UNIVERSITÉ DU QUÉBEC

UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

# PRÉVISION DE LA DEMANDE D'ÉLECTRICITÉ PAR RÉGRESSION LINÉAIRE ET RÉSEAUX DE NEURONES ARTIFICIELS : APPLICATION AU RÉSEAU DE LA VILLE DE BAIE-COMEAU

MÉMOIRE PRÉSENTÉ
COMME EXIGENCE PARTIELLE DE LA

MAÎTRISE EN GÉNIE ÉLECTRIQUE

PAR
YACINE AKROUR

DÉCEMBRE 2022

UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

MAÎTRISE EN GÉNIE ÉLECTRIQUE (M. Sc. A.)

**Direction de recherche :**

| | |
|---|---|
| Alben Cardenas Ph. D | Directeur de recherche |
| Karim Belmokhtar Ph. D | Codirecteur de recherche |

**Jury d'évaluation**

| | |
|---|---|
| Afef Ben Abdelghani Bennani Ph. D | Évaluatrice externe |
| Doumbia Mamadou Lamine Ph. D | Évaluateur interne |
| Alben Cardenas Ph. D | Directeur de recherche |

# Résumé

En raison de la croissance démographique et du développement économique, le secteur d'énergie et des réseaux électriques au nord du Canada est dans l'obligation de trouver des solutions et des nouvelles techniques de gestion de la demande d'électricité. En effet, cette région se caractérise par des températures hivernales extrêmes qui mettent déjà une surcharge sur les réseaux électriques. Il est alors très utile d'avoir un outil de prévision de la consommation énergétique qui va pouvoir anticiper les futurs besoins en électricité.

Le premier pas pour prendre de bonnes décisions, consiste à faire une prévision précise de la consommation électrique. Cette prévision permettra en effet de gérer l'achat et la production d'énergie, et ainsi optimiser les coûts de consommation. Elle va aussi permettre la planification de maintenance des réseaux et l'assurance d'un approvisionnement stable en énergie pour les clients.

Cette étude a été faite dans le cadre des réseaux intelligents (*Smart Grid*). Elle vise à faire une prévision de la consommation sur deux horizons différents à court et à moyen terme. Ceci est réalisé dans le but d'aider l'optimisation du réseau de la ville de Baie-Comeau (ville au nord du Canada). Les modèles développés dans cette étude sont basés sur les réseaux de neurones artificiels et sur la régression linéaire, avec l'implémentation d'une base de données réelle fournie par la ville de Baie-Comeau. Avec les résultats prometteurs que nous avons obtenus, notre modèle pourra permettre de stabiliser le réseau électrique et le rendre plus efficace contre les pics de consommation soudains. Par ailleurs, le gestionnaire pourra faire une planification pour réduire la facture et pour empêcher les pannes majeures dans les périodes hivernales.

# Remerciements

Un grand merci à mes deux directeurs de recherche, le professeur Alben Cardenas et le docteur Karim Belmokhtar qui ont dirigé ce travail et qui m'ont énormément aidé et orienté pour réaliser cet humble travail. Je tiens à remercier également, les membres du Jury et les professeurs du département de génie électrique et informatique de l'UQTR, sans oublier toute l'équipe de l'université de Québec à Trois-Rivières pour son aide.

Je tiens à remercier aussi les représentants de la ville de Baie-Comeau de nous avoir fourni les bases de données afin de réaliser ce travail.

Je remercie le Programme d'aide à l'internationalisation de la recherche (PAIR) de l'UQTR et le programme de bourse Universalis Causa de nous avoir aidé financièrement.

Je tiens aussi à remercier ma famille pour leur soutien, et tous ceux qui ont contribué de près ou de loin au bon déroulement de ce mémoire.

# Table des matières

# Liste des tableaux

# Liste des figures

# Liste des symboles

| | |
|---|---|
| AI | Artificial intelligence (Intelligence artificielle) |
| AR | Auto regressive (Auto régressif) |
| ARMA | Auto regressive moving average (Autorégressif et moyenne mobile) |
| ARIMA | Auto regressive integrated moving average (Moyenne mobile intégrée autorégressive) |
| ARX | Auto regressive with eXternal inputs (Autorégressif avec entrées externes) |
| ANN | Artificial neural network (Réseau neuronal artificiel) |
| DT | Decision tree (Arbre de décision) |
| FSVM | Fuzzy support vector machines (Machines à vecteurs de support flous) |
| GRNN | General regression neural network (Réseau de neurones de régression Générale) |
| LTLF | Long term load forecasting (Prévision de charge à long terme) |
| LR | Linear regression (Régression linéaire) |
| LSTM | Long short-term memory (Mémoire longue à court terme) |
| MA | Moving average (Moyenne mobile) |
| MAE | Mean absolute Error (Erreur moyenne absolue) |
| MAPE | Mean absolute Percent Error (Erreur moyenne absolue en pourcentage) |
| ML | Machine learning (Apprentissage machine) |

| | |
|---|---|
| MLP | Multi-layer perceptron (Perceptron multicouche) |
| MSE | Mean square error (Erreur quadratique moyenne) |
| MTLF | Medium term load forecasting (Prévision de charge à moyen terme) |
| MW | MegaWatt (MégaWatt) |
| RMSE | Root mean square error (Racine de l'erreur quadratique moyenne) |
| RNN | Recurrent neural network (Réseau neuronal récurrent) |
| SVM | Support vector machine (Machine vectorielle de support) |
| SVR | Support vector regression (Régression de machine vectorielle de support) |
| STLF | Short term load forecasting (Prévision de charge à court terme) |
| WSVM | Weighted support vector machine (Machine à vecteur de support pondéré) |
| WD | Wavelet decomposition (Décomposition en ondelettes) |

# Chapitre 1 - Introduction

L'énergie est le fondement du développement économique et social dont l'électricité en est l'une des principales sources. La politique énergétique d'une région est donc d'une importance cruciale et vitale, car elle ne guidera pas seulement le développement de la zone, mais affectera également l'environnement de fonctionnement de diverses industries.

La distribution d'électricité dans les régions de froid extrême est souvent très chère et non rentable pour l'économie de la zone considérée. La demande (consommation) électrique est très élevée et l'importation ou la garantie de cette énergie auprès d'autres fournisseurs est une exigence capitale pour combler le manque de ressources. L'optimisation de la production joue alors un rôle très important pour faire face à ce problème. Il est nécessaire de trouver des solutions adéquates ou idoines pour minimiser la consommation afin de stabiliser le réseau électrique.

De nouveaux outils d'informations basés sur la prévision et l'intelligence artificielle permettent de rendre les réseaux électriques intelligents et autonomes. Cette avancée technologique permet d'optimiser les réseaux électriques pour avoir une meilleure efficacité et une fiabilité énergétique. Elle permet aussi de garantir la sûreté d'alimentation des consommateurs, sans pour autant toucher au coût du transport d'électricité et ce qui réduirait les coûts des factures de consommation.

L'intelligence artificielle se base sur l'acquisition et l'analyse d'une grande quantité de données. Cela veut dire accumuler un nombre extrêmement important d'informations et de

données avec différents paramètres. L'analyse de ces derniers nous aide à prendre les meilleures décisions possibles.

Le réseau devient alors efficace et pilotable, la consommation et la production peuvent s'adapter dynamiquement à l'offre et à la demande électrique en fonction des traitements des paramètres collectés.

La collecte de données au cours des années précédentes nous permet de prévoir la consommation électrique en fonction de la demande qui varie selon l'heure de la journée et en fonction des saisons. Une bonne prévision nous permet de connaître notre réseau électrique et l'évolution de la demande. Ainsi, cela nous permettra de fournir la quantité minimale d'électricité éligible à l'importation. En effet, cela nous permettra également de réduire le coût des factures et donc de faire des économies.

## 1.1 Contexte

L'électricité au Canada est fournie grâce à une variété de sources énergétiques (le gaz naturel, les énergies nucléaires, éoliennes et solaires, l'hydroélectrique, le charbon, la biomasse et le pétrole). En fait, en 2019 le Canada a pu produire 632,2 térawattheures (TWh) d'électricité. Cependant, comme illustré dans la Figure 1-1, plus de la moitié de l'électricité produite au Canada provenait de sources hydroélectriques [1].

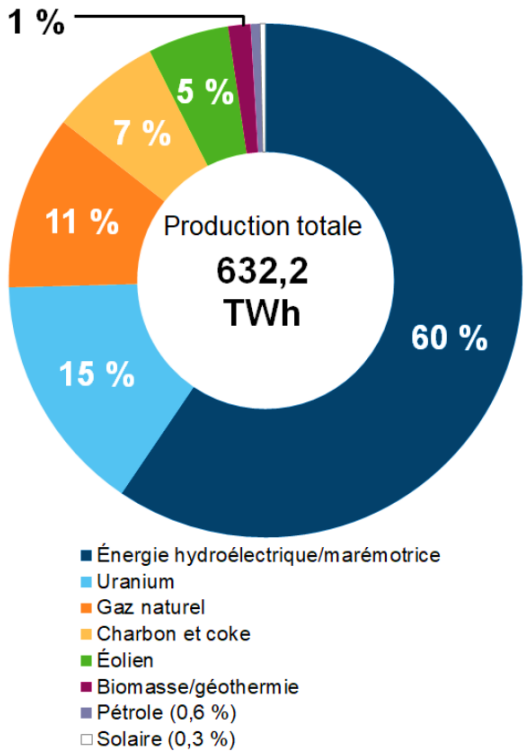



Figure 1-1 Production d'électricité selon le type de combustible (2019) au Canada [1].

