Mars 2018


# Rapport de Contrat

**Société TéSA**

par

**Jean-Yves Tourneret et Herwig Wendt**

## ÉTAT DE L'ART SUR LES TECHNIQUES DE PRÉDICTION DE TRAFIC
## RÉFÉRENCE R-S17/TC-0008-031

# Table des matières

# Chapitre 1

# Introduction

Un certain nombre de progrès ont été réalisés dans le domaine de la prédiction de trafic pour améliorer le traitement des problématiques comme la gestion dynamique de « bande passante », une meilleure prise en compte de la congestion ou la planification de l'allocation des ressources de communication.

L'objectif de l'étude est d'analyser les performances d'un ensemble de techniques de prédiction de trafic réseaux qui s'avèrent être les mieux adaptées au contexte d'un segment sol Satcom VHTS et ceci à des fins d'optimisation de l'allocation des ressources de communication court terme, mais aussi d'avoir un aperçu de ce qui se fait aujourd'hui dans le domaine des techniques de planification long terme des ressources. L'étude se concentrera donc particulièrement sur les techniques d'analyse de prédiction de trafic à court terme, mais permettra de dégager des stratégies d'analyse à plus long terme.

De nombreuses méthodes de prédiction ont d'ores et déjà été proposées dans la littérature pour la prédiction du trafic [1, 2]. Ce document présente un état de l'art de ces méthodes, avec une attention particulière pour l'application considérée dans cette étude. En suivant la nomenclature utilisée dans [1], nous répartissons ces méthodes en trois grandes familles : les méthodes de prédiction linéaires, les méthodes de prédiction non-linéaires et les méthodes hybrides. Nous résumons dans cette partie le principe de ces familles de méthodes.

# Chapitre 2

# Méthodes de prédiction linéaires

L'idée de ces méthodes est de supposer que la série temporelle observée est stationnaire de la forme AR, MA ou ARMA et d'appliquer des techniques classiques de prédiction linéaire utilisées en statistique [3] ou en traitement du signal [4] pour ce type de série temporelle. Nous résumons dans cette partie quelques-unes de ces méthodes.

## 2.1 Modèles AR, MA et ARMA

### 2.1.1 Principe

Pour simplifier, on limite cette partie au cas des modèles autorégressifs (AR) mais toute cette théorie se généralise aux modèles moving average (MA) et autorégressifs à moyenne ajustée (ARMA) sans difficulté. Dans le cas d'un modèle AR, on suppose que la série temporelle observée notée $x(n)$ vérifie une équation récursive de la forme

$$x_n = -\sum_{k=1}^{p} a_k x_{n-k} + e_n \tag{2.1}$$

qui traduit le fait que la valeur de la série temporelle à l'instant $n$ dépend de ses $p$ valeurs passées à un terme d'erreur près noté $e(n)$ qui est supposé être un bruit blanc (suite de variables aléatoires indépendantes de moyennes nulles) de variance $E[e_n^2] = \sigma_e^2$. Il existe de multiples méthodes permettant d'estimer les coefficients autorégressifs $a_k$ pour une série temporelle donnée, qui sont décrites dans des ouvrages comme [3] et [4] (voir par exemple [3, Chap. 5]). Une fois que les paramètres $a_k, k = 1, ..., p$ ont été estimés, le meilleur prédicteur de $x(n)$ en fonction de ses $p$ valeurs passées s'écrit

$$\widehat{x}_n = -\sum_{k=1}^{p} \widehat{a}_k x_{n-k}$$

où $\widehat{\boldsymbol{a}} = (\widehat{a}_1, ..., \widehat{a}_p)^T$ est l'estimateur du vecteur $\boldsymbol{a} = (a_1, ..., a_p)^T$. Ce prédicteur admet donc une forme très simple. Le seul problème est de s'assurer que la série temporelle observée est stationnaire et ensuite d'estimer les coefficients AR à partir du passé de la série temporelle. Ce type de série temporelle a reçu beaucoup d'intérêt dans la littérature car on sait que la densité spectrale de puissance (DSP) de toute série temporelle stationnaire peut être approchée avec la précision voulue par la DSP d'une série temporelle AR.

*Remarques*

— Si on veut effectuer une prédiction ARMA, il suffit de remplacer l'équation (2.1) par

$$x_n = -\sum_{k=1}^{p} a_k x_{n-k} + \sum_{k=0}^{q} b_k e_{n-k} \tag{2.2}$$

avec $b_0 = 1$ et estimer les vecteurs paramètres $\boldsymbol{a}$ et $\boldsymbol{b} = (b_1, ..., b_q)^T$. Le prédicteur ARMA de $x(n)$ dépend alors des valeurs passées de la série temporelle $x_{n-1}, x_{n-2}, ...$ et des valeurs des vecteurs $\boldsymbol{a}$ et $\boldsymbol{b}$ qu'il convient d'onc d'estimer.

— Dans le domaine de la prédiction de trafic internet, plusieurs auteurs ont revendiqué l'utilisation de modèles ARMA comme [5–8] (liste non exhaustive).

### 2.1.2 Estimation des paramètres AR

Une méthode classique d'estimation des paramètres AR d'une série temporelle auto-régressive est d'utiliser les équations de Yule-Walker reliant la fonction d'autocorrélation $r(k) = E[x(n)x(n-k)]$ aux paramètres AR. Pour cela, on peut multiplier l'équation (2.1) par $x_{n-i}$ et calculer l'espérance des deux membres. On obtient

$$r(i) = E[x_n x_{n-i}] = -\sum_{k=1}^{p} a_k E[x_{n-k} x_{n-i}] + E[e_n x_{n-i}].$$

On peut montrer que

$$E[e_n x_{n-i}] = \begin{cases} 0 \text{ si } i > 0 \\ E[e_n^2] = \sigma_e^2 \text{ si } i = 0 \end{cases} \tag{2.3}$$

où $\sigma_e^2$ est la variance du bruit $e(n)$, ce qui permet d'obtenir le système suivant

$$\boldsymbol{r} = -\boldsymbol{R}\boldsymbol{a}$$

avec $\boldsymbol{r} = [r(1), ..., r(p)]^T$ et

$$\boldsymbol{R} = \begin{pmatrix} r(0) & r(1) & \cdots & r(p-1) \\ r(1) & r(0) & \cdots & r(p-2) \\ \vdots & \vdots & \vdots & \vdots \\ r(p-1) & r(p-2) & \cdots & r(0) \end{pmatrix}.$$

Le vecteur $\boldsymbol{r}$ et la matrice $\boldsymbol{R}$ dépendent des autocorrélations du signal $x(n)$ qui peuvent être estimées à l'aide d'estimateurs standards. Le vecteur $\boldsymbol{a}$ peut alors être obtenu par résolution du système linéaire

$$\widehat{\boldsymbol{r}} = -\widehat{\boldsymbol{R}}\boldsymbol{a}.$$

D'autres méthodes d'estimation du vecteur $\boldsymbol{a}$ ont été étudiées dans la littérature. On pourra consulter [4] pour plus de détails concernant ces méthodes.

*Remarques*

— Les méthodes de prédiction linéaires ARMA sont parfois utilisées après un prétraitement des données. Par exemple, dans [9], les auteurs proposent un pré-traitement à base de différences finies des nombres de paquets reçus par seconde et d'une transformation logarithmique ("log-returns"). Le signal pré-traité est ensuite soumis à une prédiction ARMA.

— Les résultats de [5] montrent qu'une prédiction effectuée avec un modèle ARMA(1,1) (i.e., avec $p = 1$ et $q = 1$ dans (2.2), peut être suffisante pour certaines applications, et surtout que les performances de prédiction de s'améliorent pas de manière significative avec un modèle ARIMA (voir partie suivante).

— Les deux articles qui nous semblent les plus pertinents sur ce sujet sont [5] et [9].

## 2.2 Modèles ARIMA et FARIMA

### 2.2.1 Principe

En fonction du type de données considérées ou de l'échelle à laquelle on analyse ces données, les séries temporelles observées ne peuvent pas toujours être considérées comme stationnaires. Dans ce contexte, on peut utiliser des modèles linéaires plus évolués que les modèles ARMA pour définir des prédicteurs. Par exemple, on peut calculer des différences finies à l'ordre un de la série temporelle observée $x(n)$ définies par

$$y_n = x_n - x_{n-1} \tag{2.4}$$

ou des différences finies d'ordres supérieurs

$$y_n = x_n - 2x_{n-1} + x_{n-2} \tag{2.5}$$

et ensuite chercher le modèle ARMA qui s'ajuste aux données $y(n)$. On obtient alors un modèle ARIMA (autorégressif intégré à moyenne ajustée) qui a également été considéré pour la prédiction des données internet [5, 10, 11]. Certaines séries temporelles sont non stationnaires tandis que leurs différences finies le sont. Ceci se comprend facilement si on note que les équations (2.4) et (2.5) correspondent à des équations de filtrage linéaire. Le modèle ARIMA est donc plus général que le modèle ARMA et s'adapte à plus de signaux rencontrés dans les applications pratiques. Il permet également de supprimer les effets saisonniers en définissant des différences plus élaborées que (2.4) et (2.5) (voir par exemple [10, 12]).

Pour définir la notion de série temporelle ARIMA, il est classique d'introduire l'opération de différentiation $D$ tel que $D^k x_n = x_{n-k}$. On dit alors que $x_n$ est une série temporelle ARIMA(p,d,q) si $y_n = (1-D)^d x_n$ est une série temporelle ARMA(p,q), c'est-à-dire si

$$y_n = -\sum_{k=1}^{p} a_k y_{n-k} + \sum_{k=0}^{q} b_k e_{n-k}$$

avec

$$y_n = (1-D)^d x_n.$$

Pour terminer, notons que certains auteurs [13, 14] ont considéré des modèles encore plus généraux que le modèle ARIMA, en autorisant des opérations de dérivation fractionnaire conduisant aux modèles FARIMA, qui implicitement considèrent le fait que la série temporelle $x(n)$ a des propriétés de longue dépendance.

### 2.2.2 Prédiction

Les techniques de prédiction utilisées pour les séries temporelles ARMA se généralisent aux séries temporelle ARIMA. Elles nécessitent d'estimer l'ordre de différentiation $d$ et les vecteurs paramètres $\boldsymbol{a}$ et $\boldsymbol{b}$. On montre dans cette partie comment procéder pour l'exemple simple d'une série ARIMA(1,1,0).

$$y_n = x_n - x_{n-1}$$

et

$$y_n = ay_{n-1} + e_n$$

où $e_n$ est un bruit blanc.

**Prédiction à l'ordre** 1

On a

$$x_{n+1} = x_n + y_{n+1}$$

Donc

$$\widehat{x}_{n+1} = x_n + \widehat{y}_{n+1}$$

où $\widehat{y}_{n+1}$ est la prédiction d'une série temporelle AR(1), i.e.

$$\widehat{y}_{n+1} = ay_n$$

d'où

$$\widehat{x}_{n+1} = x_n + ay_n = x_n + a[x_n - x_{n-1}]$$

On en déduit la formule de prédiction suivante

$$\boxed{\widehat{x}_{n+1} = [1+a]x_n - ax_{n-1}.} \tag{2.6}$$

**Prédiction à l'ordre** 2

En suivant la démarche précédente

$$\widehat{x}_{n+2} = \widehat{x}_{n+1} + \widehat{y}_{n+2}$$

avec

$$\widehat{y}_{n+2} = a\widehat{y}_{n+1} = a^2 y_n$$

d'où la prédiction

$$\widehat{x}_{n+2} = x_n[1+a] - ax_{n-1} + a^2 y_n$$

avec $y_n = x_n - x_{n-1}$ ce qui permet d'obtenir

$$\boxed{\widehat{x}_{n+2} = [1+a+a^2]x_n - a(a+1)x_{n-1}.} \tag{2.7}$$

### 2.2.3 Application à la prédiction de trafic

— Pour comprendre la théorie des séries temporelles ARIMA, on pourra se reporter à [15, 16].

