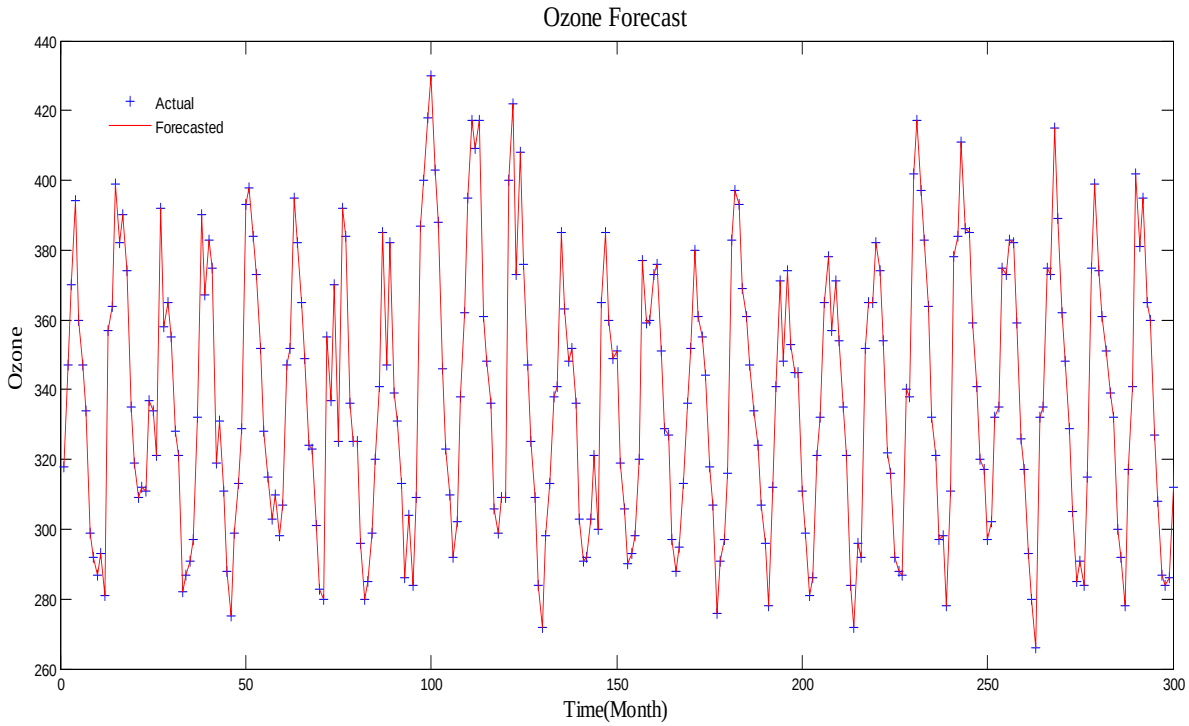


Figure V.31 La concentration d'ozone mensuelle actuel et prédit de la phase d'apprentissage

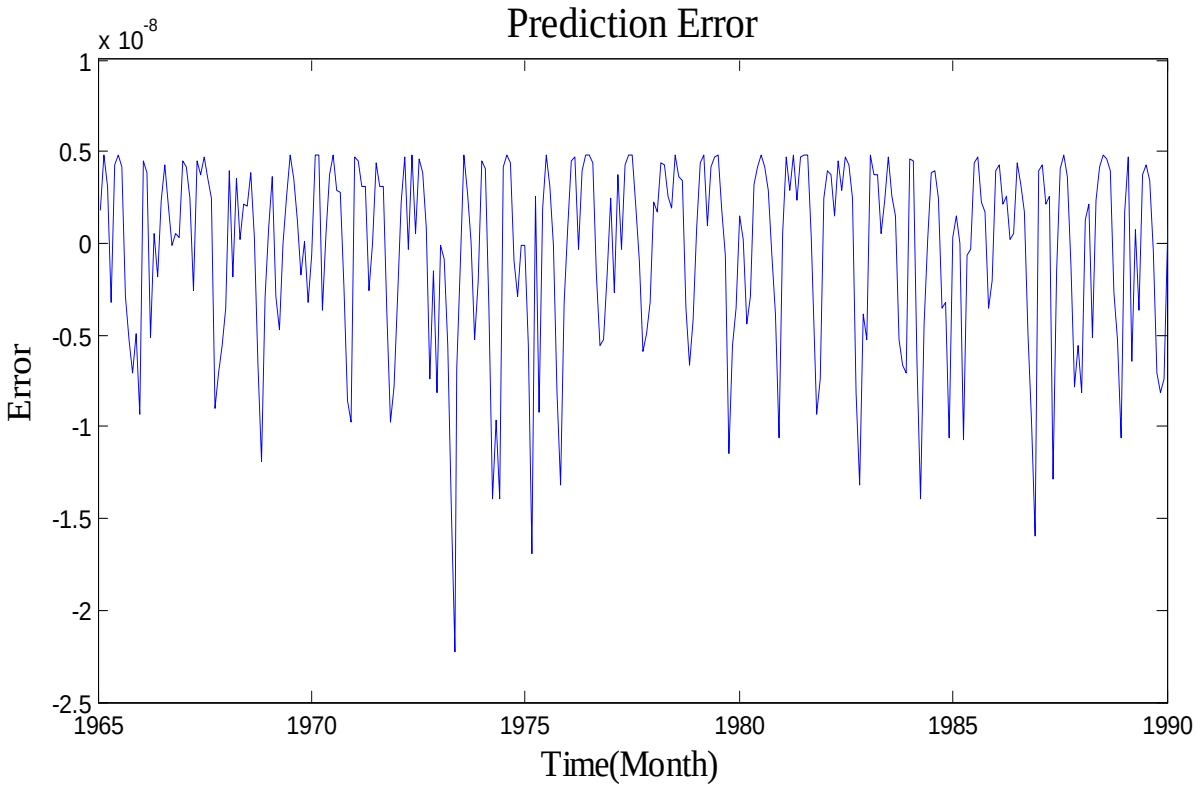


Figure V.32 Erreur de prédiction générée à la fin de phase de d'apprentissage (ozone)

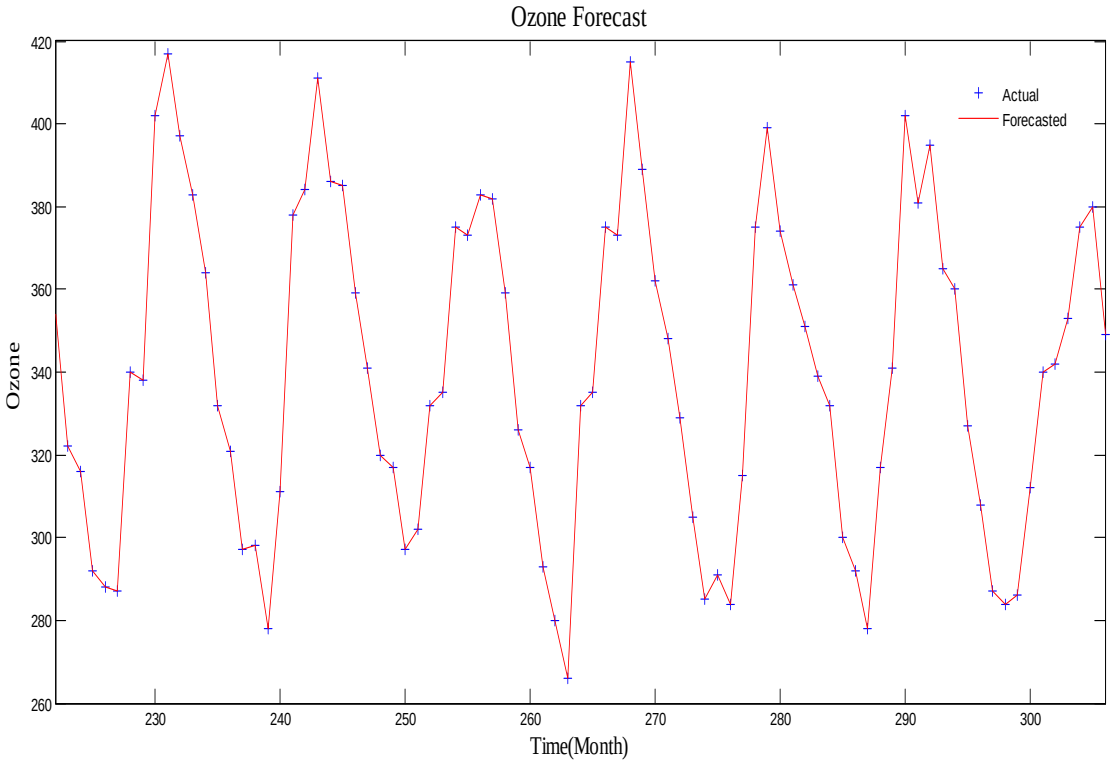


Figure V.33 Concentration d'ozone mensuelle actuel et prédit de la phase de test

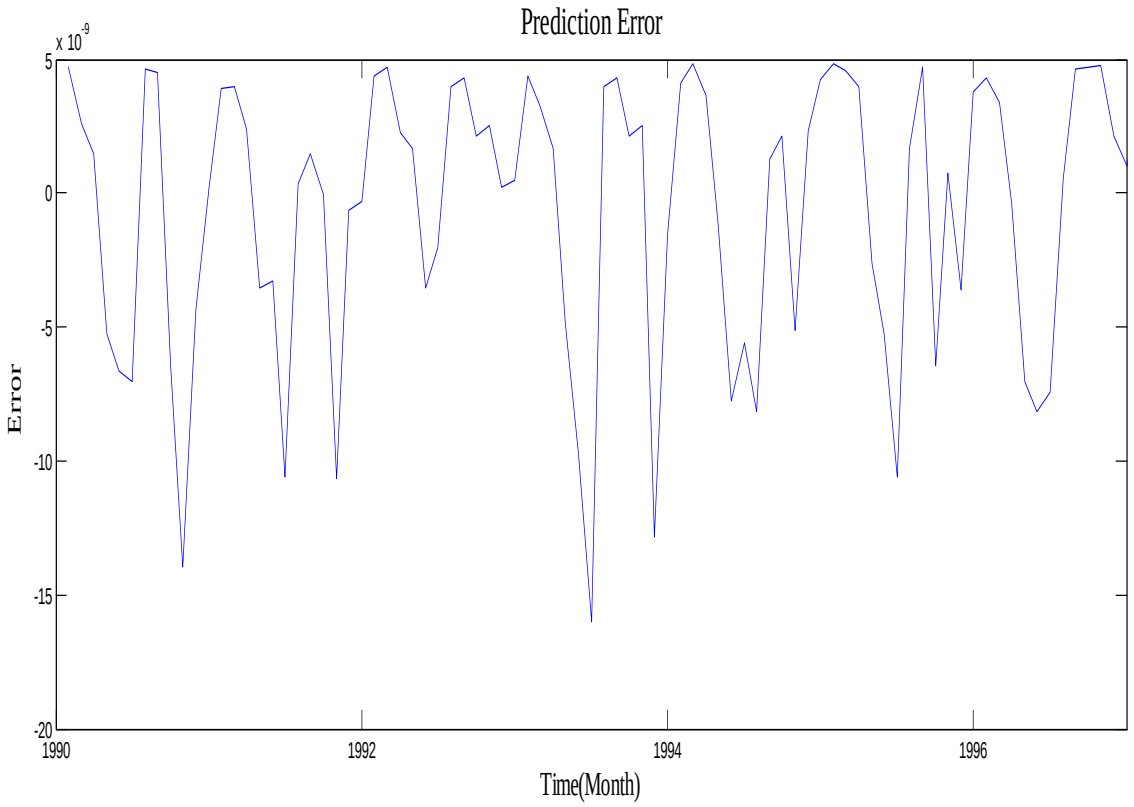


Figure V.34 Erreur de prédiction généré de la phase de test (ozone)