Si nous regardons de plus près, chaque province du Canada utilise ses ressources naturelles pour produire son électricité. Ainsi, l'Alberta, la Saskatchewan, la Nouvelle-Écosse et le Nunavut produisent la majeure partie de leur électricité au moyen de combustibles fossiles, comme le charbon, le gaz naturel ou le pétrole. En revanche, le Manitoba, le Québec, la Terre-Neuve-et-Labrador et le Yukon produisent généralement plus de 80 % de leur électricité à partir de l'hydroélectricité[1].

La production et la distribution pour chaque province est règlementée et gérée par des Sociétés privées ou publiques. Plus précisément, au Québec, Hydro-Québec qui est une société d'état québécoise. Elle exerce un quasi-monopole sur la production, le transport et la distribution de l'électricité sur l'ensemble du territoire québécois. La puissance installée s'établit à 37 248 mégawatts (MW) et elle comptait 4,46 millions de clients en 2021 [2].

La presque totalité de l'énergie électrique consommée au Québec provient des centrales hydroélectriques, et cela avec ses 61 centrales hydroélectriques. Ce qui la met le principal producteur d'électricité au Canada et l'un des plus grands producteurs mondiaux d'hydroélectricité, avec une production de 212,9 térawattheures (TWh) d'électricité en (2019) (Figure 1-2) , soit environ le tiers de toute la production d'électricité au Canada [3].

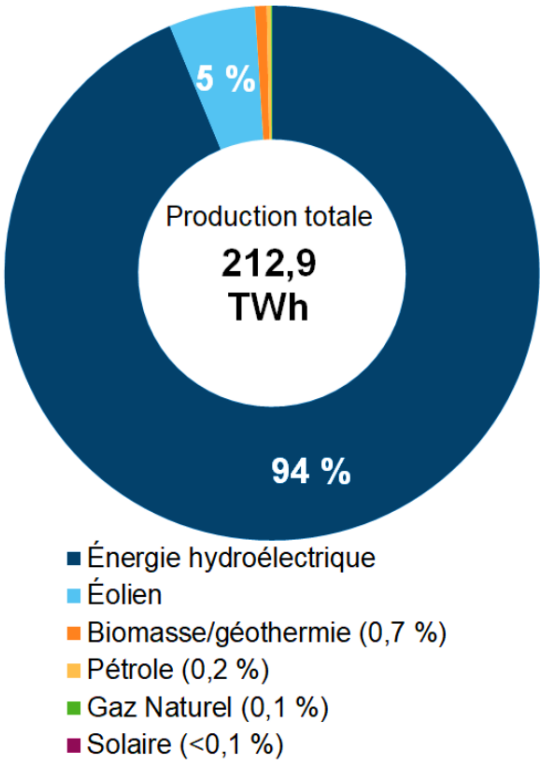



Figure 1-2 Production d'électricité selon le type de combustible (2019) au Québec [3].

La majorité des villes au Québec se situent tout au long du fleuve Saint Laurent, et 80% des habitants vivent dans la zone du sud où réside la majorité des affaires économiques [4].

Notre contexte de recherche est la ville de Baie-Comeau, elle est le chef-lieu de la municipalité régionale du comté de la Manicouagan dans la région administrative de la Côte-Nord du Québec au Canada.

La ville a une vocation principalement industrielle, la structure industrielle est une partie de l'économie de la région, elle est fortement influencée par la présence de sources d'énergie, principalement de sources hydroélectriques, abondantes, peu chères et renouvelables. Elle abrite près de 27 000 habitants [5]. Comme toutes les régions du nord du Canada, cette région se caractérise par un climat d'une température annuelle moyenne avoisinant le un degré Celsius, et des températures extrêmes en dessous de -30°C.

La ville est membre de l'Association des redistributeurs d'électricité du Québec (AREQ) et compte avec près de 5 000 clients dans son réseau électrique, et plus de 110 km de lignes de transport [6].

La ville de Baie-Comeau est responsable de la commercialisation et la distribution d'électricité pour le secteur Marquette, en contrepartie Hydro-Québec est responsable sur le secteur Mingan [6]. L'énergie opérée par la ville est fournie par Hydro-Québec, ce qui représente à la ville une facture annuelle d'achat de plus au moins 17 Millions $ CAD du budget total de 76 M$CAD en 2020 [5], alors qu'en 2014 elle était de 12M$CAD ce qui représente une évolution de 42%. Cette évolution croissante a bien évidemment des causes, c'est ce que nous mène à parler des difficultés et de la problématique de recherche.

## 1.2 Problématique

La ville de Baie-Comeau rencontre des problèmes au niveau de l'opération des réseaux de distribution d'énergie dans les conditions météorologiques extrêmes.

Ces problèmes sont liés à la surconsommation lors des périodes de pic hivernales, causées principalement par le chauffage de l'eau et des espaces d'habitation, et par la consommation de l'électricité des habitations au moment des pics. Sans oublier aussi le secteur industriel (usine production d'aluminium et des pâte à papier et des sites de minage de cryptomonnaie)

[7] qui consomment déjà des dizaines de Mégawatts additionnels, ce qui coûte à la ville un budget important lors de cette période.

Une modélisation dynamique et une prévision de la demande énergétique du réseau de distribution serait d'une grande utilité pour la ville dans la mise en place de stratégies pour la gestion optimale des ressources énergétiques.

## 1.3 Objectif

L'objectif de ce mémoire est de réaliser une modélisation dynamique du réseau de distribution électrique, et d'appliquer cette modélisation au système électrique de la ville de Baie-Comeau.

Les objectifs spécifiques sont :

1. Proposer un modèle dynamique permettant de reproduire et d'anticiper la demande énergétique d'un réseau de distribution;
2. Intégrer au modèle proposé les particularités du réseau opéré par la ville de Baie-Comeau pour tenir compte de la problématique de l'opération des réseaux de distribution dans les conditions météorologiques de froid extrême;
3. Implémenter et valider le modèle proposé avec des données spécifiques du réseau opéré par la ville de Baie-Comeau.

## 1.4 Méthodologie

Nous avons adopté la méthodologie suivante :

1. Analyse (étude) des données techniques du réseau opéré par la ville de Baie-Comeau (tension, puissance, facteur de puissance, facteur d'utilisation) et des facteurs météo (température, ensoleillement, facteur éolien) particuliers à la région;
2. Revue de la littérature sur la modélisation dynamique des réseaux électriques. Un accent particulier sera accordé à la modélisation et la prévision basée sur les données et sur l'intelligence artificielle [8] [9] ;
3. Proposition d'un modèle dynamique permettant de reproduire et d'anticiper la demande énergétique d'un réseau de distribution à partir des caractéristiques techniques, des profils historiques, des facteurs météo et calendriers[10] ;
4. Implémentation sous MATLAB et/ou Python et validation du modèle en utilisant des données réelles fournies par la ville de Baie-Comeau.

Le but principal du modèle issu de ce travail sera de permettre la validation de stratégies pour optimiser l'utilisation des ressources disponibles, et améliorer le facteur d'utilisation dans les moments de pointe et ainsi réduire le coût de la facture.

## 1.5 Organisation du mémoire

Le mémoire est organisé en cinq chapitres et répartis comme suit :

Le deuxième chapitre est une étude bibliographique qui présente des modèles de prévision de séries temporelles persistantes, et aussi les différents horizons de prévision.

Le chapitre 3 concerne les théories liées au système de prévision proposé, à savoir les réseaux de neurones et la régression linéaire, ainsi que les notions nécessaires pour la réalisation de notre étude, tels que la prédictibilité des données et l'évaluation des modèles.

Le chapitre 4 parlera sur la description des bases de données réelles de la ville de Baie-Comeau, ensuite une section sera consacrée à l'étude de corrélation et la prédictibilité de données, et deux autres sections pour la modélisation et l'étude de la prévision de la demande en puissance à moyen et à court terme.

Enfin, le dernier chapitre nous permettra de tirer des conclusions générales, de résumer les résultats et d'exposer des perspectives.