— L'application des modèles ARIMA aux séries temporelles de vidéos a été proposée dans [10] où un modèle est proposé pour un groupe d'images (modèle GOP-ARIMA). On pourra aussi consulter [11] pour un article très général sur l'utilisation de séries temporelles pour la prédiction du trafic ou [17] pour l'application au trafic routier.

— Modèle ARIMA appliqué aux coefficients issus d'une décomposition en ondelettes pour la prédiction de trafic [18].

# Chapitre 3

# Méthodes de prédiction non-linéaires

## 3.1 Modèles ARCH, GARCH et ARIMA-GARCH

### 3.1.1 Définition

La principale motivation pour utiliser des modèles comme le modèle GARCH est que la variance des données ne peut pas toujours être considérée comme constante au cours du temps (comme c'est le cas pour les modèles ARMA ou ARIMA), en particulier pour les données issues du trafic internet [19–22]. Les modèles ARCH et GARCH sont définis par

$$x_n = \sigma_n e_n \tag{3.1}$$

où $\{e_n\}_{n=0,1,...}$ est une suite de variables aléatoires de moyenne nulle et de variance 1 et où le terme $\sigma_n$ permet d'obtenir une série temporelle dont la variance varie dans le temps. Suivant la définition de $\sigma_n$, on obtient le modèle GARCH ou le modèle ARCH (qui est un cas particulier de modèle GARCH avec $b_1 = ... = b_q = 0$). Pour le modèle GARCH, on a

$$\sigma_n^2 = a_0 + \sum_{i=1}^{p} a_i x_{n-i}^2 + \sum_{j=1}^{q} b_j \sigma_{n-j}^2$$

avec $a_0 > 1, a_i \geq 0, b_j \geq 0$ et

$$\sum_{i=1}^{p} a_i + \sum_{j=1}^{q} b_j < 1.$$

On suppose de plus que $e_n$ est indépendante des variables $x_{n-i}, i \geq 1$.

Les modèles ARCH et GARCH sont généralement classés dans la famille des modèles non-linéaires car $x_n = \sigma_n e_n$ et $\sigma_n$ dépend non linéairement de $x_{n-1}, ..., x_{n-p}$. Il existe dans

la littérature des techniques qui permettent d'estimer les paramètres des modèles ARCH ou GARCH, comme cela est expliqué dans la section suivante. Notons que ces modèles ont plutôt été utilisés pour l'analyse des séries temporelles financières [23, 24].

### 3.1.2 Représentation ARMA du carré d'un processus GARCH

Posons $\eta_n = x_n^2 - \sigma_n^2$. En utilisant la définition de la variance conditionnelle $\sigma_n^2$, on a

$$x_n^2 = \sigma_n^2 + \eta_n = a_0 + \sum_{i=1}^{p} a_i x_{n-i}^2 + \sum_{j=1}^{q} b_j \sigma_{n-j}^2 + \eta_n.$$

Si on note $m = \max(p, q)$ et qu'on suppose $a_i = 0$ pour $i > p$ et $b_j = 0$ pour $j > q$, on obtient

$$x_n^2 = a_0 + \sum_{i=1}^{p} (a_i + b_i - b_i) x_{n-i}^2 + \sum_{j=1}^{q} b_j \sigma_{n-j}^2 + \eta_n \tag{3.2}$$

$$= a_0 + \sum_{i=1}^{m} (a_i + b_i) x_{n-i}^2 + \sum_{j=1}^{q} b_j (\sigma_{n-j}^2 - x_{n-j}^2) + \eta_n \tag{3.3}$$

$$= a_0 + \sum_{i=1}^{m} (a_i + b_i) x_{n-i}^2 - \sum_{j=1}^{q} b_j \eta_{n-j} + \eta_n \tag{3.4}$$

ce qui est l'équation de récurrence d'un modèle ARMA(m,q). Cette récurrence peut être utilisée pour estimer les paramètres $a_i, i = 0, ..., p$ et $b_j, j = 1, ..., q$.

### 3.1.3 Prédiction

Comme $e_n$ est indépendant de $x_{n-k}, \forall k \geq 1$, on a

$$E[x_n | x_{n-1}, x_{n-2}, ...] = \sigma_n E[e_n] = 0$$

donc la prédiction de $x_n$, étant donné le passé $\{x_i, i < n\}$, vaut 0. L'intérêt des modèles GARCH n'est donc pas dans la prédiction de la série temporelle elle-même, mais plutôt de la prédiction du carré de cette série temporelle. En effet

$$E[x_n^2 | x_{n-1}, x_{n-2}, ...] = \sigma_n^2 E[e_n^2] = \sigma_n^2.$$

De même $E[x_{n+i}^2 | x_{n-1}, x_{n-2}, ...]$ se calcule de manière récursive (voir par exemple [25, p. 53]). Ces quantités nous permettent de déterminer des intervalles de confiance pour les prédictions de $x_n$ et $x_{n+k}, k > 0$.

### 3.1.4 Modèles ARMA-GARCH et ARIMA-GARCH

On dit qu'une série temporelle $x_n$ est ARMA-GARCH lorsque $x_n$ suite une récursion ARMA, c'est-à-dire

$$x_n = -\sum_{k=1}^{p} a_k x_{n-k} + \sum_{k=0}^{q} b_k e_{n-k} \tag{3.5}$$

avec une innovation $e_n$ qui est une série temporelle GARCH, i.e., telle que

$$e_n = \sigma_n \eta_n$$

avec

$$\sigma_n^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i e_{n-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{n-j}^2.$$

Pour simplifier, considérons une série $x_n$ AR(1) avec une innovation GARCH(1,1), i.e.,

$$x_n = -a_1 x_{n-1} + e_n, \quad e_n = \sigma_n \eta_n \tag{3.6}$$

avec

$$\sigma_n^2 = \alpha_0 + \alpha_1 e_{n-1}^2 + \beta_1 \sigma_{n-1}^2.$$

On obtient alors

$$x_{n+h} = -a_1 x_{n+h-1} + e_{n+h} = e_{n+h} - a_1 e_{n+h-1} - a_1 x_{n+h-2} = ...$$

d'où

$$x_{n+h} = e_{n+h} - a_1 e_{n+h-1} - ... + (-a_1)^h e_n + (-a_1)^{h+1} x_{n-1}$$

ce qui permet de déterminer le prédicteur

$$\widehat{x}_{n+h} = E\left[x_{n+h} | x_u u < n\right] = (-a_1)^{h+1} x_{n-1}.$$

Comme on peut l'observer ce prédicteur est le même que dans le cas d'un modèle AR(1). En revanche, la variance de $x_{n+h}|x_u, u < n$ dépend des paramètres $\alpha_i$ et $\beta_j$ (voir expression dans [25, p. 56] pour une série $x_n$ AR(1) avec une innovation GARCH(1,1)).

La généralisation aux séries ARIMA-GARCH est immédiate après avoir appliqué une différentiation à la série temporelle $x_n$.

### 3.1.5 Application à la prédiction de trafic

— L'utilisation des modèles GARCH pour la prédiction de trafic a été préconisée dans [19] où les performances de prédiction sont meilleures que celles obtenues avec des modèles ARMA, ARIMA ou ARIMA-GARCH. Notons que la forme du prédicteur utilisée n'est pas explicitée dans l'article.

— L'utilisation de modèles ARIMA-GARCH (modèles ARIMA dont le bruit est une série temporelle GARCH) a fait l'objet de plusieurs études comme [20–22]. Peu de détails concernant les méthodes de prédiction utilisées sont données dans ces articles.

## 3.2 Réseaux de neurones

L'idée des méthodes de prédiction basées sur les réseaux de neurones est de chercher un prédicteur de la série temporelle $x_n$ (observation à l'instant $n$) en fonction de ses $p$ valeurs précédentes concaténées dans le vecteur $\boldsymbol{x}_{n-1} = (x_{n-1}, x_{n-2}, ..., x_{n-p})^T$ à l'aide d'une relation non-linéaire de la forme

$$\widehat{x}_n = g\left(\boldsymbol{x}_{n-1}\right) = g\left(x_{n-1}, x_{n-2}, ..., x_{n-p}\right)$$

qui est la sortie d'un réseau de neurones d'entrée $\boldsymbol{x}_{n-1}$ comportant plusieurs couches de neurones. La structure générale d'un réseau de neurones (appelé aussi perceptron multicouches) avec $p$ entrées, 3 couches de neurones et $q$ sorties est représentée sur la figure 3.1.

Pour la prédiction, il suffit de mettre à l'entrée de ce réseau $x_1 = x_{n-1}, ..., x_p = x_{n-p}$ et ne considérer qu'une sortie $y_1 = \widehat{x}_n$. Chaque noeud de ce réseau calcule une fonction non linéaire de ses entrées. Par exemple, les sorties des noeuds de la première couche s'écrivent

$$f\left(\sum_{k=1}^{p} a_{ki} x_{n-k} - b_i\right), \quad i = 1, ..., N_1$$

où $N_1$ est le nombre de neurones de la première couche, le coefficient $a_{ki} \in \mathbb{R}$ est le poids de la première couche allant de l'entrée $x_k$ au noeud $i$, $b_i \in \mathbb{R}$ est un réel (appelé biais ou offset) associé au noeud $i$ de la première couche et $f$ est une non-linéarité. Un exemple classique de non-linéarité utilisée en pratique est la fonction sigmoide définie par

$$f(t) = \frac{1}{1 + e^{-at}}, \quad t \in \mathbb{R}$$

avec $a > 0$. Il existe des algorithmes qui permettent d'estimer les coefficients $a_{ki}$ et $b_i$ de chaque couche $i$, à partir d'un ensemble d'apprentissage, c'est-à-dire un ensemble d'entrées et de sorties [26]. Dans le cas de la prédiction, il suffit de créer cet ensemble d'apprentissage à l'aide des éléments de la série temporelle situés aux instants précédant l'instant courant $n$. Après avoir déterminé ces coefficients, le prédicteur de $x_n$ s'écrit

$$\widehat{x}_n = g\left(\boldsymbol{x}_{n-1}\right).$$

Ce type de prédicteur a été utilisé dans de nombreuses applications, notamment pour la prédiction de données vidéo [27] ou de trafic internet [2, 28, 29].

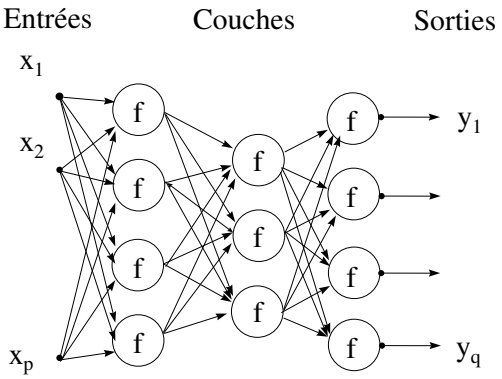



FIGURE 3.1 – Exemple de réseau de neurons à $p$ entrées et $q$ sorties.

### 3.2.1 Application à la prédiction de trafic

— L'utilisation de réseaux de neurones pour la prédiction de trafic vidéo a été étudiée dans [27]. Dans cet article, les auteurs montrent clairement l'intérêt d'une prédiction non-linéaire par rapport à une prédiction linéaire. Différentes structures de réseaux de neurones sont considérées dans cet article : 1) perceptron multi-couches, 2) fonctions radiales de base, 3) réseau de neurones récurrent, 4) Modèle linéaire MA + perceptron multi-couches. Les différentes structures donnent des résultats de prédiction comparables avec une légère préférence pour les réseaux récurrents. Le fait d'utiliser un modèle linéaire avant le réseau de neurones permet d'accélérer la convergence de l'apprentissage. Le nombre de couches et de noeuds par couche a été déterminé par validation croisée.

— Les auteurs de [28, 29] ont proposé d'utiliser une stratégie de prédiction non-linéaire basée sur la somme d'un terme linéaire (de type autorégressif) et d'un terme non-linéaire construit comme la sortie d'un réseau de neurones et ont obtenu de meilleures performances de prédiction avec cette méthode qu'avec plusieurs autres techniques de prédiction comme la méthode ARIMA.