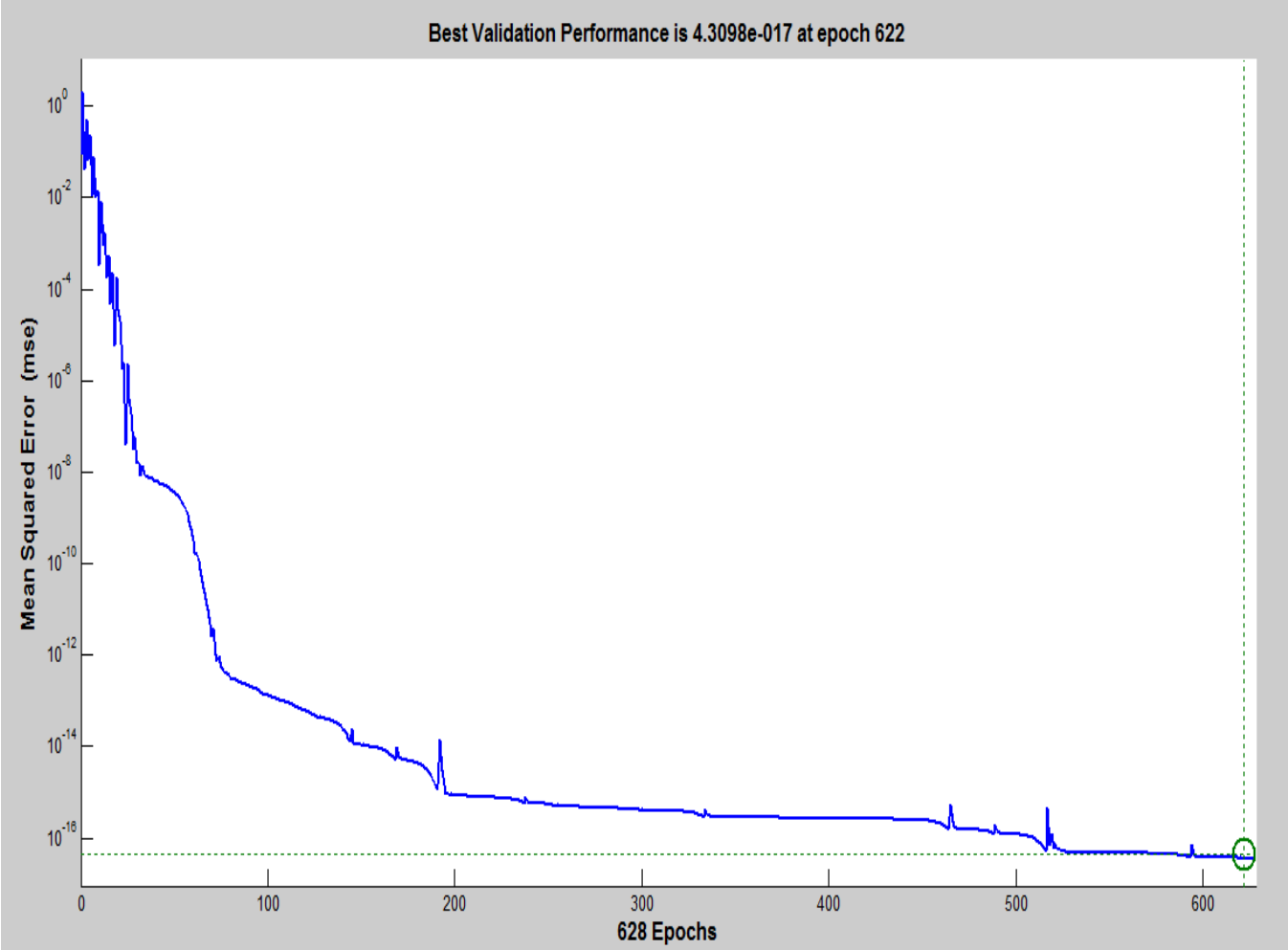


Figure V. 34 Evolution de MSE durant la phase d'apprentissage (dioxyde carbone)

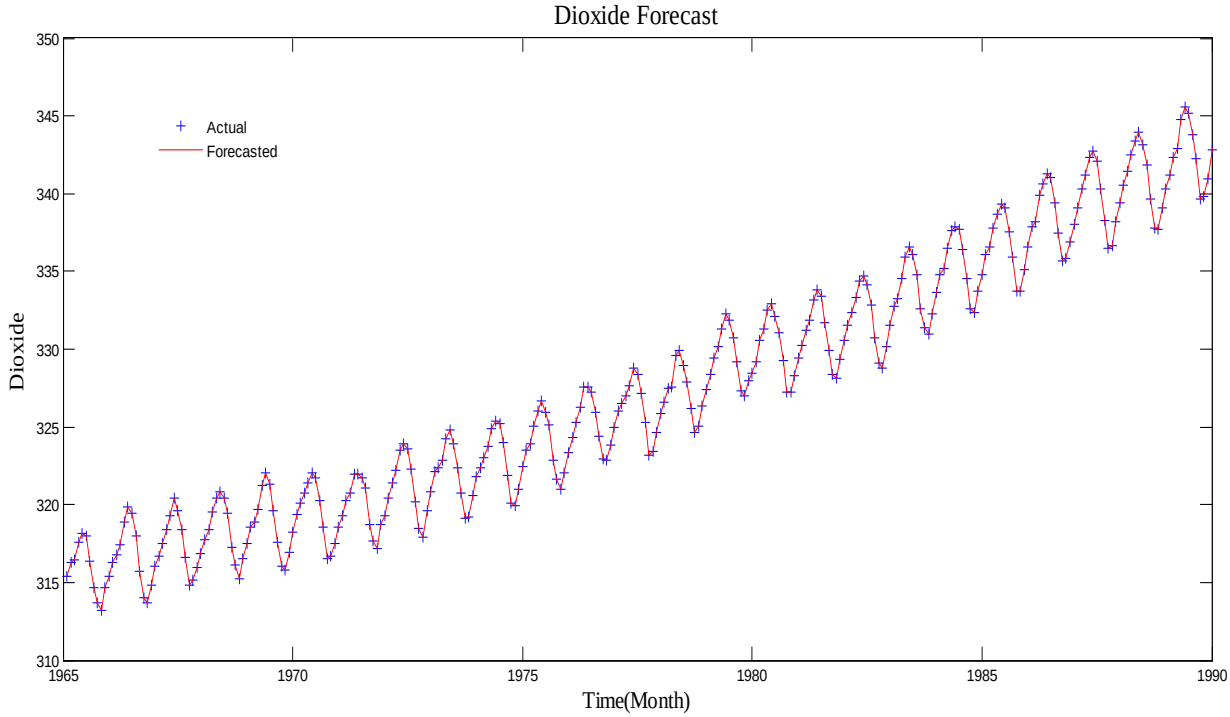


Figure V.36 Concentration de dioxyde carbone mensuelle actuel et prédit de la phase d'apprentissage

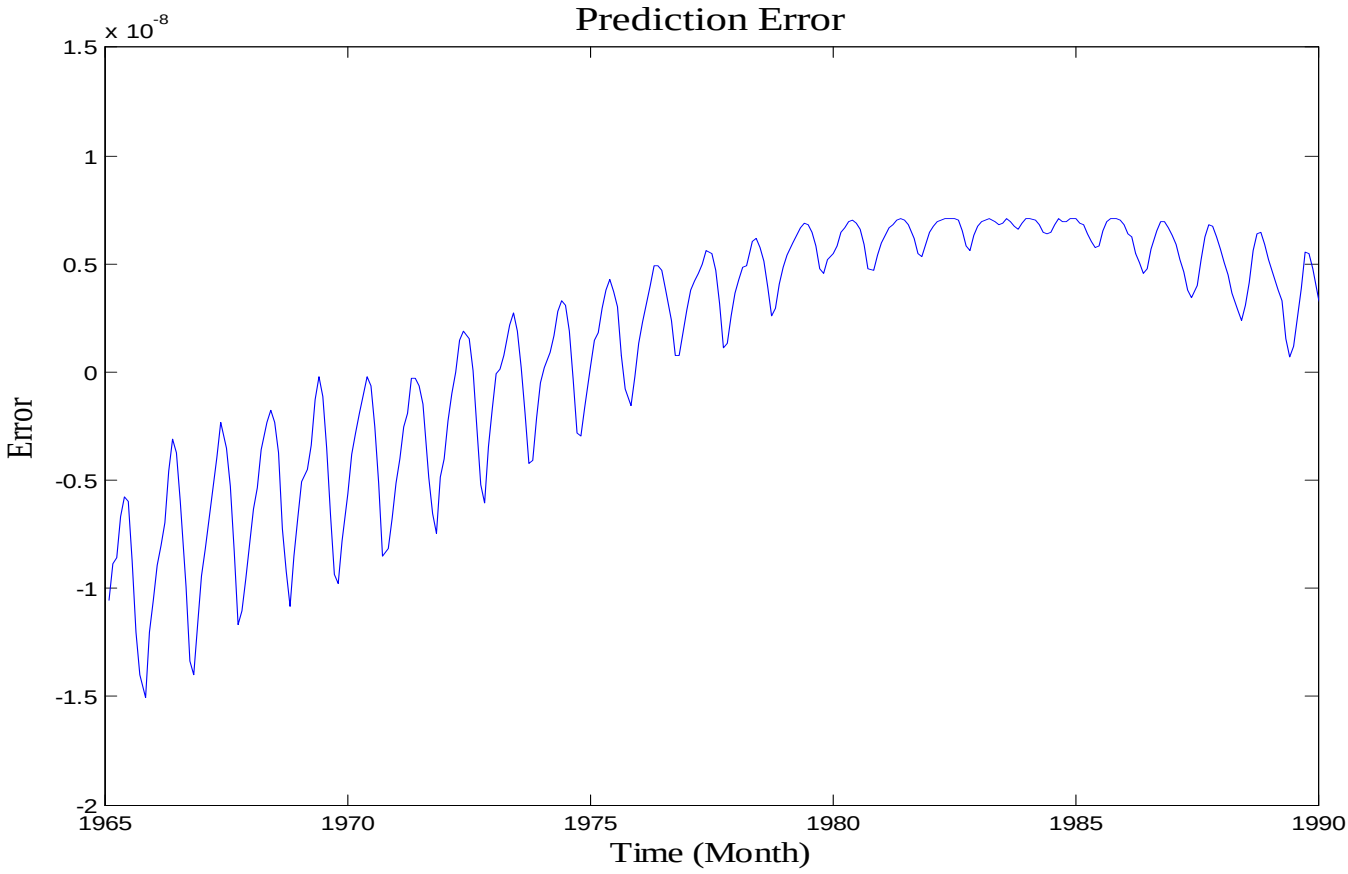


Figure V.37 Erreur de prédiction générée à la fin de phase de d'apprentissage (dioyde carbone)