# Chapitre 2 - État de l'art

## 2.1 Introduction

Dans la littérature, il existe plusieurs méthodes et algorithmes de prévision de la demande en puissance. Ces méthodes se surpassent les unes des autres et donnent différents résultats selon le domaine d'application et les paramètres externes. Ces dernières années, les méthodes les plus populaires sont celles liées à l'intelligence artificielle et celle-ci grâce à l'évolution des processeurs d'ordinateurs avec leur surprenante capacité de calcul. Ces algorithmes sont plus flexibles et peuvent faire face à des degrés de complexité plus élevés. L'une des techniques la plus fondamentale utilisée dans de nombreux domaines, est la prévision de séries chronologiques. Dans ce chapitre, nous présenterons une étude bibliographique dans laquelle nous examinerons les différents modèles et algorithmes les plus utilisés dans le domaine des prévisions.

## 2.2 Méthode et données de prévision

Les méthodes de prévision appropriées dépendent largement des données disponibles pour l'analyse. On distingue deux types de procédé : qualitatif et quantitatif.

### *2.2.1 Prévisions qualitatives*

S'il n'y a pas de données historiques disponibles, ou si les données disponibles ne sont pas pertinentes et fiables pour les prévisions, alors cette méthode est tout de même nécessaire. Ces méthodes ne sont pas purement de la prévision au hasard, il existe des approches

structurées et bien développées pour obtenir de bonnes prévisions sans utiliser de données historiques. Les deux approches les plus répondues dans ce cas sont la méthode *Delphi* [11] et la prévision par analogie [12].

### *2.2.2 Prévisions quantitatives*

Ces méthodes peuvent être appliquées dans le cas où on a des informations chiffrées disponibles sur le passé en premier lieu, et deuxièmement s'il est raisonnable de supposer que certains aspects des schémas passés se poursuivront dans le futur.

Il existe un large éventail de méthodes de prévisions quantitatives, souvent développées dans des disciplines différentes à des fins spécifiques. Chaque méthode a ses propres propriétés, les précisions et les coûts qui doivent être pris en compte lors du choix d'une méthode spécifique.

La plupart des problèmes de prévision quantitative utilisent soit des données de séries chronologiques, qui sont collectées à intervalles réguliers dans le temps, soit des données transversales celles-ci sont collectées à un moment donné. Notre étude se basera donc sur les méthodes de prévision quantitative parce que nous avons en possession une base de données numérique réelle prête à être utilisée pour faire la prévision et l'analyse.

## 2.3 Prévision des séries chronologiques

Tout ce qui est observé séquentiellement dans le temps est une série chronologique, souvent aussi appelée série temporelle. Généralement elle est observée à des intervalles de temps régulier, par exemple par heure, jour, semaine, mois ou année.

Lors de la prévision de données de séries chronologiques, l'objectif est d'estimer comment la séquence d'observations se poursuivra dans le futur.

Les méthodes de prévision de séries chronologiques les plus simples utilisent uniquement des informations sur la variable à prévoir et ne tentent pas de découvrir les facteurs qui affectent son comportement. Autrement dit, ils extrapoleront les tendances et les modèles saisonniers, mais ils ignorent toutes les autres informations externes, ceci est le modèle le plus simple appelé Auto Régressifs (AR) ou univariées dans certaines sources [13].

Le deuxième type de modèle est multivarié, il utilise l'ensemble des variables explicatives, appelé en anglais « *Autoregressive With eXternal inputs* (ARX) » [13]. Dans la littérature on trouve que les chercheurs utilisent des méthodes basées sur ces deux concepts cités ci-dessus. On commence par citer et définir les méthodes les plus référées.

### *2.3.1 Auto-Régression*

Dans un modèle d'auto-régression la prévision de la variable d'intérêt est obtenue avec l'utilisation d'une combinaison linéaire des valeurs passées de la variable. Le terme auto-régression désigne qu'il s'agit d'une régression de la variable sur elle-même. Le modèle auto régressif d'un ordre *p* peut s'écrire comme suit [14] :

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t \qquad (2\text{-}1)$$

Où $\varepsilon_t$ est un signal de bruit blanc, avec des valeurs décalées de $y_t$ comme prédicteurs, et $\phi_p$ sont les paramètres de régression. *c* représente la moyenne des changements entre des observations consécutives. Le changement de ces paramètres va donner des prévisions de série chronologique. Les modèles AR peuvent être appliqués à une variété d'échantillons de série temporelle.

### *2.3.2 ARIMA*

ARIMA ou en anglais *AutoRegressive Integrated Moving Average* qui veut dire Moyenne mobile intégrée autorégressive. Ce modèle est donné comme ARIMA (p, d, q) et peut être écrit comme dans la relation suivante [14] :

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t \qquad (2\text{-}2)$$

Où $y'_t$ représente la série différenciée, p est l'ordre de la portion autorégressive, *d* est le degré de différenciation première impliqué, $q$ est l'ordre de la partie moyenne mobile, et *c* est la moyenne des changements entre des observations consécutives.

Les mêmes conditions de stationnarité et d'inversibilité qui sont utilisées pour les modèles autorégressifs s'appliquent également à un modèle ARIMA.

Les ordres (p, d, q) de ARIMA divergent sur un modèle dissemblable comme dans le tableau suivant [14] :

Tableau 2-1 Cas particuliers des modèles ARIMA.

| | |
|---|---|
| Bruit blanc | ARIMA (0,0,0) |
| Marche au Hazard | ARIMA (0,1,0) |
| Auto Régressive (AR) | ARIMA (p,0,0) |
| Moyenne mobile (MA) | ARIMA (0,0,q) |

### *2.3.3 Lissage exponentiel*

Le lissage exponentiel a été proposé à la fin des années 1950 par Winter[15] et Holt [16]. C'est une méthode de prévision de séries chronologiques pour des données univariées qui peut être étendue pour prendre en charge des données avec une tendance systématique ou une composante saisonnière. C'est une méthode de prévision pratique qui peut être utilisée comme alternative des méthodes de ARIMA. Cette méthode implique l'utilisation d'une équation de prévision et deux équations de lissage (une pour le niveau et l'autre une pour la tendance) [16], [14] :

Équation de prévision : $\hat{y}_{t+h|t} = \ell_t + hb_t$ (2-3)

Équation de niveau : $\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$ (2-4)

Équation de tendance : $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$ (2-5)

Où $\ell_t$ est l'estimation du degré de la série à l'instant *t*, $b_t$ l'estimation de la tendance (la pente) de la série au moment *t,* $\alpha$ est le paramètre de lissage. La prévision anticipée *h* est égale au dernier niveau estimé plus *h* fois la dernière valeur de tendance estimée. Et enfin $\beta^*$ est le paramètre de lissage de la tendance $0 \leq \beta^* \leq 1$ .

### *2.3.4 Estimation des moindres carrés*

Le principe des moindres carrés permet de choisir efficacement les coefficients en minimisant la somme des carrés des erreurs. Alors, il faut bien choisir les coefficients $\beta_0, \beta_1, \dots, \beta_k$ pour avoir le minimum d'erreur possible.

$$\sum_{t=1}^{T} \varepsilon_t^2 = \sum_{t=1}^{T} \left(y_t - \beta_0 - \beta_1 x_{1,t} - \beta_2 x_{2,t} - \cdots - \beta_k x_{k,t}\right)^2 \quad (2\text{-}6)$$

Elle est nommée l'estimation des moindres carrés car elle donne la plus petite valeur pour la somme des erreurs au carré. Trouver les meilleures estimations des coefficients est appelé ‘ajustement’ [17].

## 2.4 Apprentissage machine pour la prévision en série chronologique

L'apprentissage machine (ML) est une technique d'intelligence informatique qui utilise des algorithmes programmés pour analyser les données d'entrée et en tirer des enseignements via un processus supervisé ou non supervisé pour prédire les valeurs de sortie [18].

### *2.4.1 Apprentissage machine supervisé*

L'apprentissage supervisé est une approche d'apprentissage automatique définie par son utilisation d'ensemble de données à classifier. Ces ensembles de données sont conçus pour entraîner ou « superviser » des algorithmes afin de classer les données ou de prédire les résultats avec précision. À l'aide d'entrées et de sorties étiquetées, le modèle peut mesurer sa précision et apprendre au fil du temps.

L'apprentissage supervisé peut être séparé en deux types de catégorie: la classification et la régression.

#### 2.4.1.1 Classification

La classification est le processus de prévision des étiquettes en fonction des catégories comme représentée dans la Figure 2-1.

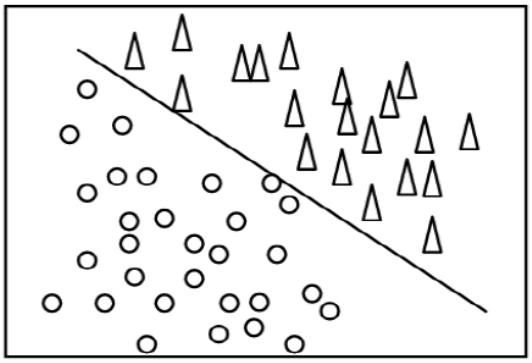


Figure 2-1 Classification.