## 3.3 Machines à vecteurs supports

Le domaine de la reconnaissance des formes a apporté des éléments de solution convaincants aux problèmes de classification et de prédiction au travers des méthodes à noyaux reproduisants [30]. Ces méthodes sont pour la plupart issues d'algorithmes à l'origine linéaires auxquels on a pu appliquer deux principes clés : l'astuce du noyau et le théorème de représentation [31]. Les méthodes à noyaux permettent à présent d'élaborer des modèles linéaires généralisés dans des espaces transformés de dimension importante, voire infinie, sans qu'aucun calcul n'y soit effectué explicitement. Dans le domaine du traitement des séries temporelles en particulier, pour la prédiction ou le débruitage notamment, plusieurs approches convaincantes ont été proposées [32–35]. Etant donné un espace fonctionnel $H$, le problème consiste à rechercher une fonction de $H$ minimisant la somme des $p$ erreurs quadratiques entre les réponses désirées du modèle et les sorties effectivement obtenues pour le $i$ème vecteur d'entrée

$$\min_{\psi \in H} \sum_{i=1}^{p} \left[ d_i - \psi(\boldsymbol{v}_i) \right]^2$$

où $i = 1, ..., p$ et $p$ est le nombre de vecteurs $(d_i, \boldsymbol{v}_i)$ de la base d'apprentissage. Pour un problème de prédiction, on a $\boldsymbol{v}_i = [x_{i-1}, ..., x_{i-p}]^T$ et $d_i = x_i$, où $\boldsymbol{x}$ est un signal de la base d'apprentissage. Lorsque $H$ désigne un espace de Hilbert à noyau reproduisant, en vertu du théorème de représentation, on sait que la solution du problème ci-dessus peut s'écrire sous la forme [20]

$$\psi(.) = \sum_{j=1}^{p} \alpha_j \kappa(., \boldsymbol{v}_j)$$

où $\kappa$ est le noyau reproduisant associé à $H$, et $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_p)^T$ est un vecteur de coefficients à estimer. Dans ces conditions, le problème à résoudre devient alors

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{d} - \boldsymbol{K}\boldsymbol{\alpha}\|$$

où $\boldsymbol{K}$ est la matrice de Gram dont les éléments sont $\kappa(\boldsymbol{v}_i, \boldsymbol{v}_j)$, et $\boldsymbol{d} = (d_1, ..., d_p)^T$ est le vecteur des sorties désirées correspondantes. Parmi les noyaux reproduisants les plus utilisés, en raison de leur bonne efficacité dans la majorité des situations, on compte le noyau gaussien défini par

$$\kappa(\boldsymbol{v}_i, \boldsymbol{v}_j) = \exp\left( \frac{\|\boldsymbol{v}_i - \boldsymbol{v}_j\|^2}{2\sigma^2} \right)$$

et le noyau de Laplace

$$\kappa(\boldsymbol{v}_i, \boldsymbol{v}_j) = \exp\left( \frac{\|\boldsymbol{v}_i - \boldsymbol{v}_j\|}{\sigma^2} \right).$$

Après avoir estimé le vecteur de coefficient $\boldsymbol{\alpha}$, le prédicteur de $x_n$ est défini par

$$\widehat{x}_n = \psi(\boldsymbol{x}_{n-1}) = \sum_{j=1}^{p} \widehat{\alpha}_{\omega_j} \kappa(\boldsymbol{x}_{n-1}, \boldsymbol{v}_j)$$

avec $\boldsymbol{x}_{n-1} = (x_{n-1}, ..., x_{n-p})^T$.

Il est important de noter que des méthodes d'apprentissage séquentielle ont été développées pour traiter (prédire) les données au fil de l'eau. Elles consistent toutes à considérer un modèle de taille réduite $m$ tel que

$$\psi(.) = \sum_{j=1}^{m} \alpha_{\omega_j} \kappa(., \boldsymbol{v}_{\omega_j})$$

où $\left\{\kappa\left(., \boldsymbol{v}_{\omega_j}\right)\right\}_{j=1}^{m}$ désigne un dictionnaire dont les éléments sont choisis à l'aide d'un critère de parcimonie. Les méthodes d'identification de modèles à noyau reposent alors sur deux étapes, répétées à chaque itération pour un nouveau vecteur de données $(\boldsymbol{v}_i, d_i)$ :

— Contrôle de l'ordre du modèle réduit grâce au critère de parcimonie choisi, par ajout et/ou retrait d'éléments du dictionnaire.

— Mise à jour des paramètres $\alpha_{\omega_j}$ au sens du critère des moindres carrés. Les approches décrites dans la littérature se distinguent par la nature des choix techniques faits pour chacune de ces deux étapes. On citera les algorithmes NORMA [35] (Parcimonie par fenêtre temporelle glissante, et algorithme de mise à jour des poids de type gradient stochastique), KNLMS [32] (Parcimonie par critère de cohérence du dictionnaire, et algorithme de mise à jour des poids de type gradient stochastique), SSP [33] (Parcimonie par projection, et algorithme de mise à jour des poids de type gradient stochastique fonctionnel) et KRLS [34] (Parcimonie par critère de dépendance linéaire, et algorithme de mise à jour des poids de type moindres carrés récursifs).

### 3.3.1 Application à la prédiction de trafic

Les méthodes à noyaux reproduisants exposées ci-dessus ont été appliquées à la prédiction du trafic internet dans les référence suivantes

— Dans [36] (a priori très bon article), les auteurs proposent de construire une méthode de prédiction à base de machines à vecteurs supports qui minimise le coût global sur $n$ données d'apprentissage défini par

$$C \sum_{i=1}^{n} L\left[f(\boldsymbol{x}_i) - y_i\right] + \|\boldsymbol{\beta}\|^2$$

avec $f(\boldsymbol{x}) = \boldsymbol{\beta}^T \phi(\boldsymbol{x}) + \beta_0$ et

$$L[f(\boldsymbol{x}), y)] = \begin{cases} 0 \text{ si } |f(\boldsymbol{x}) - y| < \epsilon \\ |f(\boldsymbol{x}) - y| - \epsilon \text{ sinon} \end{cases}$$

où le terme $\|\boldsymbol{\beta}\|^2$ est utilisé pour lisser les résultats et la constante $C$ est déterminée par validation croisée. Le noyau utilisé est celui associé aux fonctions radiales de base. Les features utilisées pour la prédiction sont **le délai dans la file d'attente**, **la fréquence ou durée des pertes** et **le temps aller-retour moins son minimum**. L'apprentissage est effectué à l'aide d'une centaine de vecteurs de features et les résultats de prédiction sont bons lorsque l'il y a peu de changement dans le trafic de fond ou lorsqu'on laisse le temps à TCP de s'adapter.

— Dans [37], les auteurs appliquent une méthode de prédiction à base de machines à vecteurs supports issue de [38] à la prédiction du "travel time index". Les résultats obtenus sont meilleurs que 5 autres méthodes de prédiction (ARMA, filtre de Kalman, moyenne des éléments passés, réseau de neurones à fonctions radiales de base). Notons que le vecteur de features utilisé contient les valeurs passées du paramètre d'intérêt et une information plus globale sur le passé appelée "historical information" qui n'est pas vraiment définie.

— Dans [39], les auteurs s'intéressent à la prédiction du trafic routier (nombre de véhicules par heure) à l'aide d'une méthode SVM avec un noyau qui permet de prendre en compte les tendances saisonnières de ce trafic.

## 3.4 Méthodes hybrides

D'après [1], les méthodes hybrides utilisent conjointement plusieurs méthodes de prédiction et ont montré de bonnes performances pour l'analyse de données internet. Les combinaisons de méthodes qui ont été testées dans ce contexte sont nombreuses et concernent notamment

— Les modèles ARIMA et GARCH [22] : l'idée est de construire un modèle ARIMA dont la variance évolue au cours du temps, ce qui a déjà été étudié dans la partie 3.1.

— Plutôt que de choisir un prédicteur linéaire ou non-linéaire, on peut détecter dans un premier temps la présence d'éventuelles non-linéarités dans les données et ensuite utiliser une méthode de prédiction adaptée [2].

— Les modèles ARMA/ARIMA/FARIMA et les réseaux de neurones [40] : l'idée est de construire un modèle linéaire, e.g., de type ARMA dont l'entrée est construite à partir de la sortie d'un réseau de neurones. On met donc en cascade un réseau de neurones qui capture les non-linéarités du trafic et un modèle linéaire qui capture le lien entre le trafic à l'instant présent et les instants passés. Dans [2], les auteurs ont une approche différente qui consiste à modéliser la série temporelle à l'aide d'un modèle FARIMA et de construire un réseau de neurones dont l'entrée est constituée des résidus entre la série temporelle et les valeurs fournies par le modèle FARIMA.

# Chapitre 4

# Simulations

## 4.1 Données synthétiques

Nous proposons dans cette partie d'étudier les performances des différentes méthodes de prédiction linéaires et non-linéaires pour trois catégories de signaux

i) Signal auto-régressif

ii) Signal auto-régressif avec modulation d'amplitude (i.e., avec tendance périodique), avec modulations rapides et lentes.

iii) Signal non-linéaire obtenu par transformation d'un signal auto-régressif avec trois exemples différents de non-linéarités.

La définition des signaux de test est effectuée dans les paragraphes suivants. La taille des signaux est fixé à $N = 1000$ échantillons pour toutes les expériences numériques considérées dans ce rapport.

### 4.1.1 Signal auto-régressif

Un signal auto-régressif (AR) est obtenu par l'équation de filtrage à réponse impulsionnelle infinie (RII) suivante

$$x_n = \sum_{r=1}^{R} a_r x_{n-r} + \varepsilon_n \tag{4.1}$$

ou $\varepsilon_n$ sont des variables aléatoires gaussiennes centrées **réduites ? ?** indépendantes. Les coefficients AR $a_r,\ r = 1, \ldots, R$ sont fixés arbitrairement comme les coefficients d'un filtre RIF passe-bas de fréquence de coupure normalisée égale $1/2$ avec $R = 20$.

### 4.1.2 Signaux auto-régressifs avec modulations

Nous modélisons le cas de tendances périodiques dans le signal à l'aide d'une modulation sinusoïdale

$$x'_n = x_n\big(1.2 + \sin(2\pi f_1 n)\big) \tag{4.2}$$

ou

$$x'_n = x_n\big(1.2 + \sin(2\pi f_2 n)\big) \tag{4.3}$$

ou $x_n$ est le signal AR défini dans (4.1), et $f_1$ et $f_2$ sont des fréquences de modulation rapide ($f_1$ est choisie telle que 25 périodes sont contenues dans la fenêtre d'observation) et lente ($f_2$ est choisie telle que 5 modulations sont observées dans cette même fenêtre d'observation).

### 4.1.3 Transformations non-linéaires de signaux AR

Enfin, nous considérons trois exemples de signaux obtenues à partir de transformations non-linéaires de signaux AR

$$x'_n = \text{sign}(x_n)x_n^2 \tag{4.4}$$

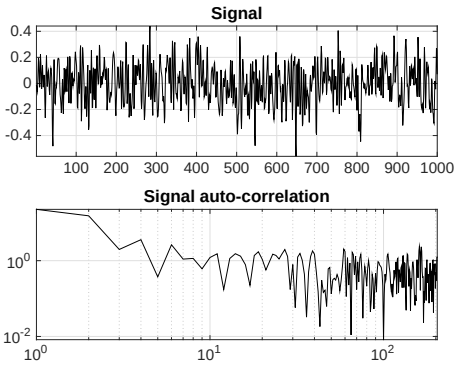
$$x'_n = \text{sign}(x_n)\log(10|x_n| + 1/2) \tag{4.5}$$

$$x'_n = \cos((\pi/2)x_n) \tag{4.6}$$

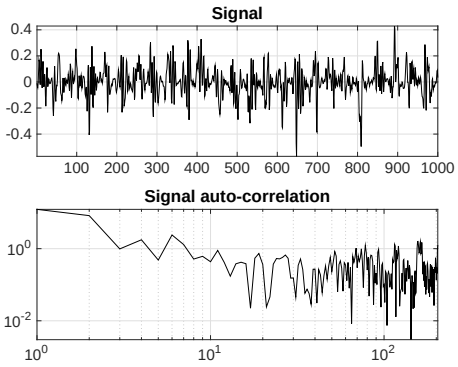
ou $x_n$ est le signal AR défini dans (4.1).