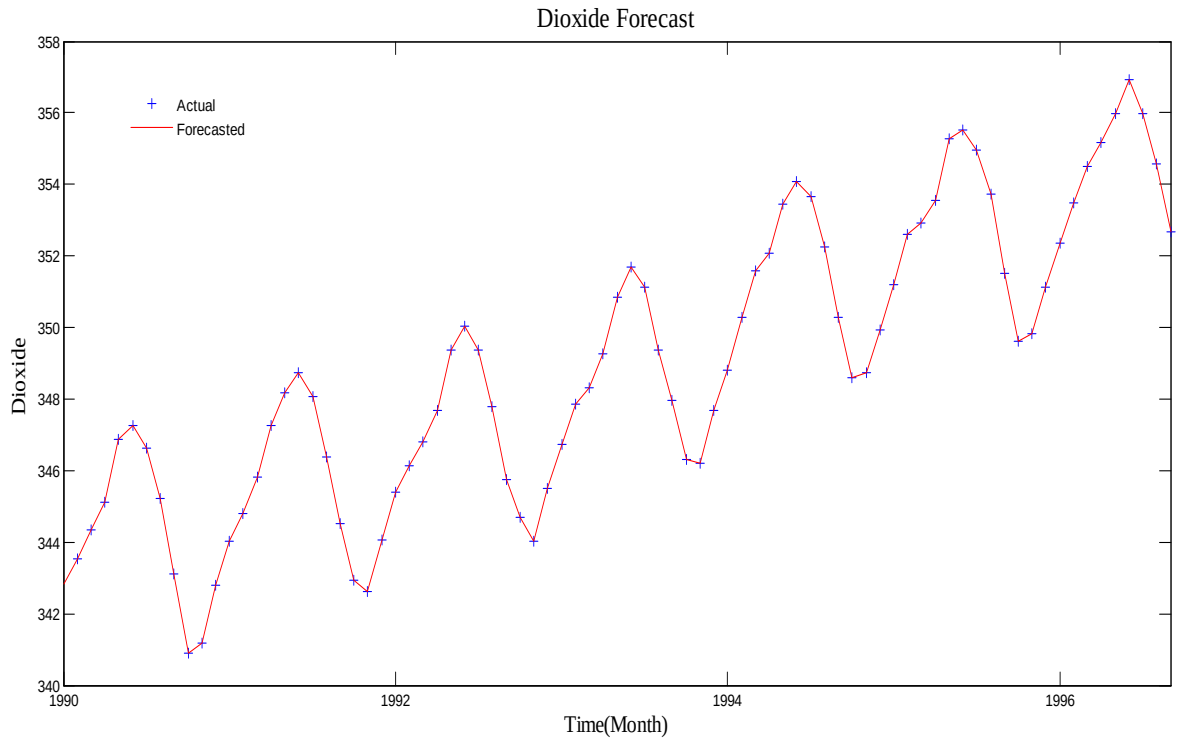


Figure V.38 Concentration de dioxyde carbone mensuelle actuel et prédit de la phase de test

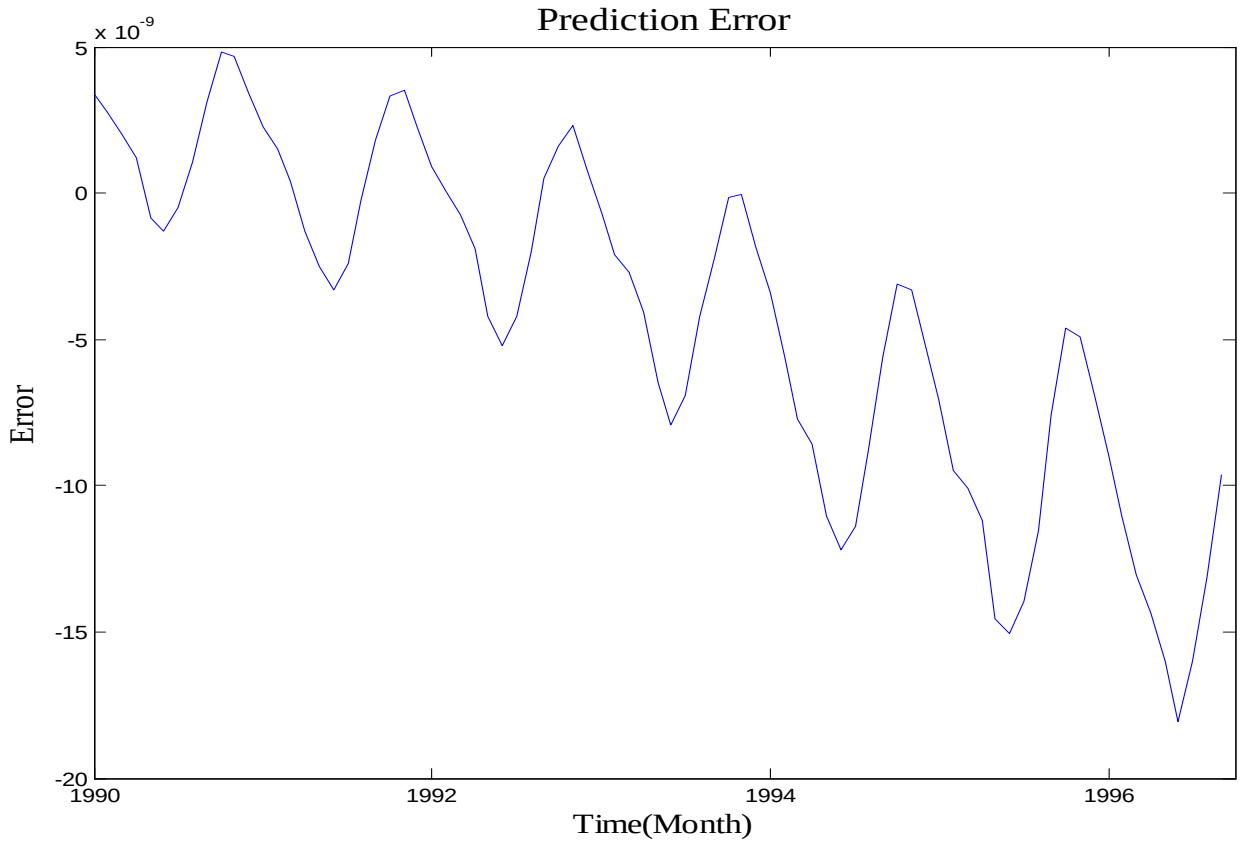


Figure V.39 Erreur de prédiction généré à phase de test (dioxyde carbone)

V.4. Comparaison avec d'autre méthode

Pour comparer les résultats de la méthode proposée appliquer à prédiction de la pollution d'air avec quelques méthodes existantes deux indicateurs de performances sont utilisés : Erreur biais moyenne(MBE), Erreur Absolue moyenne (MAE)

Le tableau V.14 montre la comparaison des résultats entre différentes méthodes de prédiction appliqué aux séries temporelles pollution d'air, les résultats des différentes méthodes sont pris de [41], [42], [43]. On note que les indicateurs de performances (MSE, MBE, MAE) de la méthode proposé est plus faible que les autres méthodes. On peut dire que la méthode proposé produit des indicateurs de performances de plus en plus faibles ainsi qu'on Augmente le nombre de donnée utilisée dans la phase d'apprentissage.
Dans cette deuxième on essai de changer le domaine d'application à l'écologie et aussi changer l'horizon temporelle (mois) pour tester l'efficacité de la méthode.
On constate que les résultats dans cette deuxième application sont moins que celle de la première application cella est due nombre de données utilisées dans la phase d'apprentissage. Le nombre de données utilisées dans cette deuxième application « 300» et celle utilisées dans la première application « 2008 ».

Méthode	MSE	MAE	MBE
Méthode proposé (ozone) (phase d'apprentissage)[51]	3.3099×10^{-17}	$6.566.10^{-8}$	$-3.177.10^{-10}$
Méthode proposé (ozone) (phase de test)[51]	$6.9057.10^{-15}$	$6.816.10^{-9}$	$-4.30.10^{-9}$
Méthode proposé (dioxyde) (phase d'apprentissage)[51]	4.3098×10^{-17}	$2.66.10^{-9}$	$6.59.10^{-10}$
Méthode proposé (dioxyde) (phase de test)[51]	$2.0916.10^{-17}$	$3.56.10^{-9}$	$-3.003.10^{-9}$
Brunelli & Piazza [41]	/	0.014424 /	/
Yu & Wenfang [42] BPNN (phase de test)	5.52	4.2	- 0.01
Yu & Wenfang [42] SVM (phase de test)	18.01	13.38	0.09
Biljana & Mile [43]	/	0.001	/

Tableau V.14 comparaison des résultats

V.5. Application dans le domaine de Météorologique

La prédiction de degrés de température pour un période de temps donnée est très utile dans les régions qui reposent sur l'énergie électrique retenue de l'énergie solaire. On a besoin de cette prédiction pour différents besoins comme planifier et gérer une construction pour prédire la puissance de sortie des transformateurs de l'énergie solaire à l'énergie électrique. La variation de la température peut causer les fluctuations dans la tension et la fréquence. On sait que la température dans la nature est extrêmement incertaine dans le temps et dans l'espace et pour cette raison aucune technologie n'est assez efficace. La température est un des paramètres météorologiques les plus difficiles à prédire. Dans notre cas l'objectif est de mesurer l'efficacité de prédicteur hybride RNA/AG pour prévoir le degré de la température. La prédiction de la température d'un pas de semaine, jour. Nous disposons des données de la station météo de Melbourne en Australie [44].

V.5.1. les données de prédiction

Pour mieux tester la performance de la méthode proposée, nous avons procédé la prédiction de la température journalière et hebdomadaire de Melbourne. Les données de la température utilisée sont les valeurs moyennes journalières pour une période de 10 ans à partir de l'année 1981 jusqu'à l'année 1990 pris de bureau de météorologie de l'Australie à Melbourne. Les valeurs de température moyenne journalière sont utilisées pour l'obtention des valeurs de la température hebdomadaire.
L'objectif de cette partie est la prédiction de la température pour un pas : jour ; semaine.

V.5.2 Résultats de simulation

2 prédicteurs neuronaux sont utilisée pour prédire séparément la température journalière et la température hebdomadaire de Melbourne. Les données sont divisées en deux partie : Les premières 7 ans valeurs de température sont utilisées pour **la phase d'apprentissage**, et les 3 ans qui restent sont utilisées pour la phase de test.

1. La prédiction de la température journalière : un nombre de 2555 valeurs de températures sont utilisées pour la phase d'apprentissage et un nombre de 1095 valeurs de températures sont utilisées pour la phase de test.
2. La prédiction de la température hebdomadaire : un nombre de 400 valeurs de températures sont utilisées pour la phase d'apprentissage et un nombre de 121 valeurs de températures sont utilisées pour la phase de test.

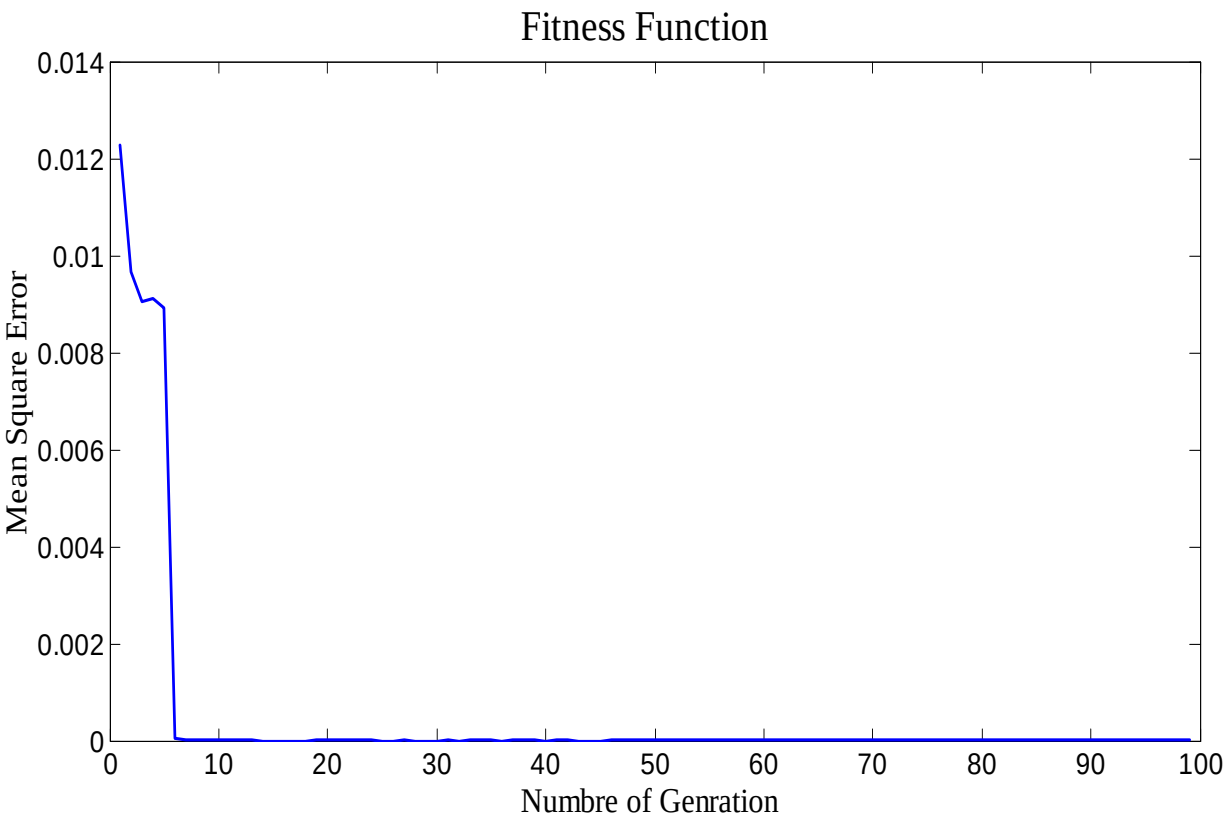


Figure V. 40 Evolution de la fonction objectif à travers les générations (température journalière)