Pour le processus de classification, le classificateur est construit et il est utilisé pour une classification ultérieure des données. Construire un classificateur est un processus d'apprentissage, il peut être construit à l'aide d'algorithmes de classification à partir de l'ensemble de données d'apprentissage. Chaque type consiste en un ensemble de données d'entraînement avec des étiquettes. Les données de test sont ensuite utilisées pour évaluer les règles de classification à l'aide du classificateur. Les attributs d'entraînement peuvent avoir des valeurs nominales et numériques, tandis que les attributs cibles peuvent avoir plus de deux résultats possibles [19].

Les classificateurs linéaires, les machines à vecteurs de support [19], les arbres de décision et les forêts aléatoires [20] sont tous des types courants d'algorithmes de classification.

### 2.4.1.2 Régression

La régression est une méthode d'analyse statistique pour identifier la relation entre les variables. La relation peut être identifiée entre les variables dépendantes et indépendantes. Elle peut être décrite à l'aide de la fonction de distribution de probabilité représentée dans l'équation 2-7, et présentée dans la Figure 2-2. Cette figure montre la droite de régression la

mieux ajustée sur les données. Elle peut servir de modèle prédictif pour de nouveaux événements basés sur ceux déjà observés [19].

$$Y = f(X, \beta) \qquad (2\text{-}7)$$

Où Y est une variable dépendante, X est une variable indépendante et β est un paramètre inconnu.

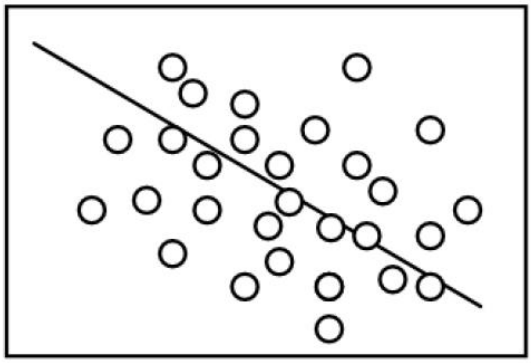


Figure 2-2 Régression [19].

Certains algorithmes de régression populaires sont : la régression linéaire, la régression logistique et la régression polynomiale [21].

### 2.4.1.3 Réseaux de neurones artificiels

Les réseaux de neurones artificiels (ANN pour *Artificial Neural Network*) sont des méthodes prédictives basées sur des modèles mathématiques qui tentent d'imiter le cerveau. Ils permettent des relations non linéaires complexes entre la variable de réponse et ses prédicteurs. Les ANN ont été proposés au début des années 1960, mais ils ont reçu peu d'attention jusqu'au milieu des années 80. La véritable percée dans la recherche sur les ANN est venue avec la découverte de la méthode de rétropropagation, en raison des ordinateurs

personnels rapides et peu coûteux [22]. Plusieurs algorithmes permettent la prévision avec les ANN, on cite les plus apparents dans la littérature :

Modèle réseau neuronal de régression généralisée (GRNN) [23], le modèle autorégressif de réseau de neurones (NNANR) [23] et le modèle de perceptron multicouche (MLP).

#### 2.4.1.4 Machine à Vecteurs de Support (SVM)

Développé par Vapnik et ses collègues en 1995 [24], le modèle de Machine à Vecteurs de Support (SVM) a été introduit pour les problèmes de reconnaissance de formes et de régression. Ces algorithmes SVM ont une base théorique solide ancrée dans la théorie de l'apprentissage statistique et ont été appliqués dans de nombreux domaines, y compris dans la prévision de la demande d'énergie. Des exemples des algorithmes utilisés par les chercheurs basés sur cette méthode sont : la machine à vecteurs de support flous (FSVM) [25], et la machine à vecteurs de support pondérés (WSVM) [26].

#### 2.4.1.5 Ondelette

La transformation en ondelettes a été largement appliquée à de nombreux domaines, y compris dans les systèmes électriques. Elle n'utilise pas de relation temps-fréquence, mais plutôt une relation d'échelle de temps ce qui est différent avec l'analyse de fourrier [27]. La transformée en ondelettes permet d'utiliser de longs intervalles de temps où nous pouvons voir des informations plus précises sur les basses fréquences, et des régions plus courtes où nous voulons des informations sur les hautes fréquences.

L'ondelette est définie comme la somme sur tout le temps du signal multipliée par une mise à l'échelle. La version décalée de la fonction d'ondelettes $\psi(t)$ appelé *Mother wavelet* satisfait les conditions suivantes [27] :

$$C_\psi = \int_{-\infty}^{+\infty} |\hat{\psi}(w)|^2 \frac{dw}{|w|} < \infty \quad (2\text{-}8)$$

Pour un signal $f(t) \in R$ la transformée d'ondelettes est définie comme la relation

$$Wf(a,b) = \langle f, \psi_{a,b} \rangle = |a|^{-1/2} \int_{-\infty}^{+\infty} f(t)\psi\left(\frac{t-b}{a}\right) dt \quad (2\text{-}9)$$

Où $a, b \in R, a \neq 0$ .La Transformée d'ondelettes est réversible, f*(t)* est définie comme

$$f(t) = \frac{1}{c_\psi} \iint_{R^2} Wf(a,b)\psi_{a,b}(t) \frac{da}{a^2} db \quad (2\text{-}10)$$

Des exemples d'études faites avec des méthode plus approfondies basées sur cette approche sont: *Lifting Wavelet Transform* (LWT) [28], et décomposition en ondelettes (WD) [29].

### *2.4.2 Apprentissage machine non supervisé*

L'apprentissage non supervisé adopte des algorithmes d'apprentissage automatique pour regrouper et analyser des ensembles de données non étiquetées. Ces algorithmes découvrent des modèles cachés dans les données sans intervention humaine. Les modèles d'apprentissage non supervisés sont utilisés pour trois tâches principales : le regroupement, l'association et la réduction de la dimensionnalité [30].

#### 2.4.2.1 Regroupement

Le regroupement est utilisé pour « regrouper » des données non étiquetées en fonction de leurs ressemblances ou de leurs différences. Par exemple, les algorithmes de *clustering K-*

*means* attribuent des points de données similaires en groupes, où la valeur *K* représente la taille du groupement et la granularité.

#### 2.4.2.2 Association

L'association est un autre type de méthode d'apprentissage non supervisé qui utilise différentes règles pour trouver des relations entre variables dans un ensemble de données.

#### 2.4.2.3 Réduction de la dimensionnalité

La réduction de la dimensionnalité est une technique d'apprentissage utilisée lorsque le nombre d'entités (ou de dimensions) dans un jeu de données donné est trop élevé. Elle restreint le nombre d'entrées de données à une taille gérable tout en prémunissant l'intégrité des données. Souvent, cette technique est utilisée dans l'étape de prétraitement des données.

### 2.5 Horizon de prévision

La prévision de la demande en puissance se base sur la consommation d'énergie précédente. Autrement dit avoir l'historique de consommation d'une région géographique donnée pour projeter l'énergie requise sur une base horaire, quotidienne, hebdomadaire, mensuelle ou annuelle qui sera nécessaire pour combler les besoins à l'avenir.

Sur la base de l'horizon de prévision, il existe trois classes distinctes de prévision de charge qui peuvent être nommées : prévision de charge à court terme (STLF) qui couvre un horizon d'une heure à une semaine, ensuite à moyen terme (MTLF) qui couvre un horizon d'une semaine à une année, et enfin à long terme (LTLF) qui couvre la période d'une année à approximativement jusqu'à dix ans [31].

Généralement, selon le contexte étudié on peut déterminer l'horizon nécessaire pour la prévision, parce que chaque cas est unique et a des buts différents à atteindre, par exemple la prévision à court terme est le cas d'étude qui permet d'avoir un meilleur fonctionnement sécurisé du système de distribution et cela est bénéfique sur le plan économique. Entre autre, la prévision à moyen terme est importante pour prévoir les périodes des opérations et de maintenance, tandis que la prévision à long terme est faite pour améliorer la planification, le rendement des opérations du système et la gestion des actifs [32].

## 2.6 Étude de la performance des modèles

Dans la littérature, plusieurs modèles ont été mis à l'épreuve pour la prévision de la charge électrique, pour des périodes de temps et de conditions différentes. Aussi, des recherches comparatives ont été faites pour comparer la performance des modèles utilisés. Ces études nous donnent quelques idées sur quelle méthode choisir parmi la vaste gamme de techniques citées plus haut.

Dans notre domaine de recherche qui est la prévision de la charge, on trouve que les chercheurs utilisent principalement les techniques quantitatives basées sur les séries temporelles. On trouve aussi les modèles d'apprentissage machine, aussi d'autres nouvelles hybridations ont été proposées récemment, telles que les algorithmes évolutionnaires chaotiques hybrides avec SVR ou SVM des moindres carrés avec des séries temporelles floues et l'algorithme de recherche d'harmonie globale [17].