Les 6 signaux de test définis par les équations (4.1-4.6), et leurs fonctions d'auto-corrélation estimées, sont montrées dans la figure 4.1 (a - f).

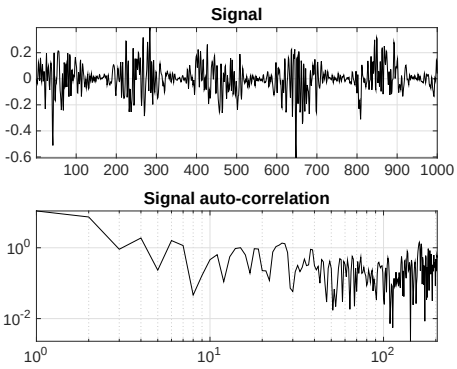
**(a) Signal linéaire - eq.** (4.1)



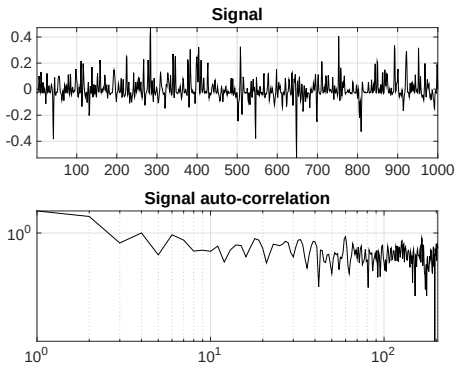

**(b) Signal linéaire avec modulation rapide - eq.** (4.2)



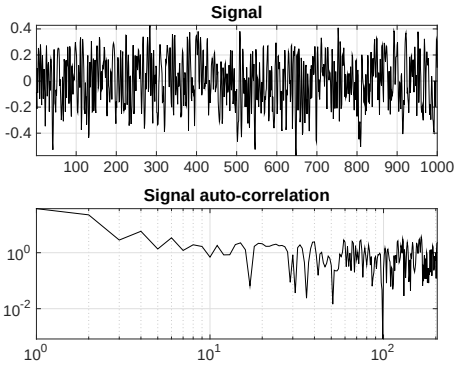

**(c) Signal linéaire avec modulation lente - eq.** (4.3)




**(d) Signal nonlinéaire 1 - eq.** (4.4)




**(e) Signal nonlinéaire 2 - eq.** (4.5)




**(f) Signal nonlinéaire 3 - eq.** (4.6)

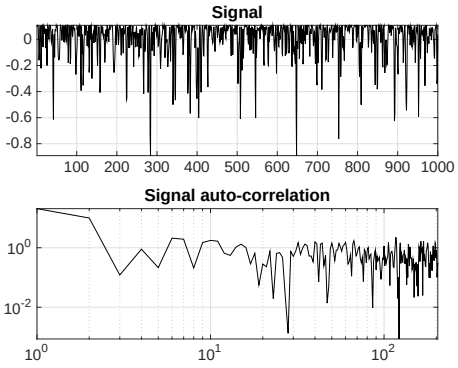



FIGURE 4.1 – Données synthétiques : signaux considérés dans cette étude leurs fonctions d'autocorrélation estimées.

## 4.2 Méthodes de prédiction et évaluation de performances

### 4.2.1 Méthodes de prédiction

Nous illustrons l'utilisation de plusieurs méthodes de prédiction linéaires et non-linéaires pour les données synthétiques définis dans la section précédent. Les résultats pour les méthodes suivantes ne seront pas rapportés ici :

**GARCH.** Il est connu que l'estimation des paramètres d'un modèle GARCH est un problème difficile. Des expériences préliminaires conduites sur des cas d'exemple simples, en utilisant des solutions commerciales (Econometrics Toolbox de MATLAB), nous laissent conclure que l'estimation est peu robuste et fiable. En plus, la méthode GARCH ne vise pas à prédire les données elle-mêmes, mais seulement de fournir de l'information sur l'incertitude (intervalles de confiance). De ces faites, nous pensons que l'utilisation de la méthode GARCH est d'intérêt limité dans le cadre de cette étude.

**ARIMA et FARIMA.** L'étape de prédiction à l'aide des modèles ARIMA et FARIMA est identique à l'application d'un modèle ARMA à des données après pré-traitement (calcul de différences, ou dérivé fractionnaire). Leur illustration sur des données synthétiques stationnaires a donc peu d'intérêt. Les modèles ARIMA et FARIMA seront testés sur des données réelles si ceux-ci présentent des non-stationnarités.

**Méthodes hybrides.** Les méthodes hybrides seront également adaptées aux cas d'usage et testées sur les signaux réels.

### 4.2.2 Paramètres des méthodes de prédiction

Nous précisons ici les paramètres utilisés pour les différentes méthodes de prédiction étudiées dans la suite de ce chapitre.

#### Modèles AR et ARMA

Les ordres des modèles AR et ARMA sont fixés à $p = 20$ et $q = 5$. Nous estimons les coefficients $\boldsymbol{a}$, pour chaque position $n$, à partir de la matrice de covariance empirique $\widehat{\boldsymbol{R}}$ calculé sur une fenêtre glissante de taille 30, et procédons de la même façon pour l'estimation des coefficients $\boldsymbol{b}$.

**Réseaux de neurones**

Le réseaux de neurones utilisé comprend 3 couches cachées, avec 8, 10 et 8 neurones avec $p = 5$ entrées et $q = 1$ sortie. Notons que cette structure de réseau a été déterminée par validation croisée de manière à obtenir les meilleures performances de prédiction. Notons également que le nombre d'entrée du réseau est tel que la prédiction de la valeur $x_n$ est effectuée à partir des $p = 5$ valeurs précédentes, $\boldsymbol{x}_{n-1} = (x_{n-1}, x_{n-2}, ..., x_{n-5})^T$. Enfin, le réseau est appris en utilisant les 50 premiers couples $(\boldsymbol{x}_{n-1}, x_n),\ n = 6, \ldots, 55$, et **expliquer que les dernières valeurs sont prédites sur des fenêtres glissantes et comparées à la vérité terrain**.

**Machines à vecteurs supports**

Comme dans beaucoup d'études sur les machines à vecteurs supports, nous considérons un noyau gaussien, dont la variance a été fixée à $\sigma^2 = 1$ sans perte de généralité. Nous utilisons de nouveau les 50 premiers couples $(\boldsymbol{x}_{n-1}, x_n),\ n = 6, \ldots, 55$ pour l'apprentissage et **la prédiction des points $x_n$ pour $n > 55$ effectuée sur chaque fenêtre glissante**.

**Méthode de référence ("baseline")**

Pour obtenir une idée de la difficulté de la tâche de prédiction, nous calculons également une prédiction "naive", que nous appelons approche "baseline", qui consiste à utiliser l'observation précédente comme prédiction, i.e., telle que

$$\hat{x}_n = x_{n-1}.$$

### 4.2.3 Evaluation de performances

Pour quantifier et comparer la qualité des prédictions, nous proposons d'utiliser les mesures suivantes

— le coefficient de corrélation de Pearson pour le signal $x$ et sa prédiction $\hat{x}$, dénoté $\rho$ . Plus ce coefficient $\rho$ est proche de 1, meilleure est la prédiction

— le rapport signal-distortion (SDR) défini comme

$$\mathrm{SDR} = 10 \log_{10} \Big( ||x||_2^2 \,/\, ||x - \hat{x}||_2^2 \Big) \mathrm{dB}$$

qui sont deux mesures couramment utilisées pour évaluer les performances d'une méthode de prédiction.

## 4.3 Résultats

Les résultats numériques obtenus avec les 4 méthodes de prédiction considérées pour les 6 scénarios de simulation sont résumés dans le tableau suivant, et détaillés dans les 6 figures présentées dans les pages suivantes de ce rapport.

| **Coefficient de corrélation $\rho$** | | | | | | |
|---|---|---|---|---|---|---|
| Méthode | sig. 1 | sig. 2 | sig. 3 | sig. 4 | sig. 5 | sig. 6 |
| "baseline" | 0.68 | 0.67 | 0.67 | 0.62 | 0.60 | 0.49 |
| AR | 0.89 | 0.86 | 0.89 | 0.60 | 0.56 | 0.41 |
| ARMA | 0.94 | 0.90 | 0.94 | 0.62 | 0.59 | 0.39 |
| Réseau de neurones | 0.87 | 0.93 | 0.88 | 0.64 | 0.61 | 0.49 |
| SVM | 0.95 | 0.94 | 0.94 | 0.74 | 0.66 | 0.59 |
| **SDR (dB)** | | | | | | |
| Méthode | sig. 1 | sig. 2 | sig. 3 | sig. 4 | sig. 5 | sig. 6 |
| "baseline" | 1.82 | 1.70 | 1.56 | 1.04 | 1.00 | $-0.24$ |
| AR | 6.11 | 5.23 | 6.12 | 1.84 | 1.59 | 0.79 |
| ARMA | 9.23 | 7.39 | 9.07 | 1.97 | 1.58 | 0.26 |
| Réseau de neurones | 5.99 | 8.30 | 6.50 | 0.91 | 0.56 | 0.94 |
| SVM | 9.56 | 9.48 | 8.93 | 3.45 | 2.06 | 1.01 |

On peut faire les remarques suivantes

— Les performances des méthodes de prédiction linéaire (AR et ARMA) sont très bonnes pour les signaux linéaires (sig. 1 - 3), y compris ceux ayant une tendance périodique (signaux 2 et 3). Par contre, les performances sont moins bonnes pour les signaux non-linéaires ( signaux 4 à 6), ce qui est compréhensible car ces méthodes supposent que chaque point $x_n$ est une combinaison linéaire des points précédents. On peut également noter qu'on obtient de meilleurs résultats avec la méthode ARMA pour les signaux linéaires (ce qui est également compréhensible car le modèle AR est un cas particulier de modèle ARMA) tandis que ces deux méthodes (AR et ARMA) donnent des résultats similaires pour les signaux ayant subi des transformations non-linéaires.

— Le réseau de neurones donne de meilleurs résultats que les méthodes linéaires dans 2 cas seulement sur les 6 cas considérés (en terme de SDR). Le surcoût de complexité et de temps de calcul de cette méthode semble peu justifiable au vu de ces expériences.

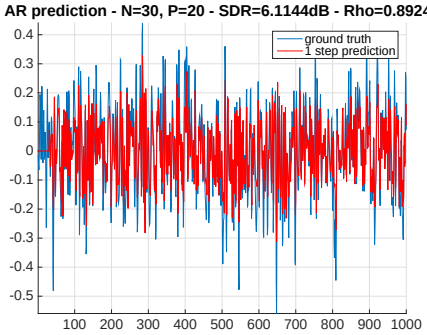
— La méthode SVM donne de meilleurs résultats dans tous les cas, ce qui confirme que cette méthode est une méthode de référence dans le domaine de la prédiction.

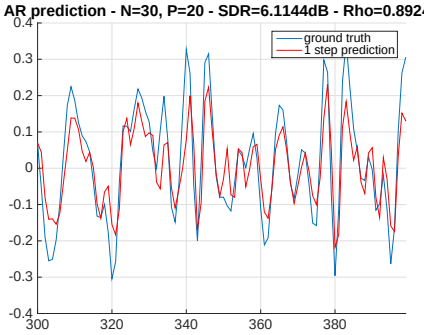
Nous pouvons en conclure que les méthodes de prédiction linéaire conduisent à des résultats très satisfaisants et robustes vis à vis de la nature des signaux traités. Des avantages pratiques de ces méthodes sont leur facilité de mise en œuvre et leur faible coût calculatoire, ce qui permet, par exemple, d'estimer de manière séquentielle (en ligne) les coefficients associés à chaque nouvelle observation. Une autre conclusion importante est que la méthode de prédiction basées sur les machines à vecteurs supports (SVM) donne systématiquement les meilleurs résultats de prédiction, au prix d'un coût de calcul plus élevé.