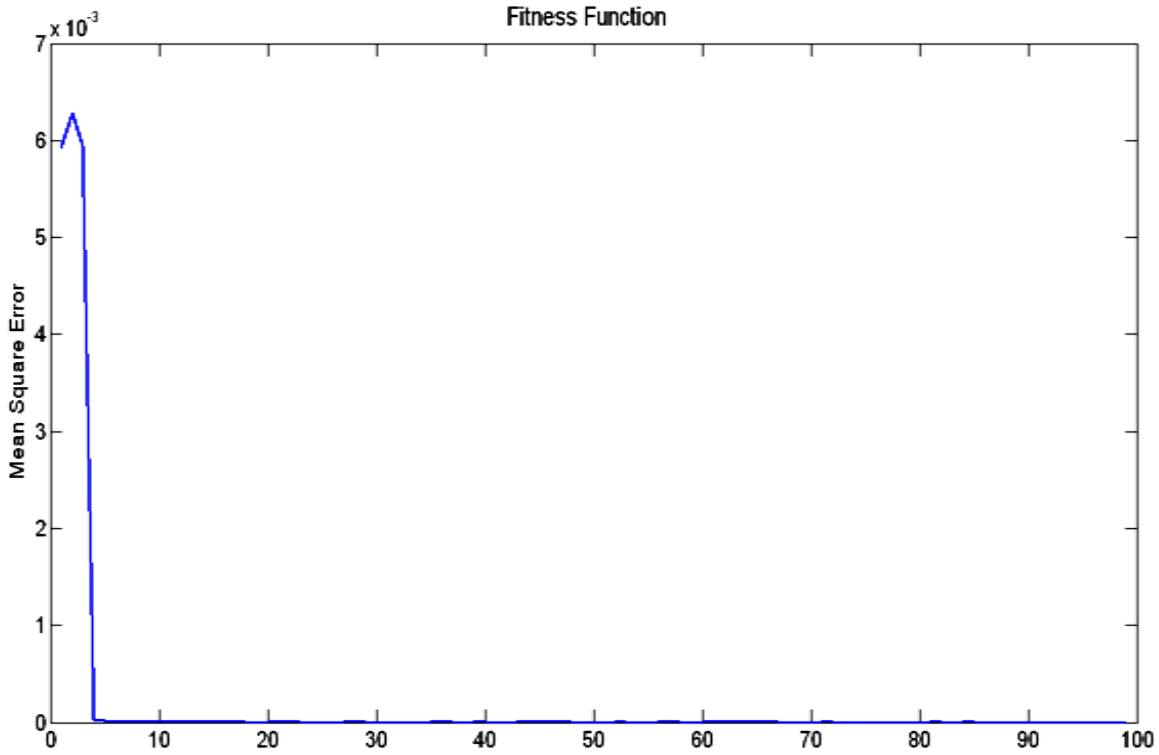


Figure V. 41 Evolution de la fonction objectif à travers les générations (température hebdomadaire)

Les résultats obtenus pendant exécution des AG 's sont portés sur les figures V. 40, 41 *Evolution de la fonction objective à travers les générations*

La valeur du critère de performance ***fitness*** est rapidement améliorée au cours des premières générations pour la prédiction de température journalière et hebdomadaire.

On remarque que l'AG à éprouvée ca efficacité dès les premières générations du processus d'optimisation, la fonction objective ***fitness*** pour la prédiction de température journalière est ramené à 1.6267×10^{-31} lors qu'il était initialement (première génération) de 0.0012, et pour la prédiction de température hebdomadaire est ramené à 2.9678×10^{-23} lors qu'il était initialement (première génération) de 0.007

Les figures V.43, 48 présentent les sorties des deux prédicteurs générées à la fin de la phase d'apprentissage ainsi que les sorties actuelles

Les figures V. 44, 49 présentent les erreurs de prédiction générées à la fin de la phase d'apprentissage.

Les figures V.45, 50 présentent les sorties des deux prédicteurs générées de la phase de test ainsi que les sorties actuelles

Les figures V. 46, 51 présentent les erreurs de prédiction générées de la phase de test.

Paramètre de prédicteur neuronal	Température journalière
Nombre de couche cachée	2
Coefficient d'apprentissage	0.8052
coefficient d'inertie	0.6031
la présence ou l'absence de biais	Absence (0)
le nombre d'entrée	5
Retard	8
le nombre de neurone dans couche cachée1	11
le nombre de neurone dans couche cachée2	11
Forme de fonction d'activation de couche cachées 1	Semi- linéaire 3
Forme de fonction d'activation de couche cachées 2	Semi- linéaire 3
Forme de fonction d'activation de couche de sortie	Semi- linéaire 3
int_{out}	[0.153 0.337]
int1	[0.151 0.291]
Int2	[0.0319, 0.0986]

Tableau V.15 : la structure optimal reçue à la fin d'AG pour Température journalière

Paramètre de prédicteur neuronal	Température hebdomadaire
Nombre de couche cachée	2
Coefficient d'apprentissage	0.5789
coefficient d'inertie	0.1281
la présence ou l'absence de biais	Présence (1)
le nombre d'entrée	3
Retard	1
le nombre de neurone dans couche cachée1	25
le nombre de neurone dans couche cachée2	25
Forme de fonction d'activation de couche cachées 1	Semi- linéaire 3
Forme de fonction d'activation de couche cachées 2	Semi- linéaire 3
Forme de fonction d'activation de couche de sortie	Semi- linéaire 3
int_{out}	[0.003, 0.409]
int1	[0.0504, 0.516]
Int2	[0.7093 0.9699]

Tableau V.16 : la structure optimal reçue à la fin d'AG pour Température hebdomadaire

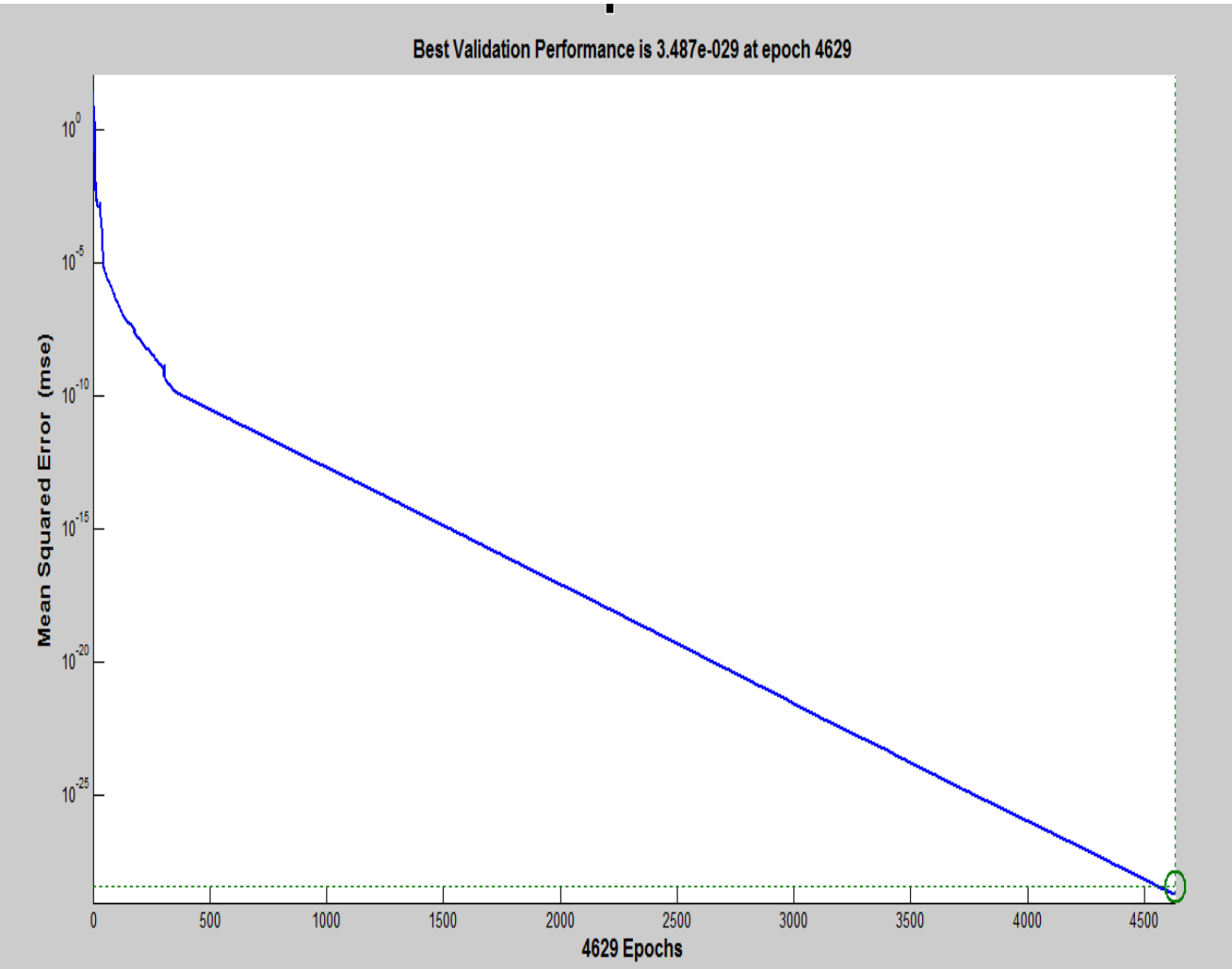


Figure V. 42 Evolution de MSE durant la phase d'apprentissage (Température journalière)