D'autres techniques incluent la logique floue [10] et les systèmes experts [11]. Dans [12], une nouvelle méthodologie basée sur l'algorithme de forêt aléatoire (*Random Forest*

*algorithm*) est utilisée pour améliorer la sélection des caractéristiques et améliorer la précision de la prévision.

À titre d'exemple, des études ont été faites pour la prévision à moyen et court terme, les auteurs de [32] ont adopté trois différentes techniques de régression (linéaire, *compound growth* et quadratique) pour prévoir la consommation d'une région. L'étude a été conduite pour la consommation pour un horizon de moyen terme pour les mois de juillet à décembre 2018.

Les mesures d'évaluation de la performance employées pour mesurer la précision de la prévision sont l'erreur moyenne en pourcentage (MAPE pour *Mean Average Percentage Error*) et l'erreur quadratique moyenne (RMSE pour *Root Mean Square Error*). D'après les trois techniques utilisées dans cette recherche, la régression linéaire a enregistré le moins de valeurs de MAPE et RMSE.

Un autre exemple de recherche, pour une prévision à court terme (quelque jours), est présenté dans [33], où les auteurs ont réalisé une étude pour améliorer la précision de la prévision. Les modèles qui ont été utilisés sont : la régression linéaire, le *long short-term memory network* et les réseaux de neurones. L'erreur relative et la RMSE ont été utilisées pour évaluer l'erreur. Les résultats sont présentés dans le Tableau 2-2. Les réseaux de neurones et la régression linéaire montrent des résultats avec une erreur relative de moins de 10 % ce qui est déjà précis pour la plupart des applications.

Tableau 2-2 Résultats d'erreurs RMSE et erreurs relatives [33].

| Modèle | RMSE | Erreur relative |
|---|---|---|
| Régression linéaire (LR) | 26.60 MWh | 2.8% |
| Long short-term memory (LSTM) | 87.50 MWh | 9.3% |
| Réseaux de neurones (ANN) | 36.98 MWh | 3.9% |

Finalement, les auteurs de [34] ont réalisé une étude de performance faite sur une recherche pour un horizon court et à long terme. Cette étude fait la comparaison entre les algorithmes basés sur la régression (2.2.1) pour les modèles utilisés suivants : la régression linéaire (LR), l'arbre de décision (DT's), le réseau neuronal profond (DNN), les réseaux de neurones récurrents (RNN), les unités récurrentes fermées (GRNN) et le *long short-term memory network* (LSTM).

Dans le tableau ci-dessous on aperçoit l'erreur (MAE) pour les six différents modèles. Les simulations sont faites avec les différentes longueurs de prévisions M = [1, 2, 12, 24, 48]. Comme on peut le voir dans le Tableau 2-3, LSTM surpasse de manière significative tous les autres modèles de prévision pour n'importe quel nombre de prévisions anticipés. Ils ont conclu alors que tous les modèles utilisant des informations temporelles (p. ex., RNN, GRNN, LSTM) surpassent clairement ceux sans modélisation des informations temporelles (LR, DT's, RNN).

Tableau 2-3 Résultats erreur MAE [34].

| M | Modèle de régression pour prévision | | | | | |
|---|---|---|---|---|---|---|
| | LR | DT | DNN | RNN | GRU | LSTM |
| 1 | 26.2% | 24.9% | 23.5% | 22.4% | 22.5% | 19.2% |
| 2 | 27.6% | 27.5% | 26.7% | 25.0% | 27.6% | 21.7% |
| 12 | 30.2% | 28.1% | 27.1% | 27.7% | 27.3% | 23.5% |
| 24 | 30.0% | 28.1% | 28.3% | 28.8% | 28.4% | 24.2% |
| 48 | 30.7% | 29.8% | 28.7% | 28.1% | 27.8% | 25.6% |

## 2.7 Conclusion

Comme vue dans ce chapitre, il y a une large gamme de méthodes et d'approches pour faire la prévision. Cependant, la ville a besoin des informations de la prévision pour anticiper les pics de consommation, donc il est utile de faire une étude à court terme et à moyen terme. En ce qui concerne la fourchette de méthode de prévision à court terme et à moyen terme, nous avons des méthodes de prévision qui donnent de très bons résultats comme montrer dans l'étude de performance.

A l'issue de cette recherche bibliographique réalisée dans ce chapitre, nous utiliserons les algorithmes de régression et ceux d'apprentissage machine supervisé, car ils sont compatibles avec nos données de recherche et permettent d'atteindre nos objectifs. Le chapitre suivant sera consacré à l'explication approfondie des méthodes choisies.

# Chapitre 3 - Réseaux de neurones et régression des séries temporelles

## 3.1 Introduction

Nous présentons dans ce chapitre les méthodes que nous avons choisi d'utiliser dans notre étude de modélisation d'analyse de prévision de la demande en puissance. Les méthodes sont basées sur les séries temporelles compte tenu des bases de données disponibles. La première méthode est celle des réseaux de neurones qui est un modèle puissant et semble pratique dans notre cas. La deuxième méthode est celle de la régression qui utilise la tendance selon laquelle le modèle de demande en puissance a une forte corrélation avec d'autres variables explicatives. Dans notre cas, une base de données des variables météorologiques est aussi disponible.

## 3.2 Le modèle de régression linéaire

La régression linéaire est le modèle de prévision le plus courant pour identifier la relation entre les variables. Hormis les types de données univariées ou multivariées, le concept est linéaire. La régression linéaire peut être soit une régression linéaire simple, soit une régression linéaire multiple.

### *3.2.1 Régression linéaire simple*

Le modèle de régression permet une relation linéaire entre la variable de prévision y et une seule variable prédictive x comme dans la relation (3-1) [14] :

$$y_t = \beta_0 + \beta_1 x_t + \varepsilon_t \qquad (3\text{-}1)$$

$\beta_0$ : Coefficient désignant l'ordonnée à l'origine.

$\beta_1$ : Coefficient désignant pente de la droite.

$\varepsilon_t$ : L'erreur.

Quand x = 0, $\beta_0$ représente la valeur prédite de y.

Un exemple de données fictive est présenté dans la Figure 3-1 :

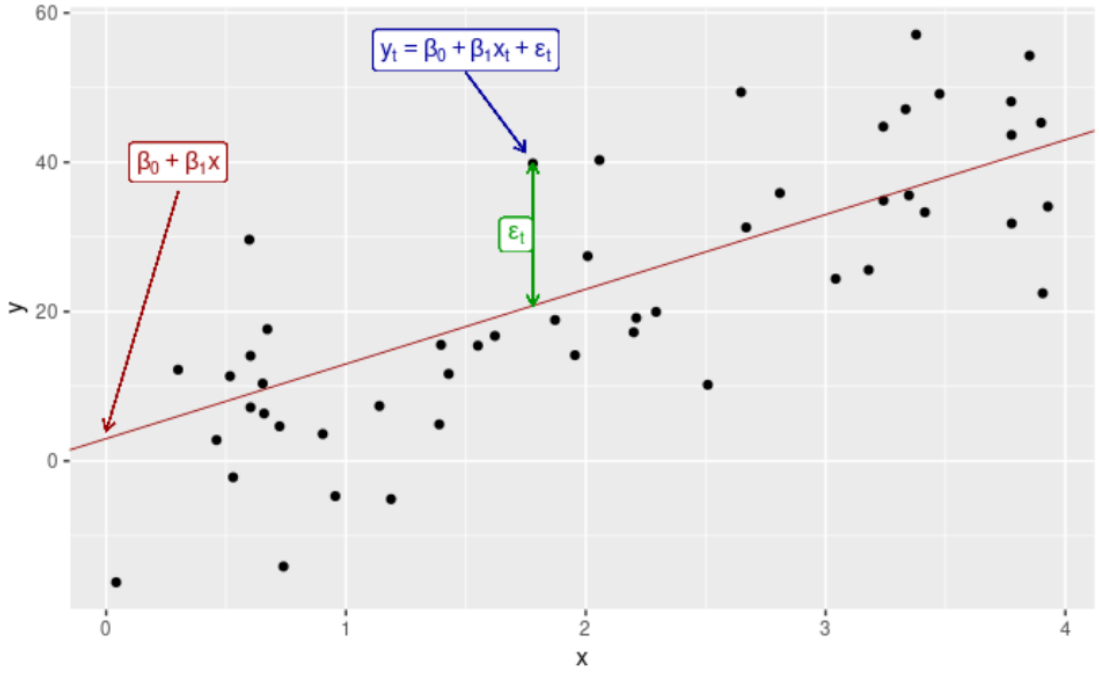



Figure 3-1 Régression linéaire simple [14].

### *3.2.2 Régression linéaire multiple*

La référence [14] explique que lorsqu'il existe deux variables prédictives ou plus, le modèle est appelé modèle de régression multiple. La forme générale d'un modèle de régression multiple est présentée dans la relation suivante :

$$y_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \cdots + \beta_k x_{k,t} + \varepsilon_t \qquad (3\text{-}2)$$

Où $y_t$ est la variable à prévoir, et $x_{1,t} \ldots x_{k,t}$ sont les variables prédictives de l'ordre $k$. Les coefficients $\beta_1 \ldots \beta_k$ mesurent l'effet de chaque prédicteur après avoir pris en compte les effets de tous les autres prédicteurs du modèle.