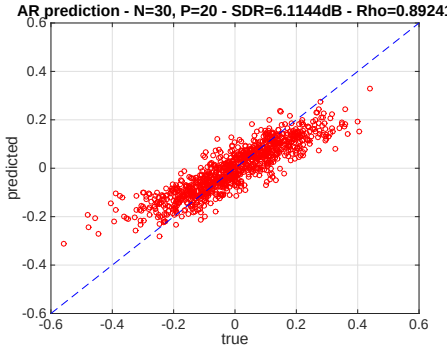
## Signal linéaire - eq. (4.1)

signal et prédiction | signal et prédiction (zoom) | scatter plot

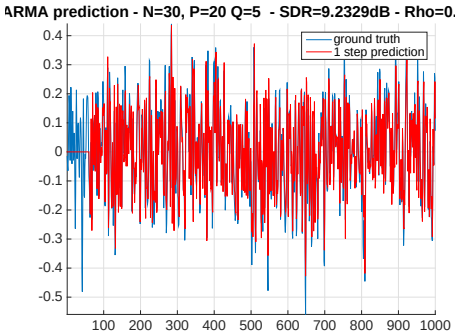
prédiction AR

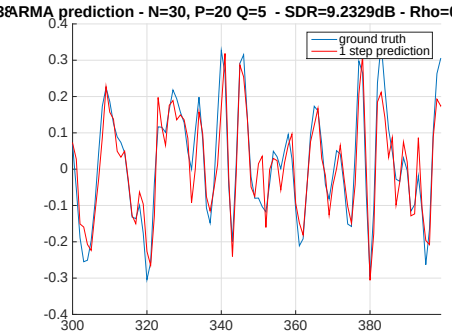


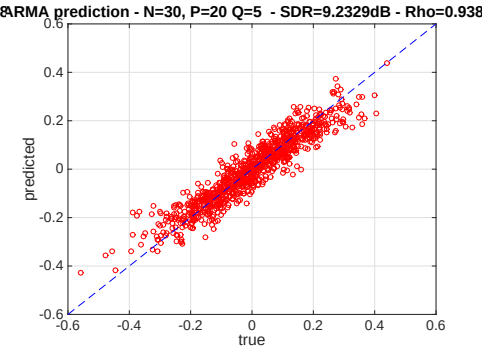



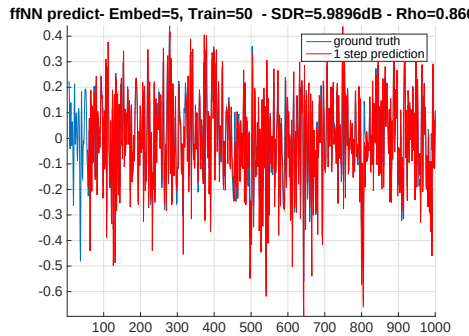

prédiction ARMA

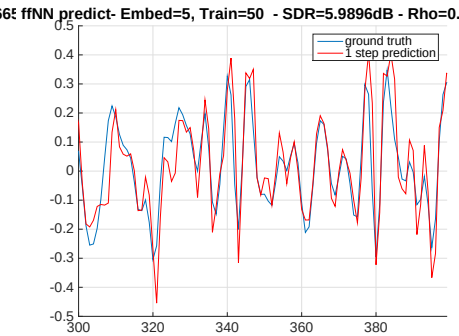


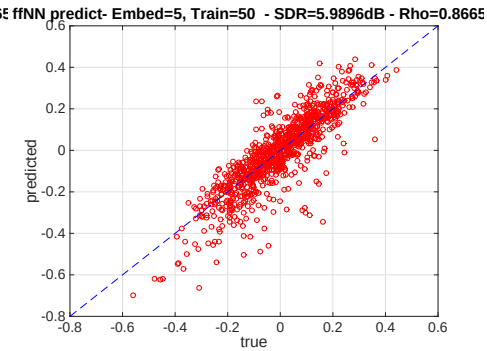



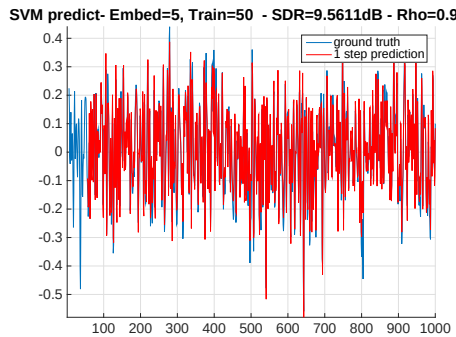

prédiction réseau de neurones

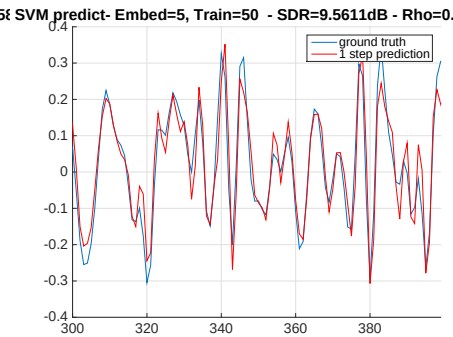


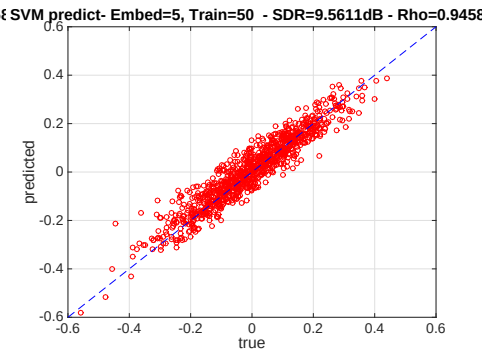




prédiction SVM



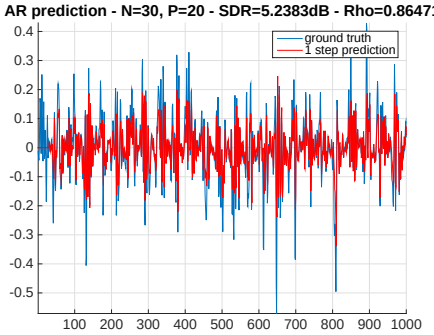



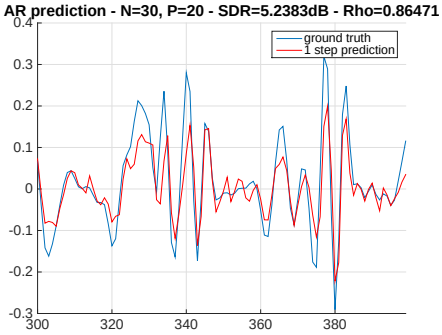



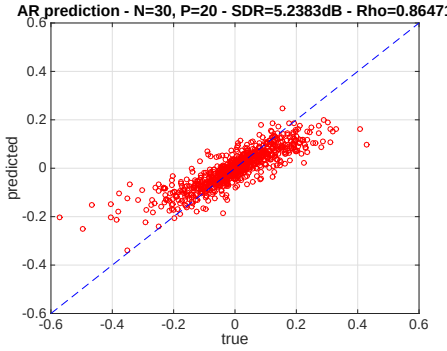
# Signal linéaire, modulation rapide - eq. (4.2)

signal et prédiction | signal et prédiction (zoom) | scatter plot

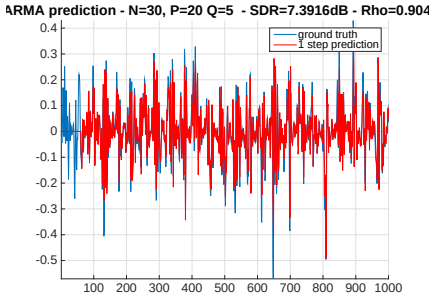
prédiction AR

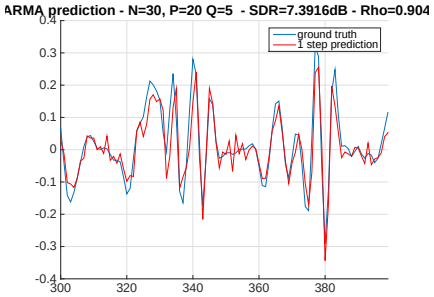


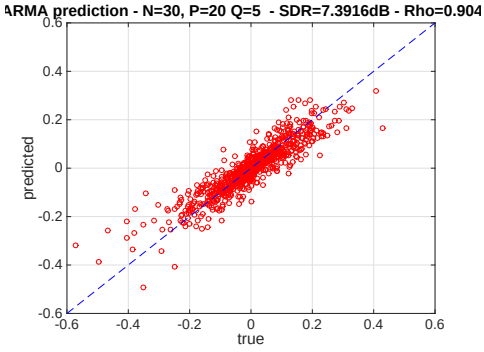



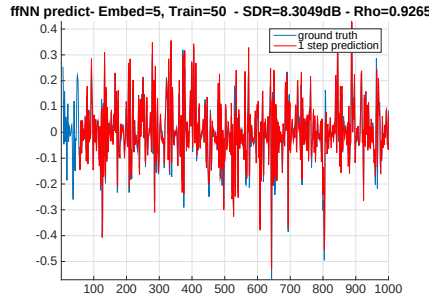

prédiction ARMA

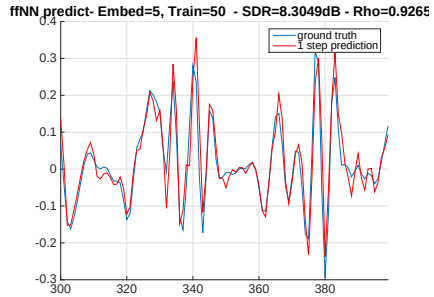


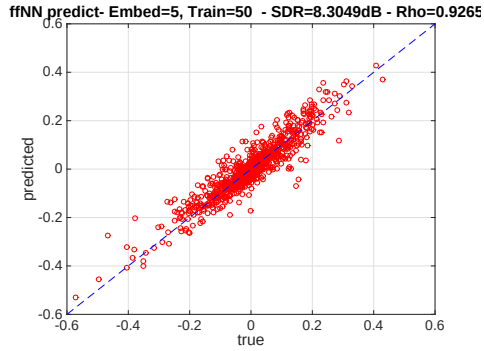



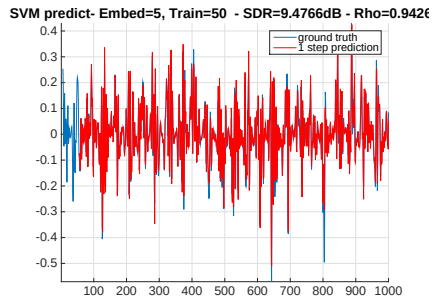

prédiction réseau de neurones

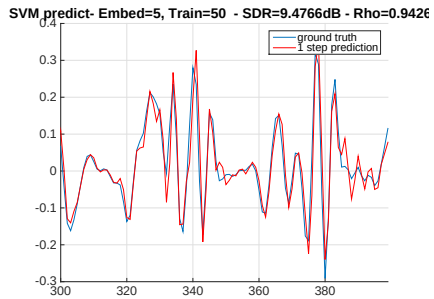


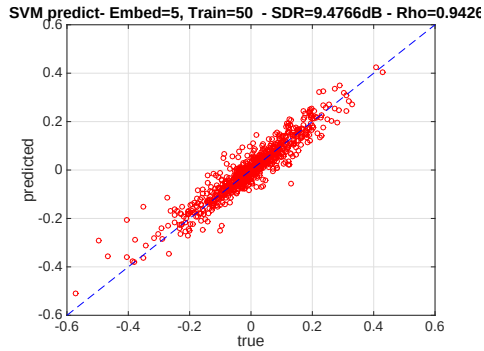




prédiction SVM



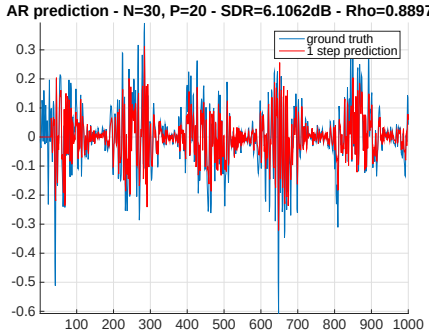



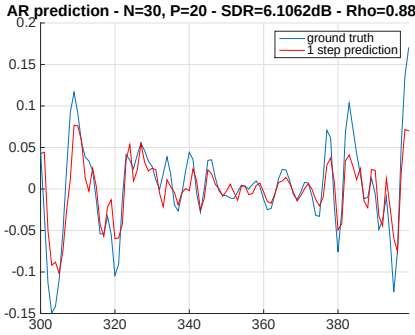



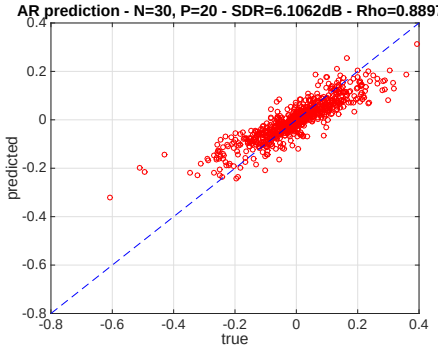
# Signal linéaire, modulation lente - eq. (4.3)