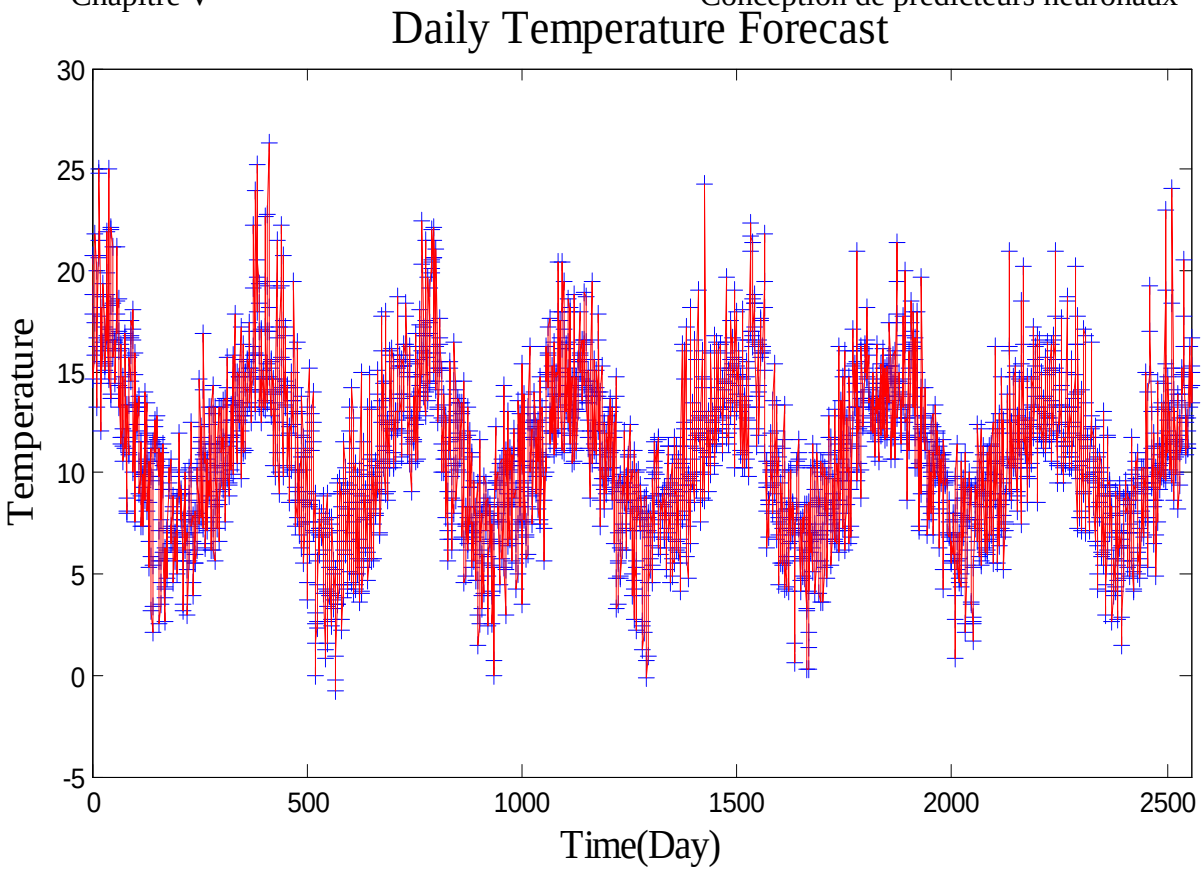


Figure V.43 La température journalière actuel et prédit généré à la fin de phase d'apprentissage

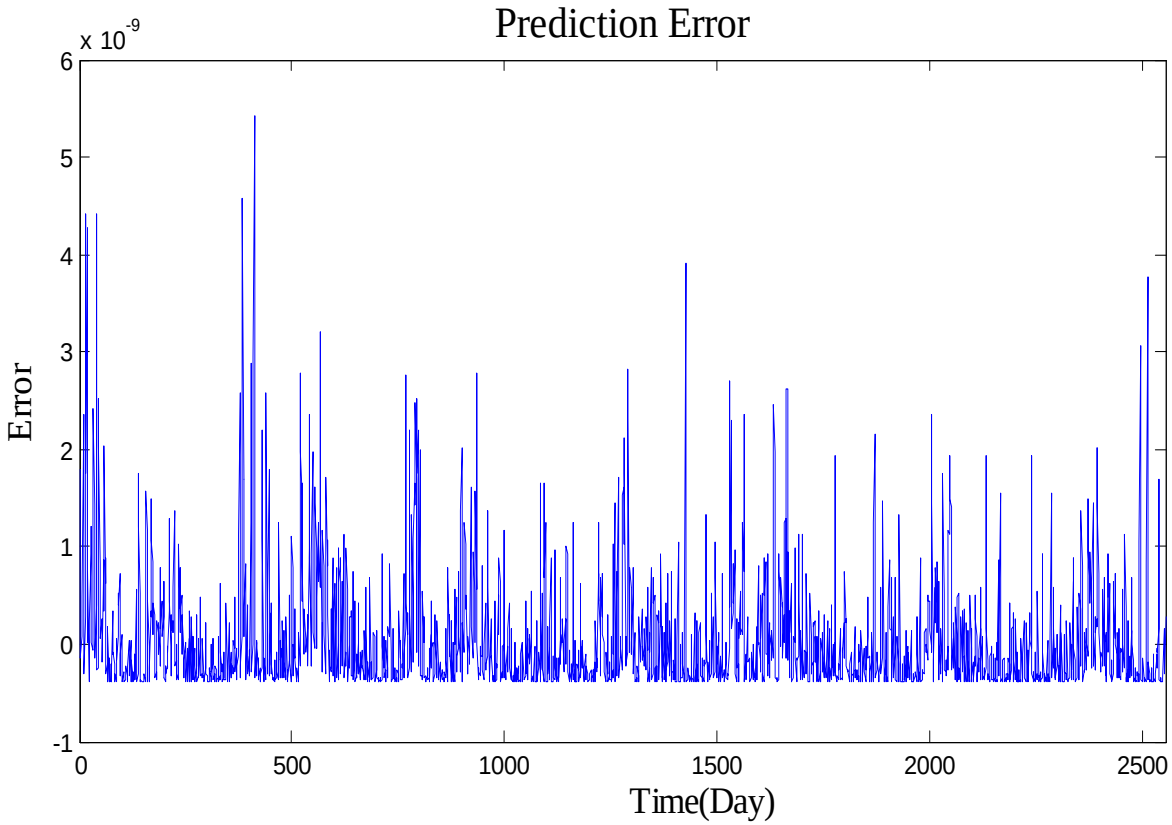


Figure V.44 Erreur de prédiction générée à la fin de phase de d'apprentissage (température journalière)

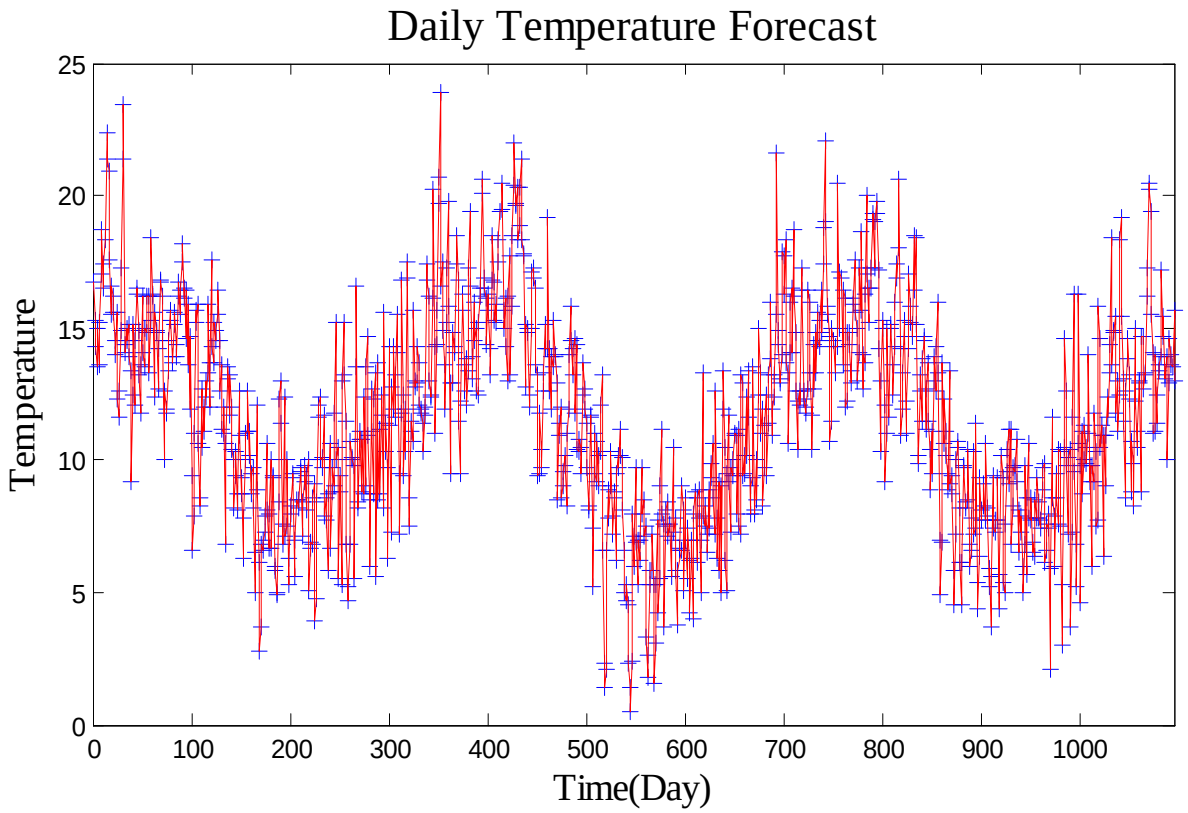


Figure V.45 La température journalière actuel et prédit généré de la phase de test

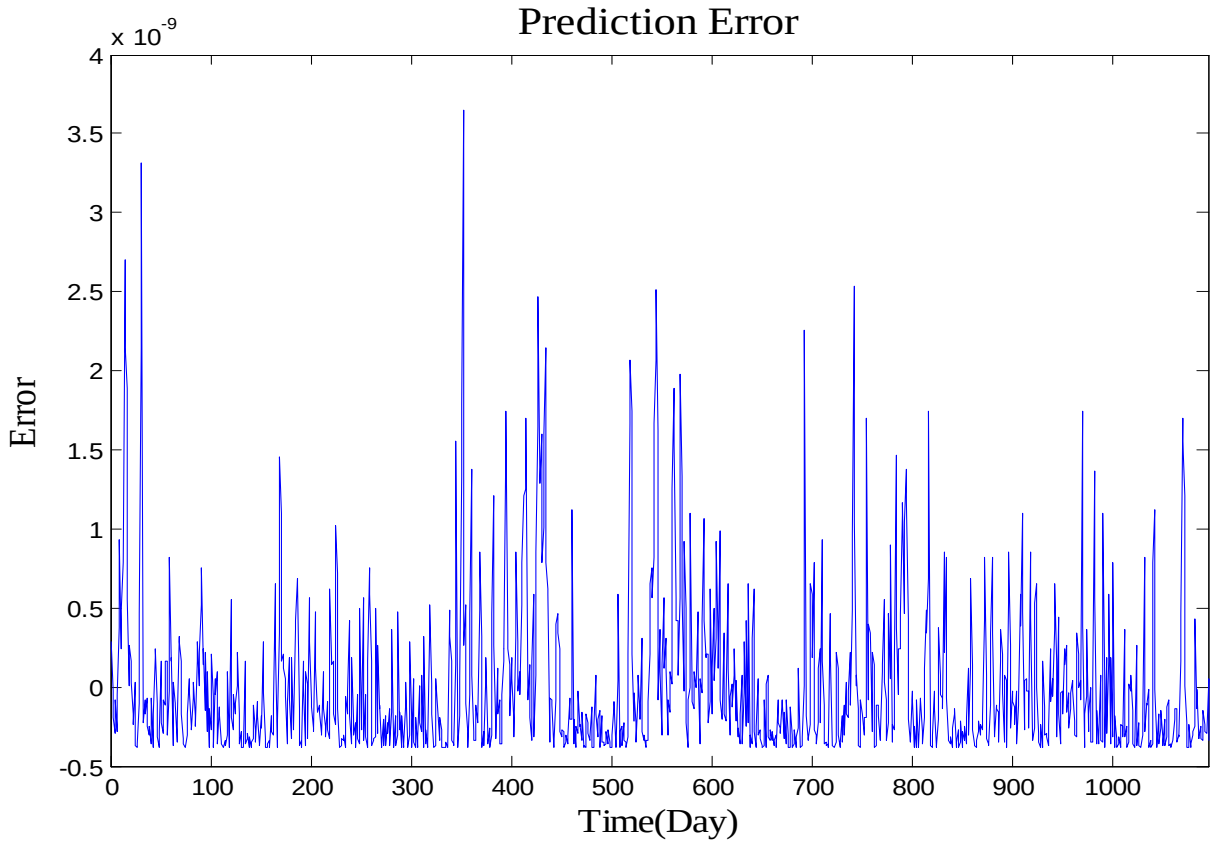


Figure V.46 Erreur de prédiction générée de la phase de test (température journalière)