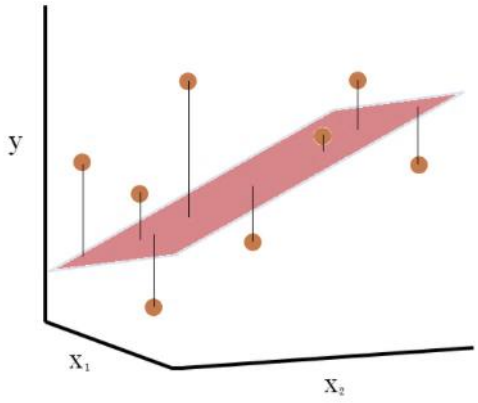



Figure 3-2 Régression linéaire multiple.

## 3.3 Le modèle de régression non linéaire

Il existe de nombreux cas dans lesquels une forme fonctionnelle non linéaire est plus adaptée. Le modèle non linéaire est juste un cas converti de la régression linéaire avec des paramètres linéaires. La variable indépendante est non linéaire et la conversion de la variable indépendante n'est pas liée au paramètre lors de la conversion. La conversion se fait si ces trois conditions sont satisfaites, la première est que le paramètre est non linéaire comme

illustré dans l'équation 3-4. En deuxième lieu, le modèle doit être en corrélation avec certains paramètres lors de la conversion de la variable indépendante comme indiqué dans l'équation 3-5. Finalement, il ne peut pas être converti en raison de la forme du terme d'erreur stochastique comme indiqué dans l'équation 3-6 [35]. Les trois équations sont définies comme suit :

$$y = \beta + \beta^2 x + \varepsilon \qquad (3\text{-}3)$$

$$y = \beta_0 + \beta_1 x^{\beta_2} + \varepsilon \qquad (3\text{-}4)$$

$$y = \beta_0 e^{\beta_1 x} + \varepsilon \qquad (3\text{-}5)$$

Où y est la variable dépendante, x est la variable indépendante, $\beta, \beta_0, \beta_1, \beta_2$ sont les paramètres et $\varepsilon$ est l'erreur.

Le modèle général de régression non linéaire peut être exprimé comme suit :

$$Y = f(X, \theta) + \varepsilon \qquad (3\text{-}6)$$

Dans cette dernière équation *Y* est le vecteur variable dépendante, *f* est la forme connue de la fonction, $\boldsymbol{X} = (x_1, x_2, \ldots x_{\mathrm{k}})^T$ est le vecteur variable indépendant, $\boldsymbol{\theta} = \left(\theta_1, \theta_2, \ldots \theta_p\right)^T$ est le vecteur de paramètres inconnus, $\boldsymbol{\varepsilon}$ est le vecteur d'erreur aléatoire. Pour répondre à l'hypothèse mentionnée dans la régression linéaire, le nombre *k* de variables indépendantes n'est pas toujours égal au nombre *p* de paramètres inconnus, et la taille d'échantillon *n* doit être supérieure au nombre *p* de paramètres inconnus.

## 3.4 Modèle de réseaux de neurones artificiels

### *3.4.1 Structure des Réseaux de neurones*

Un réseau de neurones artificiels (ANN) est défini comme un modèle de traitement d'informations qui s'anime à l'image du système nerveux du cerveau. Il est composé d'un grand nombre de corps de traitement hautement interconnectés (neurones) travaillant ensemble pour résoudre des problèmes spécifiques. L'apprentissage dans les systèmes biologiques implique des ajustements des connexions synaptiques des neurones, cela est également vrai pour les ANN.

Le bloc de construction de base de tous les cerveaux biologiques est une cellule nerveuse ou un neurone, comme le montre la Figure 3-3. Chaque neurone agit comme une unité de traitement numérique simplifiée. Essentiellement, le cerveau est composé de milliards de ces unités de traitement biologique. Dans le cerveau, chaque neurone prend plusieurs valeurs d'entrées issues des autres neurones, qui applique ensuite une fonction de transfert et envoie sa sortie à la couche suivante de ces neurones. Ces neurones envoient à leur tour leurs sorties à d'autres couches de neurones de la cascade [22].

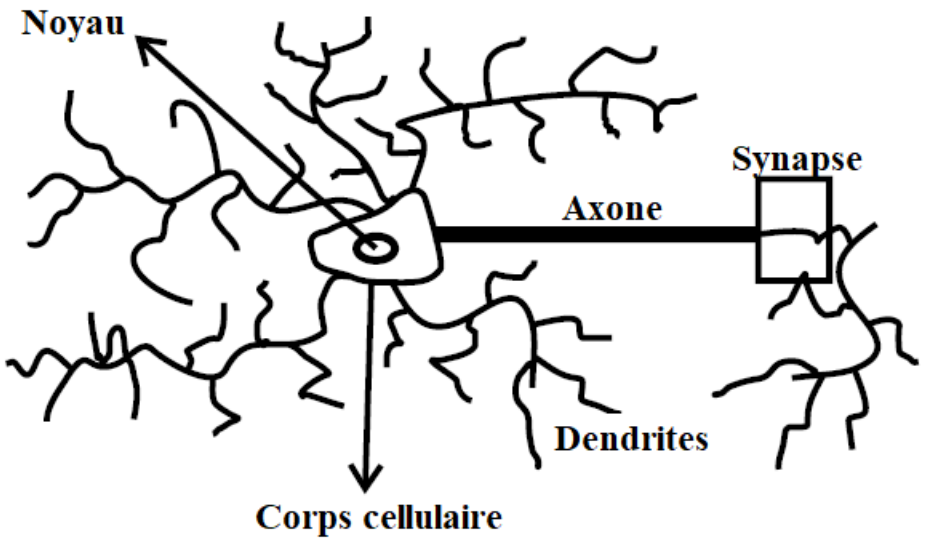



Figure 3-3 Composants de base d'un neurone [31].

En 1962, Rozenblatt introduit pour la première fois le *Perceptron* (Figure3-4), il s'agit d'une seule couche de neurones avec des poids ajustables et avec certains seuils, leurs poids doivent être supérieur au seuil pour que le neurone s'active, sinon il reste au repos. Les entrées sont $(x_1, x_2, \dots, x_n)$, les poids qui correspondent sont $(w_0, w_1, \dots, w_n)$. Comme dans la Figure 3-4 ci-dessous. Le biais (b) est un autre paramètre supplémentaire dans le réseau de neurones qui aide le modèle à s'adapter au mieux aux données. Il est utilisé pour ajuster la sortie avec la somme pondérée des entrées du neurone.

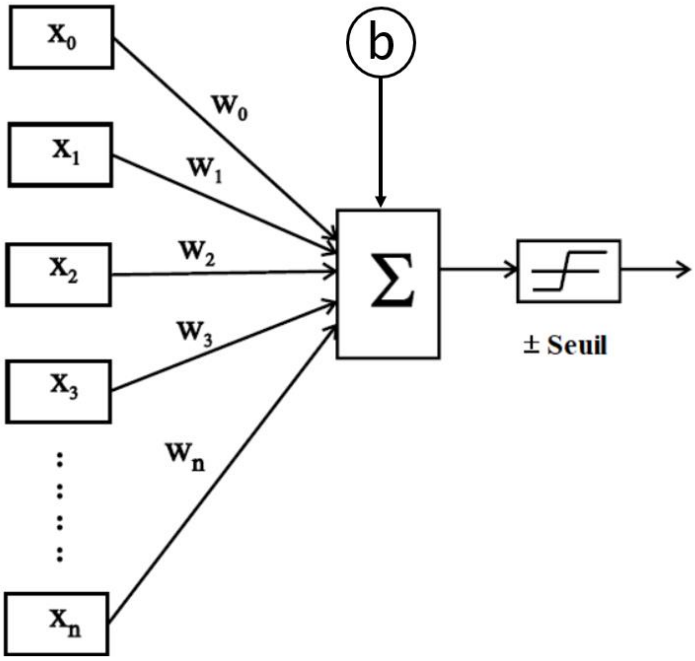



Figure 3-4 Modèle du Perceptron [22].

Avec l'avancée des recherches faites sur les ANN, on trouve que les connexions neuronales sont significativement moins nombreuses et similaires aux connexions du cerveau. Le modèle le plus utilisé est présenté dans la Figure 3-5.