signal et prédiction | signal et prédiction (zoom) | scatter plot

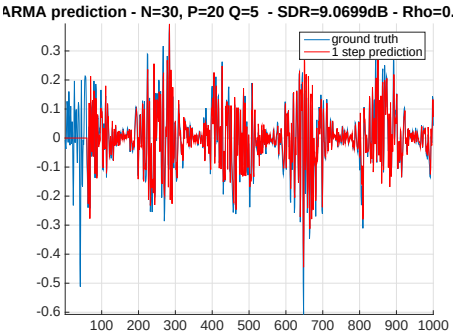
prédiction AR

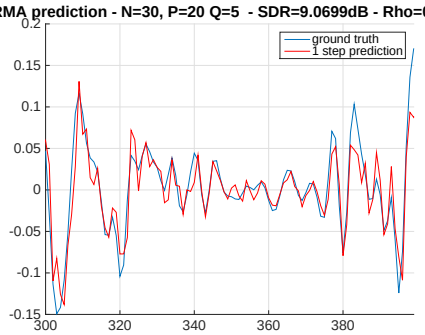


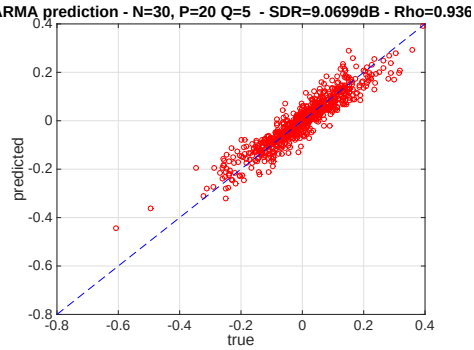



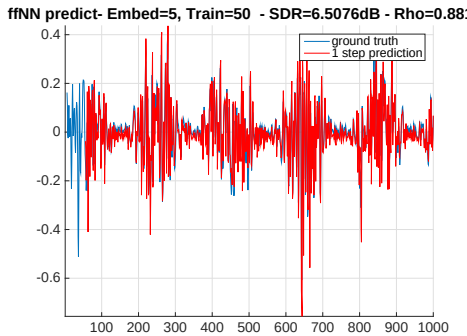

prédiction ARMA

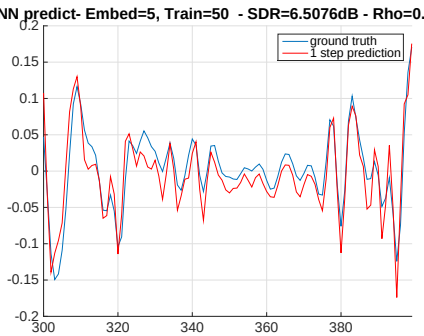


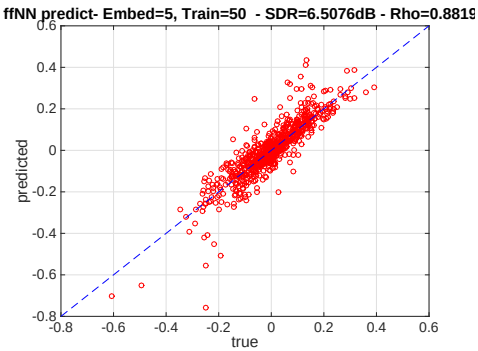



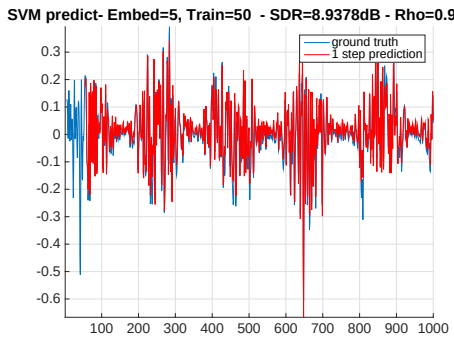

prédiction réseau de neurones

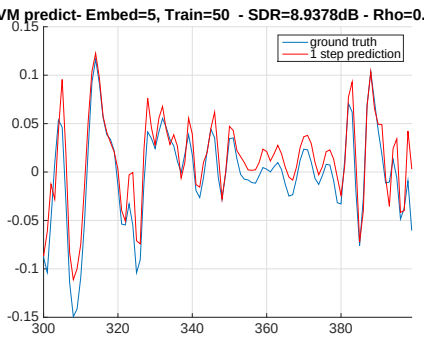


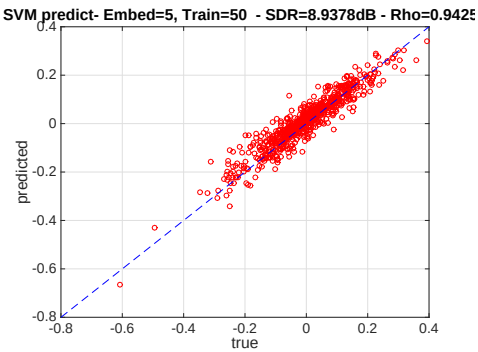




prédiction SVM



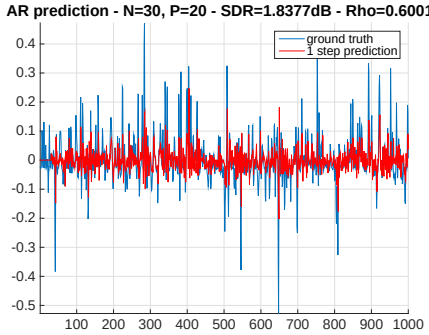



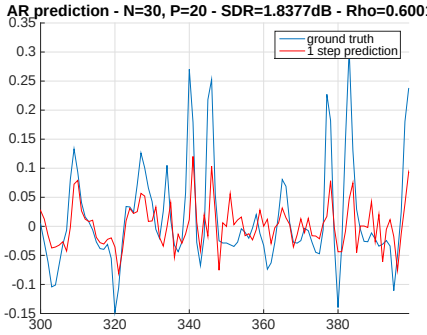



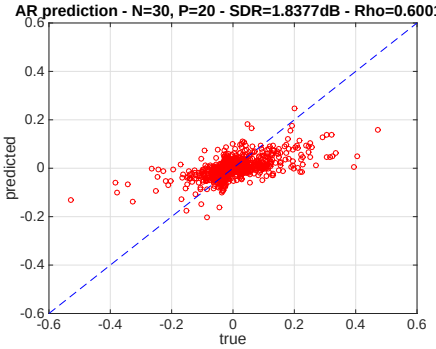
## Signal nonlinéaire 2 - eq. (4.4)