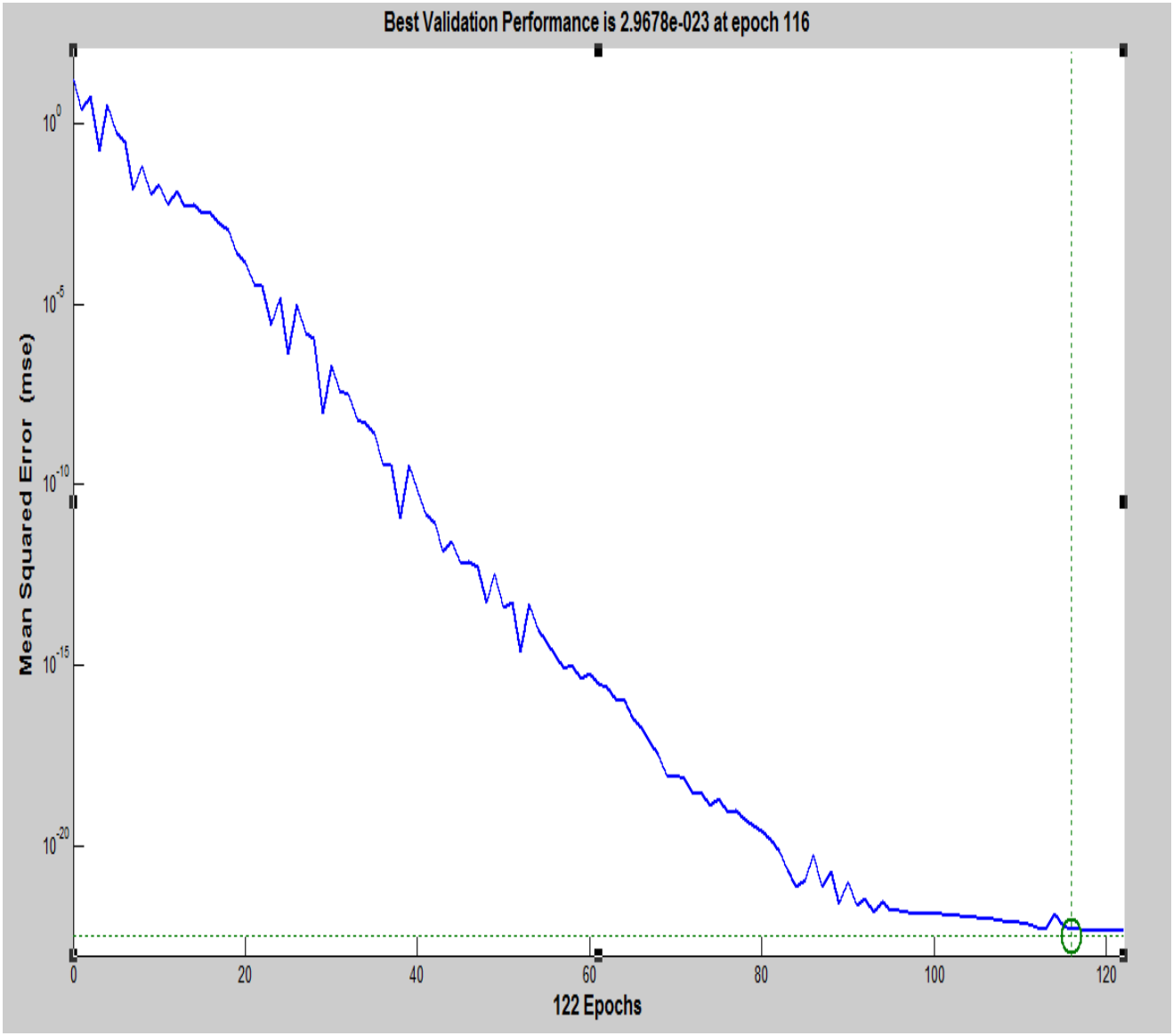


Figure V. 47 Evolution de MSE durant la phase d'apprentissage (Température hebdomadaire)

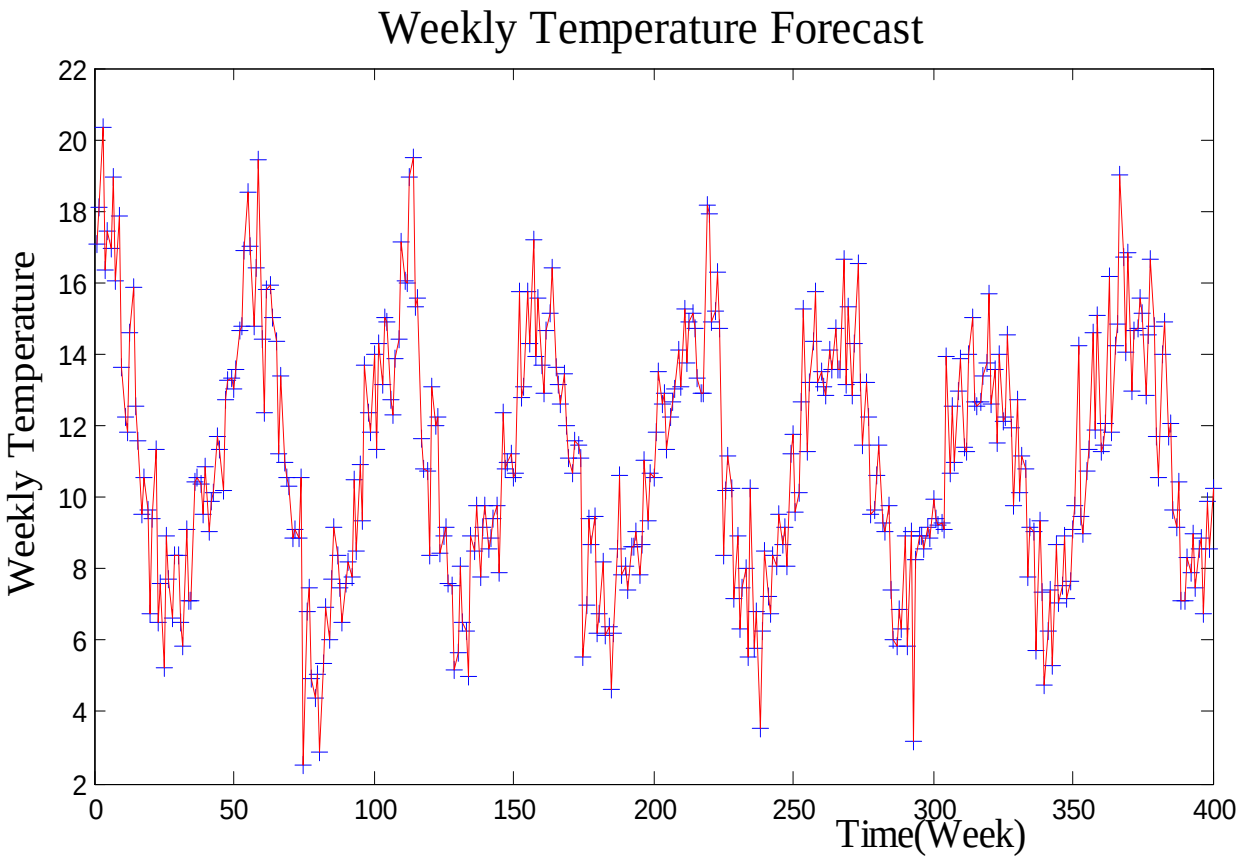


Figure V.48 La température hebdomadaire actuel et prédit généré à la fin de phase d'apprentissage

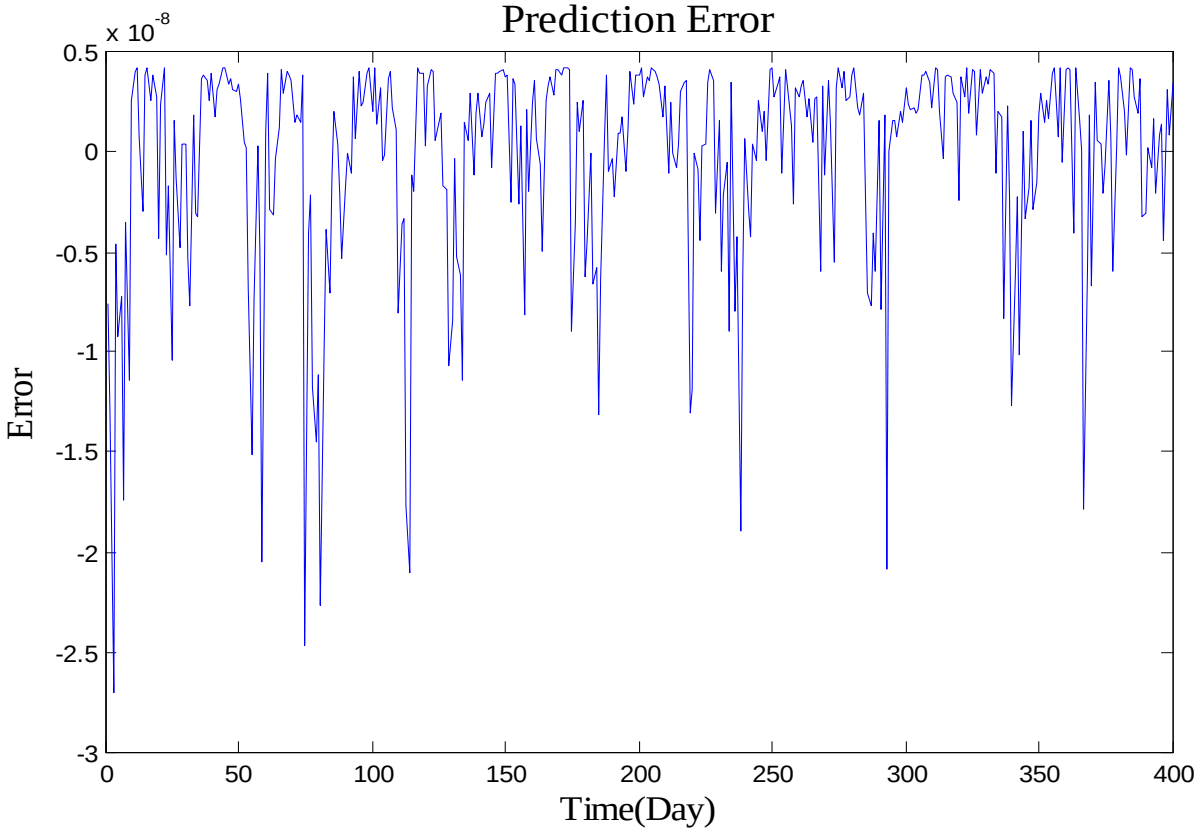


Figure V.49 Erreur de prédiction générée à la fin de phase de d'apprentissage (température hebdomadaire)