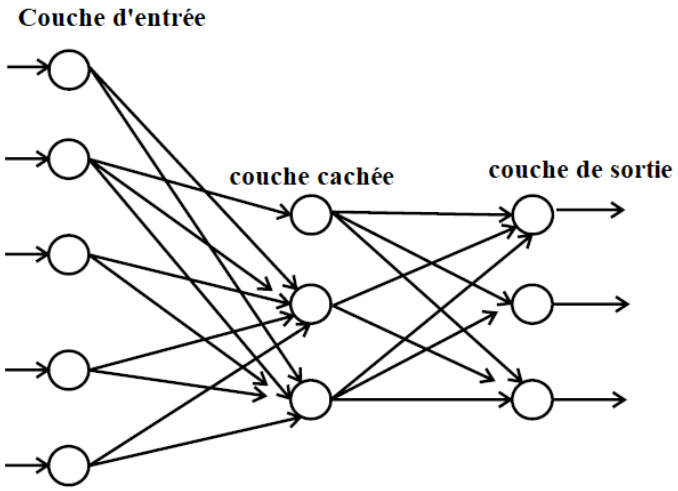



Figure 3-5 Modèle de réseau neuronal générique (ANN).

On va définir les principaux éléments du modèle de réseau neuronal générique ANN [22] :

**Neurone** : Il est représenté avec un cercle comme illustré dans la figure précédente. C'est une unité de calcul autonome. Le but de chaque neurone est de recevoir des informations provenant d'autres neurones, d'effectuer un traitement relativement simple sur les informations combinées, et d'envoyer le résultat à un ou plusieurs autres neurones.

**Couche** : C'est une collection de neurones qui peuvent être considérés comme exécutant un certain type de fonctions communes. Tous les réseaux de neurones ont une couche d'entrée et une couche de sortie pour interagir avec l'environnement externe. Chaque couche d'entrée et chaque couche de sortie comprennent au moins un neurone. Tout neurone qui n'est pas dans une couche d'entrée ou dans une couche de sortie est définie comme une couche cachée (illustrée à la Figure 3-5). Car ni son entrée ni sa sortie ne peuvent être observées de l'extérieur.

**Synapses** : C'est une liaison de communication unidirectionnelle ou bidirectionnelle entre deux cellules. Un réseau anticipatif ou en anglais *Feedforward* est celui dans lequel les informations circulent des cellules d'entrée à travers des couches cachées vers les neurones de sortie. En revanche un réseau récurrent, permet également la communication dans le sens inverse.

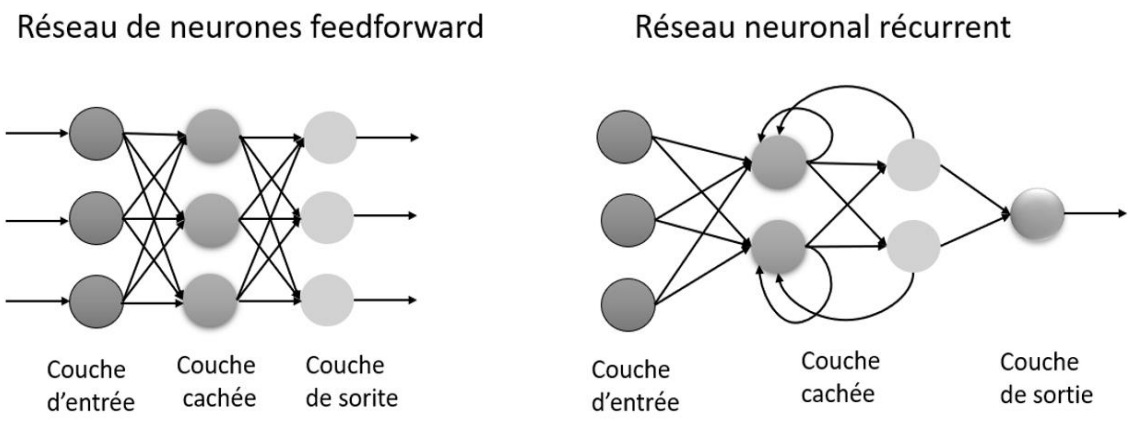



Figure 3-6 Réseau directionnel et bidirectionnel de réseaux de neurones [36].

**Poids** : $w_{ij}$, est un nombre réel qui indique l'influence d'un neurone $n_i$ sur un autre neurone $n_j$. Le cas de nullité est défini lorsqu'il n'y a pas de poids ou il n'y a aucune connexion. Ces poids peuvent être initialisés comme des valeurs données ou prédéfinies, mais ils peuvent être modifiés par l'expérience, c'est comme ça que le système apprend. Les neurones de la couche d'entrée n'ont pas de poids, c'est-à-dire que les entrées externes ne sont pas modifiées avant d'entrer dans la couche d'entrée. Alors, les poids peuvent être utilisés pour modifier l'entrée de n'importe quel neurone. Ce principe est appelé la normalisation des entrées, elle permet de traduire les données par des chiffres afin que le réseau neuronal puisse le comprendre. En effet, certaines entrées du réseau de neurones peuvent ne pas avoir une plage de valeurs définies naturellement. Les données qui ont une valeur moyenne pourraient donc augmenter lentement mais continuellement au fil du temps. Cette valeur doit être normalisée, par exemple, en indiquant au réseau combien il a changé depuis la valeur précédente. L'incrément peut être réglé pour changer et ne pas dépasser une certaine plage, ce qui deviendra en fait une bonne entrée pour le réseau neuronal.

Maintenant qu'on a vu les composants du modèle, on va voir la règle de propagation. Le neurone consiste essentiellement en un intégrateur qui effectue la somme pondérée de ses entrées. Le résultat *n* de cette somme est ensuite transformé par une fonction de transfert *f* qui produit la sortie 'a' du neurone. Comme montré dans la Figure 3-7 et l'équation 3-8 [37].

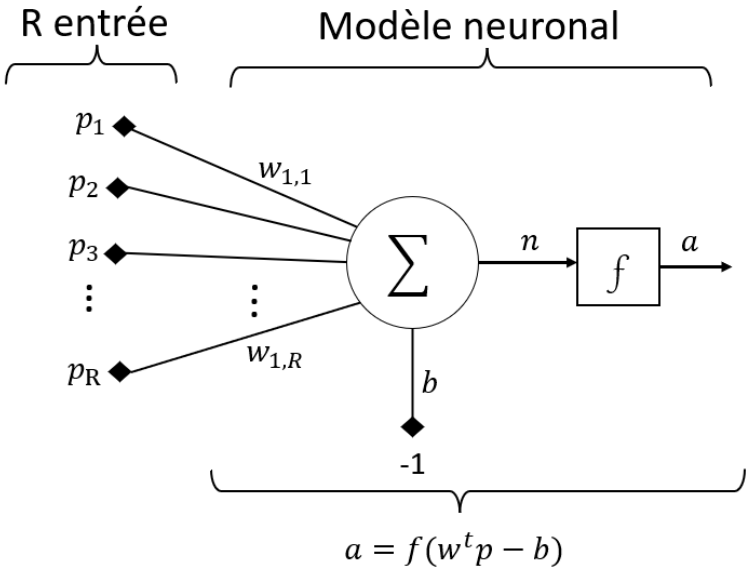



Figure 3-7 Modèle de neurone artificiel [37].

$$n = \sum_{j=1}^{R} w_{1,j} p_j - b \qquad (3\text{-}7)$$

$$n = w_{1,1} p_1 + w_{1,2} p_2 + \cdots + w_{1,R} p_R - b \qquad (3\text{-}8)$$

Cette sortie correspond à une somme pondérée des poids $w_{i,j}$ et des entrées $p_j$ moins le biais *b*.

### *3.4.2 Entraînement du réseau de neurones*

L'apprentissage est le processus par lequel le réseau de neurones s'adapte à un stimulus, et finalement, il produit une réponse souhaitée. Les expressions mathématiques, les équations d'apprentissage décrivent le processus d'apprentissage du paradigme, qui est en réalité le processus d'auto-ajustement de ses poids synaptiques.

#### 3.4.2.1 Apprentissage supervisé

Lors de la formation d'un réseau de neurones, les stimuli d'entrée produisent des réponses de sortie. Si la réponse réelle est différente de la réponse cible, la réponse est comparée au signal de sortie souhaité précédent (c'est-à-dire la réponse cible); le réseau de neurones génère un signal d'erreur, qui est ensuite utilisé pour calculer les ajustements à apporter à la synaptique du réseau pondéré afin que la sortie réelle corresponde à la sortie cible. En d'autres termes, l'erreur est minimisée, ou nulle avec le temps. Le processus de minimisation des erreurs nécessite un circuit spécial appelé superviseur d'où vient le nom d'entraînement supervisé [38].

#### 3.4.2.2 Apprentissage non supervisé

L'apprentissage non supervisé ne nécessite pas d'enseignant, c'est-à-dire qu'il n'y a pas de sortie cible. Pendant la formation, le réseau de neurones reçoit de nombreux stimuli ou modèles d'entrée différents à son entrée et organise arbitrairement les modèles en catégories. Lorsque le stimulus est ensuite appliqué, le réseau neuronal fournit une réponse de sortie indiquant la catégorie à laquelle appartient le stimulus. Si aucune classe pour le stimulus d'entrée n'est trouvée, une nouvelle classe peut être générée [30] [22].