signal et prédiction | signal et prédiction (zoom) | scatter plot

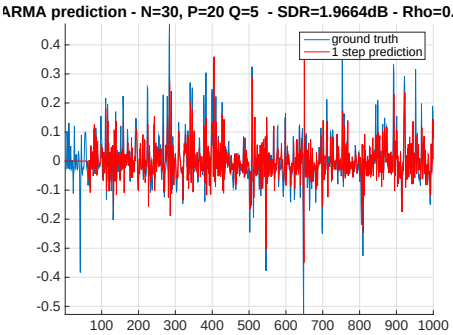
prédiction AR

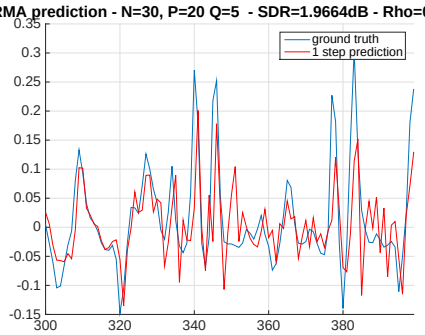


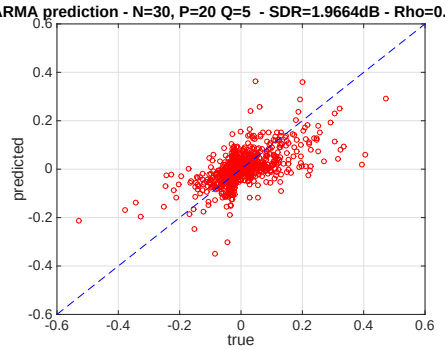



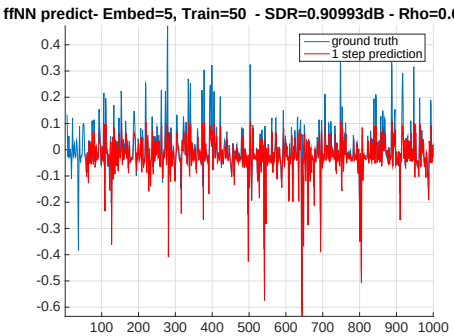

prédiction ARMA

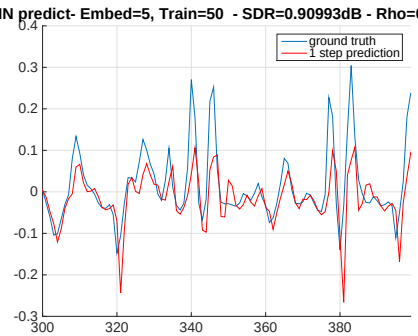


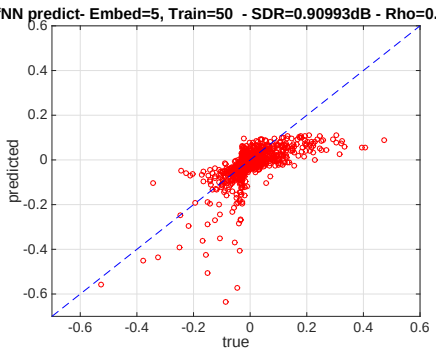



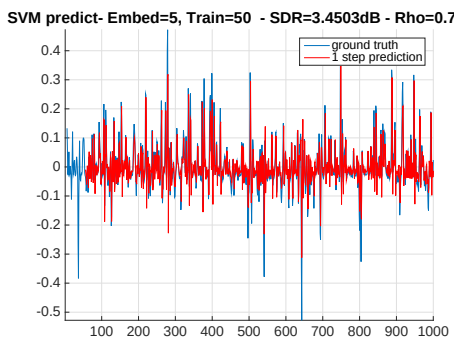

prédiction réseau de neurones

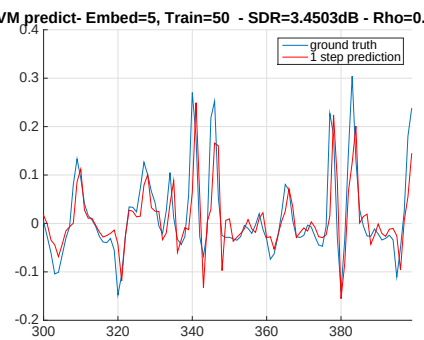


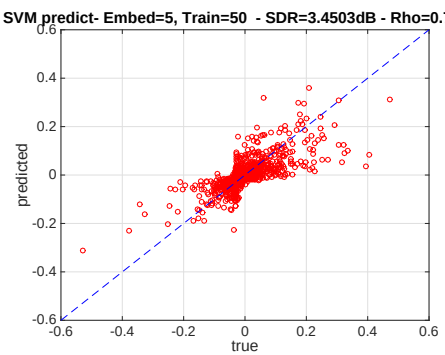




prédiction SVM



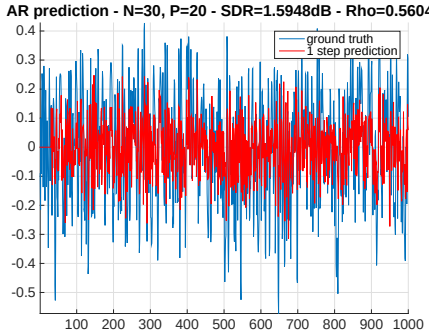



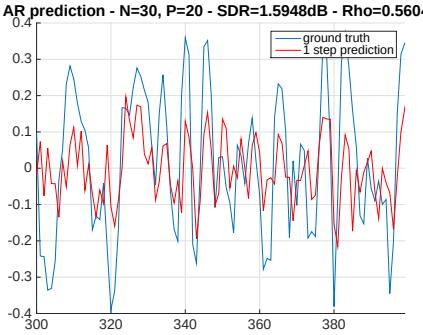



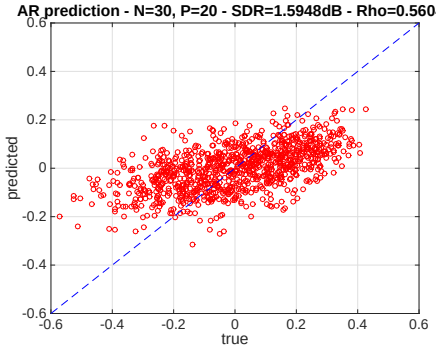
## Signal nonlinéaire 2 - eq. (4.5)

signal et prédiction | signal et prédiction (zoom) | scatter plot

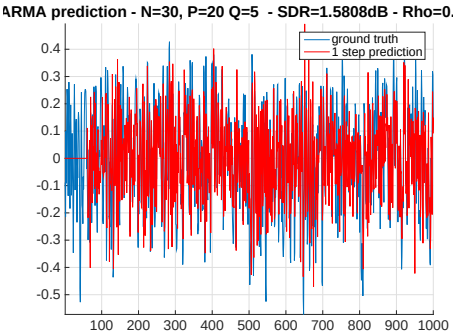
prédiction AR

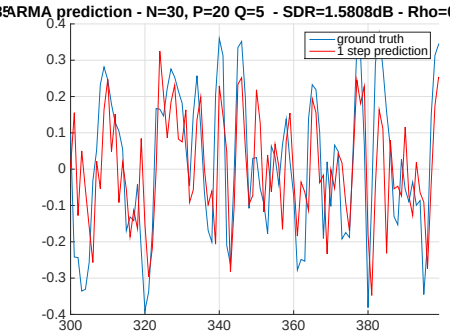


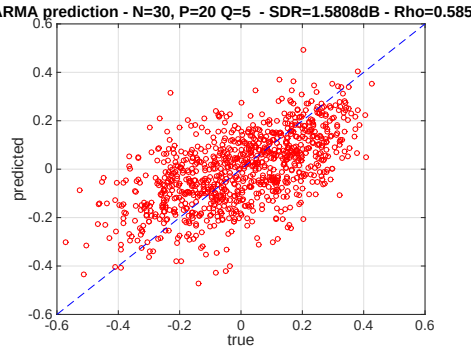



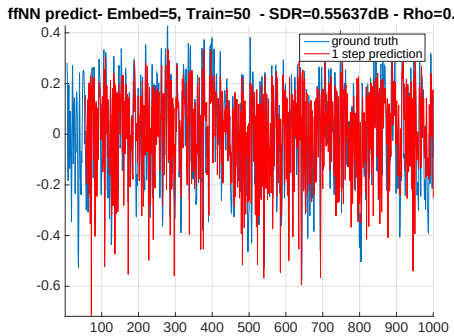

prédiction ARMA

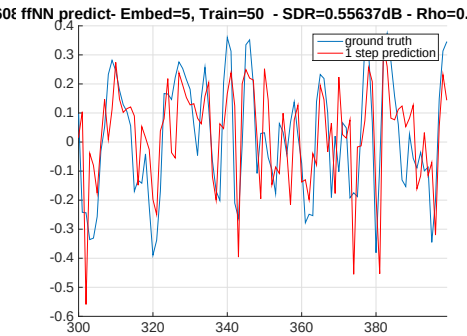


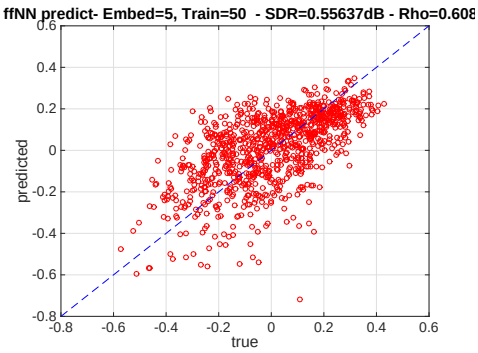



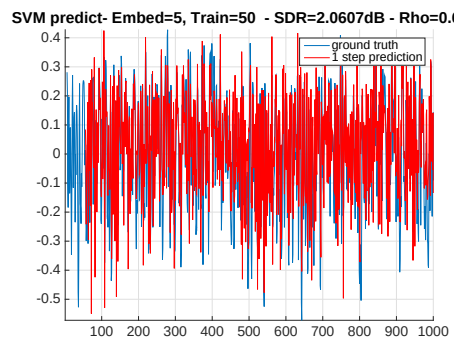

prédiction réseau de neurones

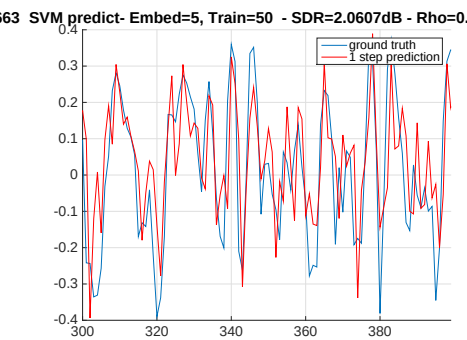


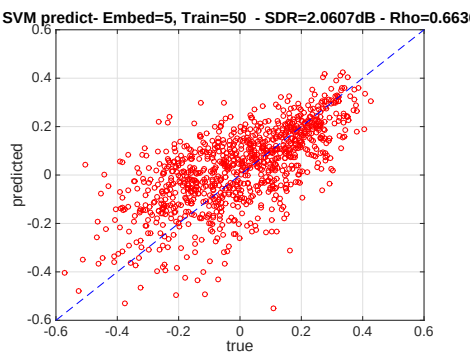




prédiction SVM



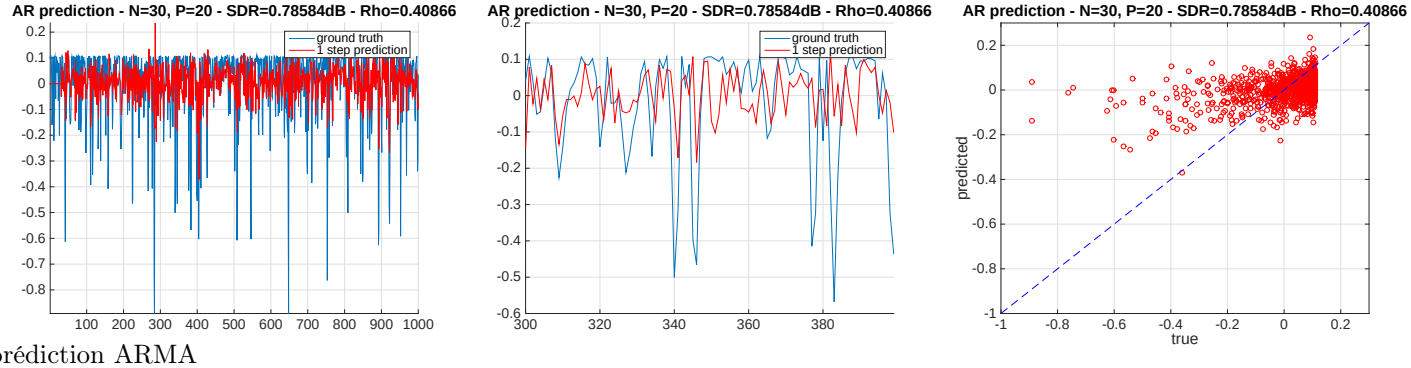



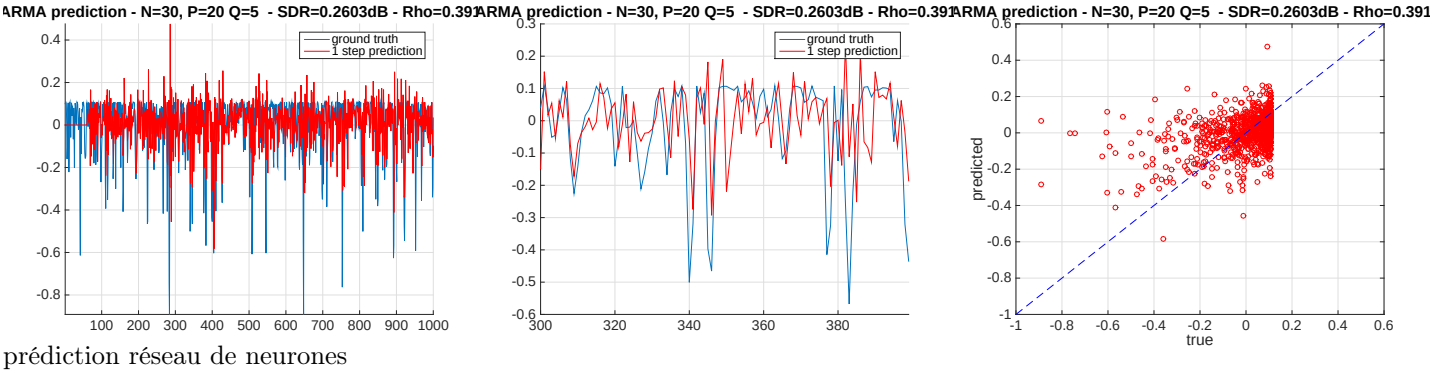
## Signal nonlinéaire 3 - eq. (4.6)