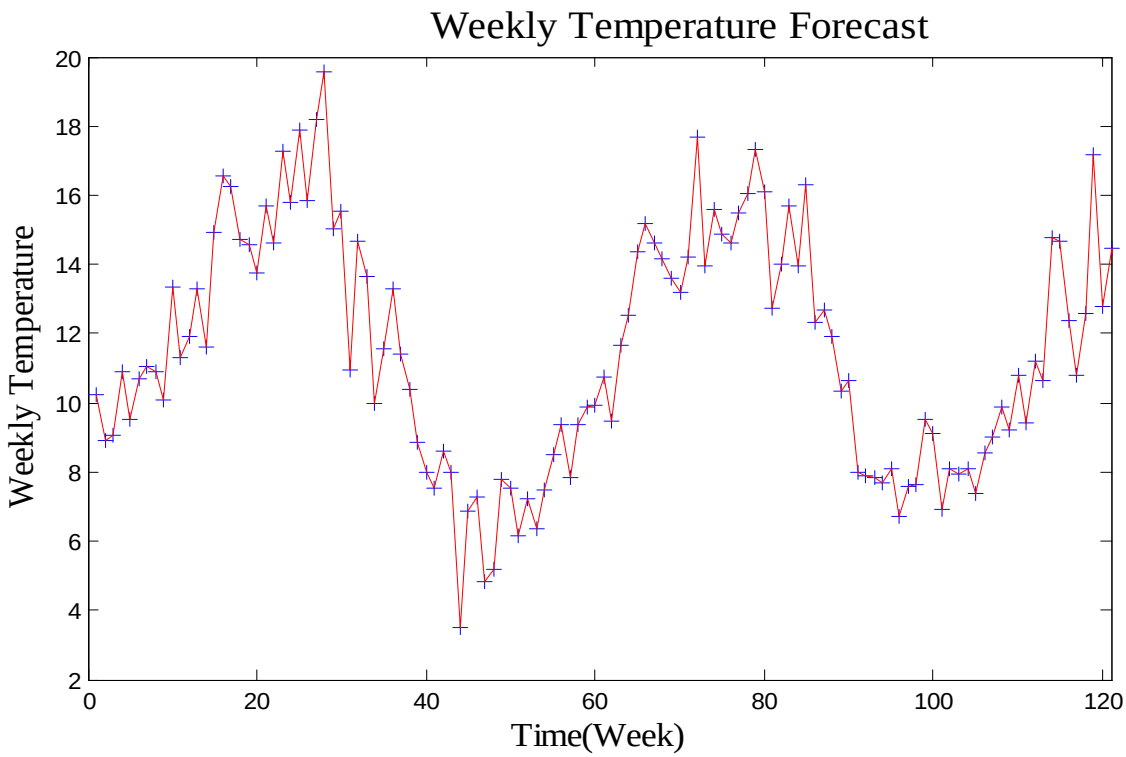


Figure V.50 La température hebdomadaire actuel et prédit généré de la phase de test

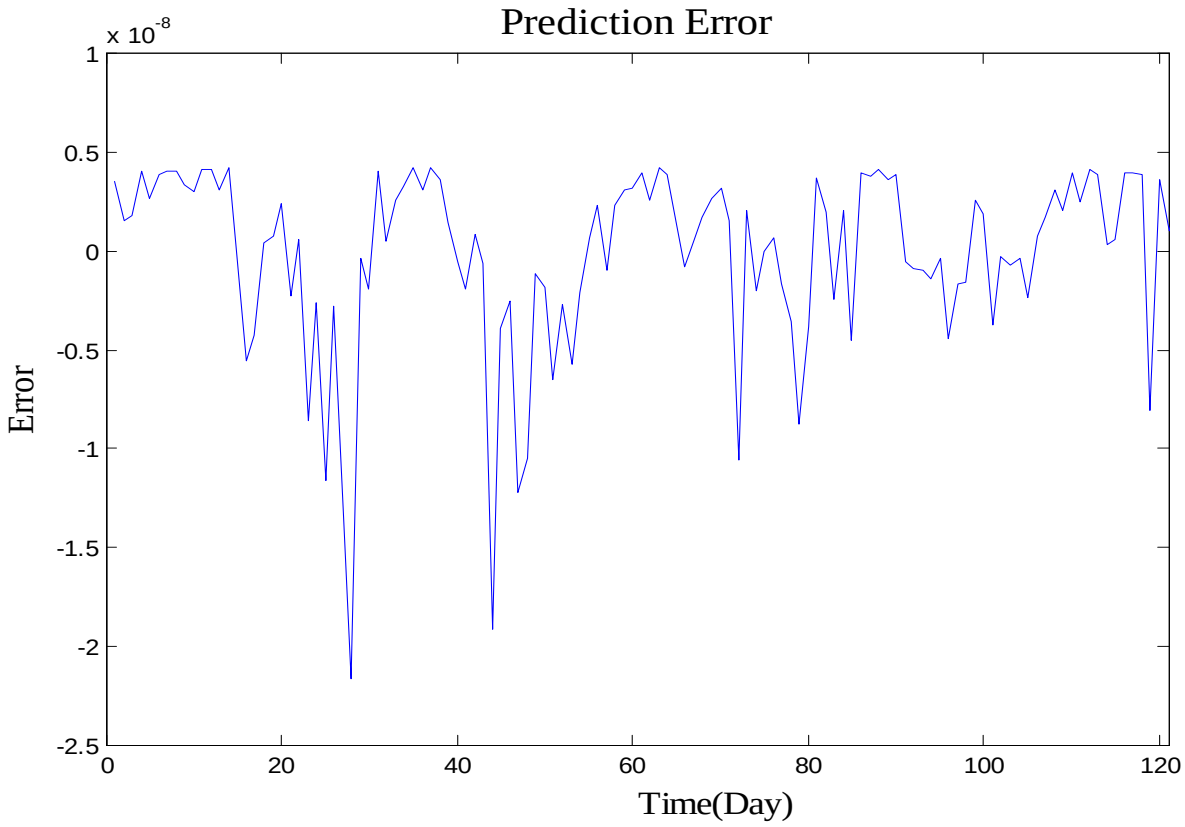


Figure V.51 Erreur de prédiction générée de la phase de test (température hebdomadaire)

V.5.4. Comparaison avec d'autres méthodes

Nous présentons dans cette partie une analyse comparative des résultats de la prédiction de température de prédicteur hybride RNA/AG proposé avec d'autres résultats existants en terme 3 indicateurs de performances : Erreur Absolue moyenne (MAE), Le taux d'erreur de prédiction moyenne (AFER), erreur quadratique moyenne (MSE).

Le tableau 7 montre la comparaison entre différentes résultats des méthodes de utilisées pour la prédiction de température, les résultats des différentes méthodes sont pris de [46 , 47 , 48 , 50], on note que les indicateurs de performances(MSE, MAE, MSE) de la méthode proposé sont plus faible que les autres méthodes.

On peut dire que la méthode proposé produit des indicateurs de performances de plus en plus faibles ainsi qu'on

1. Augmente le nombre de donnée utilisée dans la phase d'apprentissage.
2. Le bons choix de la population initiale.

Les résultats de la méthode proposé appliquer à la prédiction de la température journaleire sont mieux que celle de la température hebdomadaire.

Méthode	MSE	MAE	AFER
méthode propose: température Journaleire (phase d'apprentissage)[45]	$3.6395.10^{-17}$%	$3.923.10^{-10}$	$3.324.10^{-11}$%
methode propose: temperature Journaleire (phase de test)[45]	$2.6445.10^{-17}$%	$3.535.10^{-10}$	$2.644.10^{-11}$%
methode propose: temperature hebdomadaire (phase d'apprentissage)[45]	$2.8046.10^{-15}$%	$3.774.10^{-9}$	$1.584.10^{-9}$%
method e propose: temperature hebdomadaire (phase)[45]	$2.189.10^{-15}$%	$3.332.10^{-9}$	$4.6480.10^{-9}$%
Lee & Wang [46]	/	/	0.64%
Radhika & Shashi (SVM) [47]	7.15%	/	/
Smith &McClendon [48]	/	/	0.09%
Hayati & Mohebi [50]	/	1.6	/

Tableau V.17 Comparaison des résultats

V.6. Conclusion

Ce chapitre a montré l'application de prédicteur hybride RNA/AG dans trois domaines : économie, écologie, météorologie.

Nous avons donc présenté dans un premier temps la méthode d'optimisation d'un prédicteur neuronal basée sur un algorithme génétique réel. Cette méthode consiste en l'optimisation simultanée de la topologie de réseaux de neurones (le nombre de couches, les nombres de neurones par couche, la forme de la fonction d'activation utilisée par couches, la présence ou l'absence de biais, le nombre d'entrée de prédicteur, le retards), les paramètres de contrôle de réseaux de neurones (paramètre d'apprentissage, paramètre d'inertie), et les intervalles initiaux des poids synaptiques des couches cachées et de couche de sortie. Nous avons d'abord proposé une méthode de représentation de l'ensemble des paramètres à optimiser du prédicteur neuronal et l'espace de recherche de chacun des paramètres. Cette représentation fait intervenir un codage en nombre réels. Ensuite nous avons décrit l'étape d'initialisation des chromosomes en spécifiant les sous ensembles de chaque gène dans le chromosome.
Enfin pour tester l'efficacité de la méthode, des simulations dans le domaine d'économie ont été considéré. Les résultats obtenus montrent l'importance et l'intérêt des AG dans l'amélioration des performances du prédicteur neuronal. Les résultats obtenus sont très satisfaisants dans la mesure des indicateurs de performances. Néanmoins, l'inconvénient de la méthode réside au niveau de choix de la population initiale qui influe sur la qualité d'optimisation et aussi sur la vitesse de convergence des AG's. La deuxième simulation effectue dans le domaine d'écologie. Les résultats de simulations ont confirmé l'efficacité de la méthode proposé malgré la réduction de nombre de données utilisées dans la phase d'apprentissage, les indicateurs de performances sont plus élevée que celles de la première simulation mais toujours satisfaisante. La troisième et la dernière simulation concerne le domaine météorologie. Là aussi, les résultats de simulations ont confirmé l'intérêt de la méthode: l'erreur quadratique moyenne est considérablement réduite et les résultats de simulations obtenues sont toujours satisfaisants.

VI .Conclusion et perspectives

VI. 1. Conclusion

Nous sommes intéressés dans cette thèse, au problème de la conception des prédicteurs neuronaux par les algorithmes génétiques réels. La difficulté de l'obtention de la structure de prédicteur neuronal (le nombre de couches, les nombres de neurones par couche, la forme de la fonction d'activation utilisée par couches, la présence ou l'absence de biais, le nombre d'entrée de prédicteur, le retards), les paramètres de contrôle des réseaux de neurones (paramètre d'apprentissage, paramètre d'inertie), et les intervalles initiaux des poids synaptiques des couches cachées et de couche de sortie est en effet très pénalisante lors de l'utilisation des réseaux de neurones. Le choix de ces paramètres est obligatoirement en fonction des connaissances sur le comportement de la série temporelles et les capacités de l'utilisateur. Ainsi, les performances de prédiction sont en fonction de ces paramètres, ce qui rend les résultats de prédiction très loin de l'optimum. Le but de ce travail est de concevoir un prédicteur neuronal optimal qui peut prédire la future valeur d'une série temporelle avec des indicateurs de performances très faible.

Cette thèse s'inscrit dans la continuité des travaux existants visant à simplifier l'étape de conception en proposant une méthode d'hybridation des techniques du soft computing. Ainsi, nous avons présenté dans un premier temps la méthode de conception du prédicteur neuronal par un algorithme génétique à codage en nombre réel qui optimise simultanément:

- la structure de RNA
- Les paramètres de contrôles de RNA
- Les intervalles initiaux des poids

Nous avons décrit le principe de codage en nombre réel des paramètres du prédicteur neuronal puis nous avons présenté les opérateurs génétiques permettant d'exploiter le codage en nombre réel. La méthode de sélection est celle du 'remainder selection'. Pour ne pas rester dans un cadre descriptif, nous avons présenté la simulation de cette méthode dans 3 domaines différents: économie, écologie, météorologie.
Bien que les résultats obtenus soient satisfaisants quelque soit la nature de série temporelle non stationnaire (dioxyde le carbone), saisonnière (température) en terme d'erreur prédiction, la méthode comme la plupart des méthodes proposées dans la littérature, présente un inconvénient majeur c'est l'influence de nombre de données utilisées dans la phase d'apprentissage sur la vitesse d'optimisation et aussi sur la qualité de prédiction (erreur de prédiction). L'augmentation de nombre des données utilisées dans la phase d'apprentissage améliore la qualité de prédiction et diminue la vitesse de prédiction inversement la diminution de nombre des données utilisées dans la phase d'apprentissage humilié la qualité de prédiction et augmente la vitesse de prédiction. Pour remédier à cet inconvénient, nous avons introduit des sous ensemble $S_i, \mathrm{i} = \{1,.....,7\}$ dont Chaque allèle dans le chromosome est défini. Ces sous ensembles permet de créer des chromosomes performants et donc de débuter le processus d'optimisation avec une bonne population initiale ce qui permet d'avoir un compromis entre la vitesse d'optimisation et la qualité d'optimisation (erreur de prédiction). Les performances de prédicteur hybride RNA/AG ont été vérifiées à travers les différentes séries temporelles utilisées (les taux de change USD /CAD, USD /GBP, USD /EURO, USD /JPY, concentration

d'ozone, concentration de dioxyde carbone, température journalière et hebdomadaire de Melbourne. La comparaison de la méthode proposée avec d'autres méthodes reportés dans la littérature a permis de montrer sa efficacité en terme d'indicateurs de performances.
En conclusion, nous pouvons dire que les algorithmes génétiques représentent un outil très puissant pour la conception des prédicteurs neuronaux.