signal et prédiction | signal et prédiction (zoom) | scatter plot

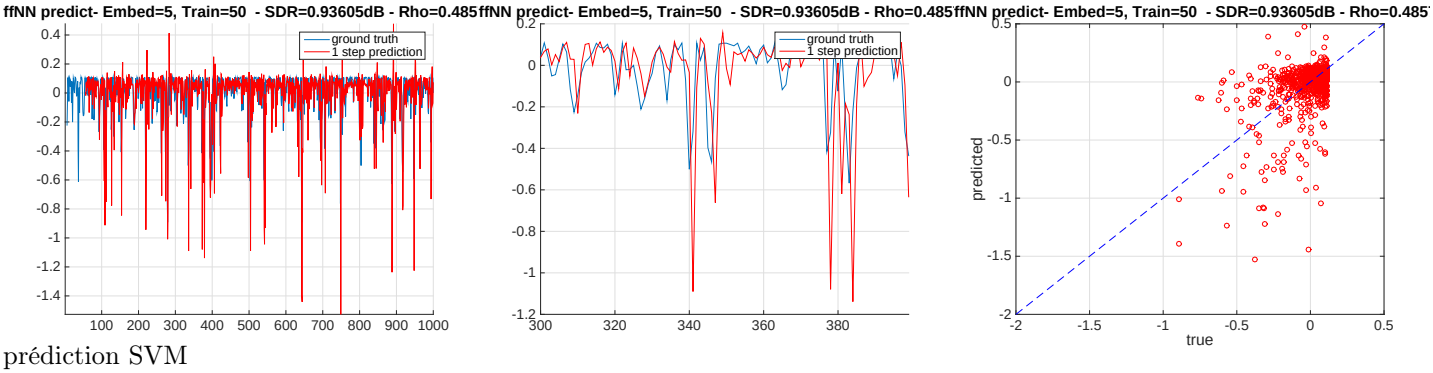
prédiction AR



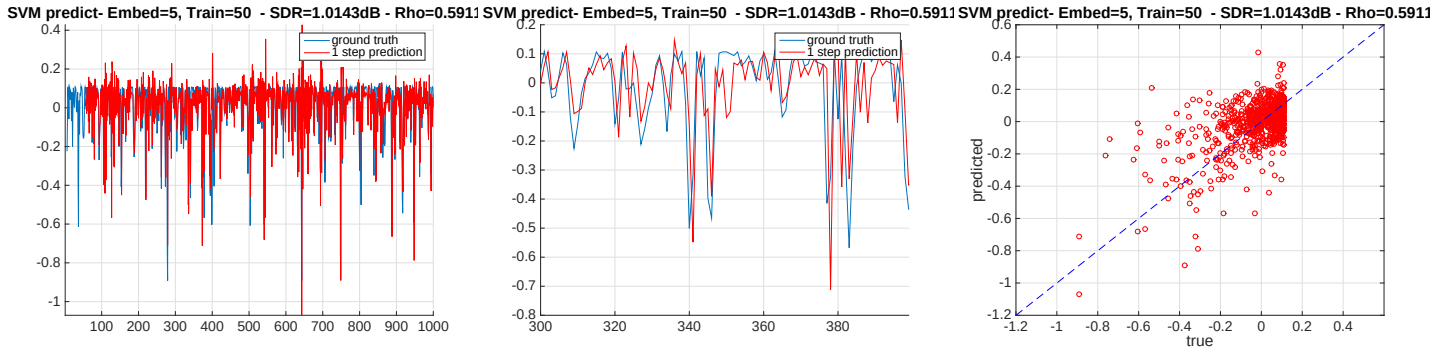

prédiction ARMA




prédiction réseau de neurones




prédiction SVM



# Chapitre 5

# Conclusion

Ce rapport a présenté un état de l'art des techniques de prédiction pour le trafic réseaux. Ces techniques peuvent être reparties dans 3 grandes classes : la famille de méthodes de prédiction linéaires, la famille de méthodes de prédiction non-linéaires, et la famille de méthodes de prédiction hybrides, qui repose sur une combinaison des deux premières. Les méthodes linéaires AR et ARMA, qui ont été utilisées dans de nombreuses applications, semblent intéressantes dans notre contexte. Néanmoins, les méthodes nonlinéaires, utilisant un approche d'apprentissage, donnent en général de meilleurs résultats. En particulier, la méthode de prédiction par machines à vecteurs supports donne de très bons résultats, au prix d'une complexité plus importante. En revanche, les résultats obtenus avec les réseaux de neurones sont peu concluants.

Les méthodes hybrides seront à adapter à des scénarios d'usage particuliers, et testé sur les données réelles.

La méthode non-linéaire GARCH, dont l'utilité dans notre contexte se limite à fournir des intervalles de confiance, s'est avéré délicat à mettre en œuvre, mais elle pourra être envisagé dans le futur si l'application nécessite l'estimation d'intervalles de confiance.

En perspectives, l'ensemble des méthodes, mis à part celles fondées sur les réseaux de neurones, semble adapté à la prédiction de trafic, que cela soit sur la voie aller ou sur la voie retour. Le type de donnée utilisée dépendra du scénario d'étude. Sur la voie retour, la prédiction de la taille des files d'attente permettra d'adapter les allocations de ressources à venir. Sur la voie aller, la prédiction de la charge globale (ou du volume global) sur un spot, permettra d'adapter les allocations des spots ou le routage dans les constellations.

# Bibliographie


[1] M. R. Joshi and T. H. Hadi, "A review of network traffic analysis and prediction techniques," ArXiV, Tech. Rep., July 2015. [Online]. Available : https://arxiv.org/pdf/1507.05722.pdf

[2] C. Katris and S. Daskalaki, "Comparing forecasting approaches for internet traffic," *Expert Systems with Applications*, vol. 42, no. 21, pp. 8172–8183, Nov. 2015.

[3] P. J. Brockwell and R. A. Davis, *Time Series : Theory and Methods.* New-York, NY, USA : Springer-Verlag, 1986.

[4] S. Kay, *Modern Spectral Estimation : Theory and Application.* Englewood Cliffs, NJ, USA : Prentice-Hall, 1988.

[5] M. F. Zhani, H. Elbiase, and F. Kammoun, "Analysis and prediction of real network traffic," *Journal of Networks*, vol. 4, no. 9, pp. 855–865, Nov. 2009.

[6] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, and J. D. Robbins, "Performance models of statistical multiplexing in packet video communications," *IEEE Trans. Commun.*, vol. 36, no. 7, pp. 834–844, July 1988.

[7] R. Grünenfelder, J. P. Cosmas, S. Manthorpe, and A. Odinma-Okafoe, "Characterization of video codecs as autoregressive moving average processes and related queueing system performance," *IEEE J. Sel. areas Commun.*, vol. 9, no. 3, pp. 284–293, April 1991.

[8] G. Rutka, "Some aspects of traffic analysis used for internet traffic prediction," *Electronics and Electrical Engineering Kaunas : Technologija*, vol. 5, no. 93, pp. 7–10, Aug. 2009.

[9] P. K. Hoong, I. K. T. Tan, and C. Y. Keong, "BitTorrent network traffic forecasting with ARMA," *Int. J. Computer Networks and Commun.*, vol. 4, no. 4, July 2012.

[10] Y. Won and S. Ahn, "GOP ARIMA : Modelling the nonstationarity of VBR processes," *Multimedia Systems*, vol. 10, no. 5, pp. 359–378, Aug. 2005.

[11] A. Adas, "Traffic models in broadband networks," *IEEE Commun. Mag.*, vol. 35, no. 1, pp. 82–89, July 1997.

[12] Y. Shu, M. Yu, J. Liu, and O. W. W. Yang, "Wireless traffic modeling and prediction using seasonal ARIMA models," in *Proc. Int. Conf. Commun. (ICC'03)*, Anchorage, AK, USA, May 2003.

[13] C. G. Dethe and D. G. Wakde, "On the prediction of packet process in network traffic using FARIMA time-series model," *Journal of the Indian Institute of Science*, vol. 84, pp. 31–39, 2004.

[14] Y. Shu, Z. Jin, L. Zhang, and O. W. W. Yang, "Traffic prediction using FARIMA models," in *Proc. Int. Conf. Commun. (ICC'99)*, Vancouver, Canada, June 1999.

[15] A. Lagnoux, "Renforcement statistique - séries chronologiques (poly de cours)," Université de Toulouse, Tech. Rep. [Online]. Available : https://perso.math.univ-toulouse.fr/lagnoux/files/2013/12/Poly_renf.pdf

[16] A. Charpentier, "Modèles de prévision - séries temporelles," Université de Rennes, Tech. Rep., 2012. [Online]. Available : http://freakonometrics.free.fr/uqamts.pdf

[17] S. V. Kumar and L. Vanajakshi, "Short-term traffic flow prediction using seasonal ARIMA model with limited input data," *Eur. Transp. Res. Rev.*, vol. 7, no. 1, p. 21, 2015.

[18] J. F. Li, L. Shen, and A. Y. Tong, "Prediction of network flow based on wavelet analysis and ARIMA model," in *Proc. Int. Conf. Wireless Networks and Information Systems*, Shanghai, China, Dec. 2009.

[19] N. C. Anand, C. Scoglio, and B. Natarajan, "GARCH - nonlinear time series model for traffic modeling and prediction," in *Proc. Network Operations and Management Symposium (NOMS'08)*, Salvador, Bahia, Brazil, April 2008.

[20] C. Chen, J. Hu, Q. Meng, and Y. Zhang, "Short-time traffic flow prediction with ARIMA-GARCH model," in *Proc. Int. Vehicle Symp.*, Baden-Baden, Germany, June 2011.

[21] B. Zhou, D. He, Z. Sun, and W. H. Ng, "Network traffic modeling and prediction with ARIMA-GARCH model," in *Proc. Third Int. Working Conf. on performance modelling*

*and evaluation of heterogeneous networks (HET-NET'05)*, Ilkley, West-Yorkshire, U.K., July 2005.

[22] B. Zhou, D. He, and Z. Sun, "Traffic predictability based on ARIMA-GARCH model," in *Proc. Int. Conf. Next Generation Internet Design and Engineering*, Valencia, Spain, April 2006.

[23] R. S. Tsay, *Analysis of Financial Time-series.* New-York, NY, USA : Wiley, 2002.

[24] P. Fryzlewicz, "Lecture notes : Financial time-series, ARCH and GARCH," University of Bristol, UK, Tech. Rep., July 2007. [Online]. Available : http://stats.lse.ac.uk/fryzlewicz/lec_notes/garch.pdf

[25] C. Francq and J.-M. Zakoian, *Structure, Statistical Inference and Financial Applications.* West Sussex, UK : Wiley, 2010.

[26] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. Upper Saddle River, NJ, USA : Pearson Education, 2009.

[27] M. Ovarec, M. Petras, and F. Pilka, "Video traffic prediction using neural networks," *Acta Polytechnica Hungaria*, vol. 5, no. 4, pp. 871–89, June 2008.

[28] P. Cortez, M. Rio, M. Rocha, and P. Sousa, "Internet traffic forecasting using neural networks," in *Proc. Int. Conf. Neural Networks*, Vancouver, BC, Canada, July 2006.

[29] ——, "Multiscale internet traffic forecasting using neural networks and time series methods," *Expert Systems*, vol. 29, no. 2, pp. 143–155, May 2012.

[30] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 3rd ed. NJ, USA : Springer, 1995.

[31] B. Schölkopf, R. Herbrich, and R. Williamson, "A generalized representer theorem," Royal Holloway College, University of London, London, UK, Tech. Rep. NC2-TR-2000-81, Aug. 2000.

[32] C. Richard, J. C. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 1058–1067, March 2009.

[33] T. J. Dodd, V. Kadirkamanathan, and R. F. Harrison, "Function estimation in Hilbert space using sequential projections," in *Proc. IFAC Conf. Intell. Control Syst. Signal Process.*, Faro, Algarve, Portugal, April 2003, pp. 113–118.

[34] Y. Engel, S. Mannor, and R. Meir, "Kernel recursive least squares," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.

[35] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[36] M. Mirza, J. Sommers, P. Barford, and X. Zhu, "A machine learning approach to TCP throughput prediction," *IEEE Trans. Networking*, vol. 18, no. 4, pp. 1026–1039, Aug. 2010.

[37] Y. Zhang and Y. Liu, "Traffic forecasting using least squares support vector machines," *Transportmetrica*, vol. 5, no. 3, pp. 193–213, Sept. 2009.

[38] J. A. K. Suykens, J. Vandewalle, and B. D. Moor, "Optimal control by least squares support vector machines," *Neural Networks*, vol. 14, no. 1, pp. 23–35, Jan. 2001.

[39] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting : An experimental comparison of time-series analysis and supervised learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 871–89, June 2013.

[40] D. Zeng, J. Xu, J. Gu, L. Liu, and G. Xu, "Short term traffic flow prediction using hybrid ARIMA and ANN models," in *Proc. Workshop on Power Electronics and Intelligent Transportation Systems*, Guangzhou, China, Aug. 2008, pp. 621–625.