V.2 Perspectives

Les travaux décrits dans cette thèse peuvent se poursuivre sur plusieurs voies de recherche :

- Tout d'abord, il serait intéressant dans le domaine d'économie d'ajouter d'autre entrées qui influence sur les valeurs de taux de change comme des prix du pétrole, les valeurs d'importation et d'exportation et les taux de croissance de PIB.
- l'amélioration de la performance du prédicteur neuronal par l 'hybridation avec d'autre techniques de soft computing en particulier le système d'inférence flou.
- En appliquant le modèle proposé à d'autres domaines comme l'énergie renouvelable (vitesse de vent, le rayonnement solaire).

Bibliographie

[1] A. Fiordaliso, *Systèmes Flous et Prévision de Séries Temporelles*, Paris: Hermès Science Publication, 1999.
[2] http://robjhyndman.com/TSDL/
[3] BOX G.E.P., JENKINS G.M., *Time Series Analysis, Forecasting and Control*, Holden Day, San Francisco, 1976.
[4] Guy Melard. Méthodes de prévision à court terme. Editions de l'Université de Bruxelles, Avenue Paul Héger, 26 - 1050 Bruxelles(Belgique), 1990.
[5] Holger Kantz and Thomas Schreiber. Nonlinear time series analysis. Cambridge University Press, New York, NY, USA, 2004.
[6] A. S. Weigend and N. A. Gershenfeld, editors. Time series prediction : Forecasting the future and understanding the past. Addison Wesley, 1993.
[7] A. C. Harvey, *Forecasting: Structural Time Series Models and the Kalman Filter*, Cambridge University Press, 1989.

[8] Steven C. Wheelwright Spyros Makridakis and Rob J. Hyndman. Forecasting :Methods and Applications. John Wiley & Sons, 1998.
[9] M. B. Priesley, ‘‘State-Dependent Models: A General Approach to Non-Linear Time Series Analysis’’, *Journal of Times Series Analysis*, Vol. 1, pp.47-71, 1980.

[10] Kim, K. (1988), *Equilibrium Business Cycle Theory in Historical Perspective*, Cambridge, University Press, Cambridge, U.K.
[11] Bloomfield, P. (1976), *Fourier Analysis of Time Series : An Introduction*, John Wiley & Sons, New York.
[12] Brillinger, D. R. (1976), Some history of the data analysis of time series in the United States, *in* (Owen 1976).
[13] Brillinger, D. R. (1975), *Time Series : Data Analysis and Theory*, Holt, Rinehart and Winston, New York.
[14] Funkhauser, H. G. (1936), ‘A note on a tenth century graph’, Osiris 1 (January).
[16] Pierre Yves Glorennec, Algorithmes d’apprentissage pour systèmes d’inférence floue, Département d’informatique INSA de Rennes/RISA.

[17] Jelena. Godjevac, idées nettes sur la logique floue, Presses Polytechniques et Universitaires Romandes, CH-1015 Lausanne, 1999, Première Edition.
[18] H. Berenji, ‘‘ Fuzzy Logic Controllers: in An Introduction to Fuzzy Logic Applications in Intelligence System ’’, Kluwer Academic Pulishers, pp. 69-96,1992.

[19] H. Buhler, Réglage par la Logique Floue, Presses Polytechniques et Universitaires Romandes, Lausanne, 1993.

[20] B. Kosko, S. Isaka, ‘‘la logique floue et ses Applications’’, Pour la Science, No.191, September 1993.

[26] Hagan, M.T. et M.B. Menhaj (1994). Training feedforward networks with marquardt algorithm. IEEE Transaction On Neural Networks **5**(6), 989–993.
[21] Burns, R.S. (2001). Advanced Control Engineering. butterworth heinemann ed.
[22] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics

of Control, Signals, and Systems **2**, 303–314.
[23] Hornik, K., M. Stinchcombe et H. White (1989). Multilayer feedforward networks are universal approximators. Neural Networks **2**(5), 359–366.
[24] Sastry, P.S., G. Santharam et K.P. Unnikrishnan (1994). Memory neuron networks for identification and control of dynamical systems. IEEE Transactions on Neural Networks (2), 306–319.
[25] Magoulasb, G.D., M.N. Vrahatis et G.S. Androulakis (1999). Improving the convergence
of the backpropagation algorithm using learning rate adaptation methods. Neural Computation **11**, 1769–1796.
[27] Baker J.E, ‘‘Adaptive selection Methods for Genetic Algorithm’’, Proceedings of the First International Conference on Genetic Algorithms, pp.101-111, 1985.

[28] J.J. Holland, Adaptation in Natural and artificial systems (second edition), M.I.T Press, 1992.

[29] Goldberg.D.E, Genetic Algorithms in Search, Optimisation, and Machine Learning, Addition-Wesly, Reading (MA), 1989.

[30] Man, K. F. et W.A. Halang (1997). Genetic algorithms for control and signal processing. In: Proceedings of the 23rd International Conference on Industrial Electronics,Control and Instrumentation. Vol. 4. New Orleans, LA, USA. pp. 1541–1555.
[31] H. P. Schwefel, Numerical Optimization of Computer Models, John Wily and Sons, 1981.
[32] F. Hoffmeister and T. Back, ‘‘Genetic Algorithms and Evolutionary Strategies: Similarities and Differences’’, in Parallel Problem Solving From Nature, Manner and Schwefel Editors, Springer Verlag, pp.455-569, 1991

[33] L. Davis, Handbook of Genetic Algorithms, Van Norstrand Reinhold, New-York, 1991.

[34] J. M. Renders, Algorithmes Génétiques et Réseaux de Neurones, Hermés, Paris, 1995.

[35] D.E.Goldberg, Algorithmes Génétiques: Explotation, Optimisation et Apprentissage Automatique, Edition Addison-Wesley, 1994.

[36] K. Jong. De, ‘‘Learning with Genetic Algorithms: An Overview’’, Machine Learning, Vol.3, pp.121-138, 1988.

[37] J.J. Grefenstte and J.K. Baker, ‘‘How Genetic Algorithms Work: A Critical look at implicit parallelism’’, Proceedings of the Third International Conference on Genetic Algorithms, pp20-27, 1989.

[38] N.P. Belfliore and A. Esposito, ‘‘Theoretical and Experimental Study of Crossover Operators of Genetic Algorithms’’, Journal of Optimization Theory and Application, Vol.99, pp271-302, November 1998.

[39] http://pacific.commerce.ubc.ca/xr
[40] R. Mechgoug, N. Titaouine, Exchange Rate Prediction using Neural – Genetic Model, Journal of Electrical and Electronics Engineering, Volume 5, Number 2, October 2012
[41] U. Brunelli, V. Piazza and L. Pignato, ‘‘ Hourly Forecasting of SO_2 Pollutant Concentration Using An Elman Neural Network”, Springer-Verlag Berlin Heidelberg, LNCS 3931, pp,65-69,2006.

[42] F. YU, Z. Wenfang and S. Dezhi, “Ozone concentration forecast method based on genetic algorithm optimized back propagation neural networks and support machine data classification”, Atmospheric Environment 45, Elsevier, pp, 1979-1985, 2011.

[43] M.B. Biljana and S. Mile, "Prediction of Missing Data of ozone concentration Using Support Vector machines and Radial Basis Neural Networks, Informatica31, pp425-430, 2007.

[44] Hyndman, R.J. *Time Series Data Library*, http://data.is/TSDLdemo
[45] R. Mechgoug, N. Golea and A. Taleb-Ahmed, Times Series Prediction to Basis of a Neural Network Conceived by a Real Genetic Algorithm, Journal Computer Technology and Application, Volume2 , Numbre1,2011
[46] L.W. Lee and L.H. Wang, ''Temperature Prediction and TAIFEX Forecasting Based on Fuzzy Logical Relationships and Genetic Algorithms'', expert systems with applications, pp 1-12, Elsevier 2006 Ltd.
[47] Y. Radhika and M. Shashi, "Atmospheric Temperature Prediction using Support Vector Machines'', International Journal of Computer Theory and Engineering, Vol. 1, No. 1,pp 55-58, April 2009.
[48] B. Smith and R. McClendon, "Improving Air Temperature Prediction with Artificial Neural Networks", Georgia Agricultural Experiment Stations Hatch Projects GE00877, 14 March 2006.
[50] M. Hayati and Z. Mohebi, ''Application of Neural Network for Temperature Forecasting '', World Academy of Science, Engineering and Technology, 28 2007.

[51] R. Mechgoug, A. Taleb Ahmed, and L. Cherroun, Optimization of Neural Predictor for Air Pollution, Proceedings of the World Congress on Engineering , Vol II, WCE 2012, London, U.K.

[52] AK. Dhamija and VK. Bhalla ''Financial Time Series Forecasting: Comparison of Neural Networks and ARCH Models'', Inter. Research Journal of Finance and Economics Vol .49, pp.185-202, 2010.
[53] Rumelhart, D.E. et J. L. McClelland (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Vol. 1. M.I.T. Press.
[54] Bhansali, R. J. (1980). Autoregressive and window estimates of the inverse correlation function. Biometrika 67, 551–66.
[55] Bhansali, R. J. (1983a). The inverse partial correlation function of a time series and its applications. J. Multivar. Anal. 13, 310–27.
[56] Cleveland, W. S. (1972). The inverse autocorrelations of a time series and their applications. Technometrics. 14, 277–98.
[57] Liska, J. et S.S. Melsheimer (1994). Complete design of fuzzy logic systems using genetic algorithms. In: *Proceedings of the Third IEEE International Conference on Fuzzy Systems*. Vol. 2. Orlando, FL, USA. pp. 1377U″ –1382.
[58] Acosta, G. et E. Todorovich (2003). Genetic algorithms and fuzzy control: a practical synergism for industrial applications. *Computers in Industry* **52**, 183–195.
[60]Karr, C.L. (1991). Applying genetics to fuzzy logic. *AI Expert* **6**(3), 38U″ 43.
[61] Herrera, F., M. Lozano et J.L. Verdegay (1998). A learning process for fuzzy control rules using genetic algorithms. *Fuzzy Sets and Systems* **100**, 143–158.
[62] Chen, C.C. et C.C. Wong (2002). Self-generating rule-mapping fuzzy controller design using a genetic algorithm. In: *IEE Proceedings on Control Theory and Applications*. Vol. 49. pp. 143–148.
[63] Belarbi, K., F. Titela, W. Bourebiaa et K. Benmahammed (2005). Engineering applications of artificial intelligence. *Design of mamdani fuzzy logic controllers with rule base minimisation using genetic algorithm* **18**, 875–880.
[64] Homaifar, A. et E. MCCormick (1993). Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms. *IEEE transactions on fuzzy systems* **3**(2), 129–139.

[65] Lee, C.C. (1990). Fuzzy logic in control systems: Fuzzy logic controller, part ii. *IEEE Transactions on Systems, Man. and Cybernetics* **20**(2), 419–435.
[66] Ministère de l'Aménagement de territoire et de l'environnement, Plan Régional pour la Qualité de l'Air en région Centre , République Française, 14 JANVIER 2002