#### 3.4.2.3 Apprentissage compétitif

L'apprentissage compétitif est une autre forme d'apprentissage supervisé qui se caractérise par des opérations et une architecture avec plusieurs neurones dans la couche de sortie. Lorsqu'un stimulus d'entrée est appliqué, chaque neurone de sortie entre en compétition avec d'autres neurones pour produire le signal de sortie le plus proche de la cible. Cette sortie devient alors la sortie principale et les autres sorties cessent de produire des signaux de sortie pour ce stimulus. Pour un autre stimulus, un autre neurone de sortie devient le neurone dominant, et ainsi de suite [22].

#### 3.4.2.4 Apprentissage par la rétropropagation d'erreur

Chaque vecteur d'entrée qui passe à travers le réseau devient un vecteur de sorite calculé. Pendant la formation, la sortie du réseau est comparée aux observations réelles et un terme d'erreur est créé. Cette erreur est renvoyée à travers le réseau, de la couche de sortie, à travers la couche cachée, et de nouveau à la couche d'entrée, avec des poids d'interconnexion entre chaque couche ajustée en fonction des paramètres d'erreurs et de taux d'apprentissage calculés.

Le processus de présentation des données d'entrée, de leur transmission à travers le réseau et de rétropropagation de l'erreur est répété pour chaque observation dans l'ensemble d'apprentissage historique. L'ensemble d'apprentissage est affiché plusieurs fois sur le réseau jusqu'à ce que les valeurs de sortie convergent vers une solution.

Algébriquement, dans un perceptron multicouche, l'équation qui décrit la sortie de la couche k est définie comme suit [37] :

$$a^k = f^k(W^k a^{k-1} - b^k) \qquad (3\text{-}9)$$

Pour k=1,…,M , où M est le nombre total des couches. Les sorties du réseau correspondent à $a^M$. L'algorithme de rétropropagation utilise l'erreur quadratique moyenne comme indice de performance, et permet un apprentissage supervisé avec un ensemble d'associations (stimulus, cible) $\{(\mathrm{p}_q, \mathrm{d}_q)\}, \mathrm{q} = 1, \ldots, \mathrm{Q}$ où $p_q$ représente l'entrée (vecteur stimulus) et $d_q$ la sortie (vecteur cible). Pour chaque moment dans le temps *t* nous pouvons propager en avant un stimulus différent p(t) sur le réseau de neurones, pour obtenir le vecteur de sortie a(t) comme dans la Figure 3-7. Cela nous permet de calculer l'erreur e(t) entre ce que le réseau produit en sortie du stimulus et la cible d(t) que lui est associée [37] :

$$e(t) = d(t) - a(t) \tag{3-10}$$

L'erreur quadratique moyenne est minimisée avec l'indice de performance *F*. Cet indice est approché par l'erreur instantanée $\hat{F}(x)$, et le vecteur x inclut tous les poids et le biais du réseau, comme dans la formule 3-11 :

$$F(\mathrm{x}) = E[\mathrm{e}^T(t)\mathrm{e}(t)] \tag{3-11}$$

Où *E* désigne l'espérance mathématique et le vecteur x regroupe l'ensemble des poids et des biais du réseau, alors on peut approximer cet indice par l'erreur instantanée :

$$\hat{F}(x) = e^T(t)e(t) \tag{3-12}$$

Alors pour optimiser x, la méthode de descente de gradient est utilisée :

$$\Delta w_{i,j}^k(t) = -\eta \frac{\partial \hat{F}}{\partial w_{i,j}^k} \;\; ; \;\; \Delta b_i^k(t) = -\eta \frac{\partial \hat{F}}{\partial b_i^k} \tag{3-13}$$

La cadence d'apprentissage est $\eta$ alors :

$$\frac{\partial \hat{F}}{\partial w_{i,j}^k} = \frac{\partial \hat{F}}{\partial n_i^k} \times \frac{\partial n_i^k}{\partial w_{i,j}^k} \tag{3-14}$$

$$\frac{\partial \hat{F}}{\partial b_i^k} = \frac{\partial \hat{F}}{\partial n_i^k} \times \frac{\partial n_i^k}{\partial b_i^k} \quad (3\text{-}15)$$

$n_i^k$ représente les niveaux d'activation d'une couche k qui dépend directement des poids et le biais de cette couche.

$$n_i^k = \sum_{j=1}^{s^{k-1}} w_{i,j}^k a_j^{k-1} - b_i^k \quad (3\text{-}16)$$

Alors (3-14) et (3-15) deviennent :

$$\frac{\partial n_i^k}{\partial w_{i,j}^k} = a_j^{k-1} \quad , \frac{\partial n_i^k}{\partial b_i^k} = -1 \quad (3\text{-}17)$$

On introduit la sensibilité $S_i^k = \frac{\partial \hat{F}}{\partial n_i^k}$ alors redevient

$$\Delta w_{i,j}^k(t) = -\eta s_i^k(t) a_j^{k-1}(t); \Delta b_i^k(t) = \eta s_i^k(t) \quad (3\text{-}18)$$

$$\text{Avec } s^k = \left(\frac{\partial n^{k+1}}{\partial n^k}\right)^T \frac{\partial \hat{F}}{\partial n^{k+1}} = \dot{F}^k(n^k)(W^{k+1})(s^{k+1}) \quad (3\text{-}19)$$

On obtient une formule récursive où la sensibilité des couches en amont (entrée $(s^k)$) dépend de la sensibilité des couches en aval (sortie $(s^{k+1})$). C'est de là que vient le terme rétropropagation, car le sens de propagation de l'information est inversé par rapport à celui de (3-19) [37] [39].

## 3.5 Corrélation

Pour mesurer la force de corrélation entre deux variables x et y par exemple, il suffit de calculer la valeur du coefficient $R$ avec (3-20) [40]. Celui-ci est toujours compris entre -1 et 1. Les valeurs négatives indiquent une relation négative et des valeurs positives indiquent une relation positive.

$$R = \frac{E\{(R_t - \mu_R)(R_t - \mu_R)\}}{\sqrt{\sum(R_t - \mu_R)^2 * \sum(F_t - \mu_F)^2}} \quad (3\text{-}20)$$

Où $\mu_F$ est la moyenne de la valeur prévue, $\mu_R$ est la moyenne de la valeur réelle et $E$ est l'opérateur de valeur attendue.

Dans le tableau suivant on voit la puissance de corrélation entre les deux variables.

Tableau 3-1 La corrélation.

| Valeur du coefficient R | Force de corrélation |
|---|---|
| 0 à ±0.25 | Corrélations négligeables |
| De ±0.25 à ±0.50 | Corrélations faibles |
| De ±0.50 à ±0.75 | Corrélations moyennes |
| De ±0.75 à ±1 | Corrélations fortes |
| ±1 | Corrélations parfaites |

### 3.6 Prédictibilité des données

Quand on fait la prévision, il est important de faire une étude de prédictibilité des données en possession. La méthode la plus répandue pour l'évaluation de la prédictibilité est celle du calcul du coefficient ou exposant de Hurst.

L'exposant de Hurst (H) peut être utilisé comme mesure de la mémoire à long terme d'une série temporelle, c'est-à-dire une mesure de l'écart de cette série par rapport à une marche

aléatoire. Un scalaire représentant la tendance relative d'une série chronologique, soit revenant brusquement à la moyenne (retour au modèle moyen), soit convergeant dans une certaine direction (modèle de tendance) [41].

Ce coefficient H est défini comme un comportement asymptotique avec une plage de remise à l'échelle en fonction de l'intervalle de la série chronologique comme suit [42] :

$$E\left[\frac{R(n)}{S(n)}\right] = Cn^H \qquad (3\text{-}21)$$

Où n est l'espace de temps de l'observation (nombre de points de données dans une série chronologique), ce dernier tend vers l'infini. R (n) est la plage de premières valeurs de N, et S(n) est l'écart-type. E(x) est la valeur attendue et C une constante.

Afin d'estimer cet exposant H, il faut d'abord estimer la dépendance de la gamme rééchelonnée sur l'espace de temps d'observation n. Ensuite, une série temporelle de longueur N est divisée en un certain nombre de séries chronologiques plus courtes de longueur *n = N, N/2, N/4, ....* Enfin, la gamme rééchelonnée moyenne est calculée pour chaque valeur de n.

Le calcul de la gamme rééchelonnée pour une série de temps de longueur n , *X = X1, X2,...Xn* est évalué comme suit[43]:

Calcul de la moyenne m :

$$m = \frac{1}{n}\sum_{i=1}^{n} X_i \qquad (3\text{-}22)$$

Et la création d'une série moyenne ajustée $Y_t = X_t - m$ ,avec variant de 1 à n.

Calcul de la série à écart cumulatif :

$$Z_t = \sum_{i=1}^{t} Y_i \quad \text{avec } t = 1,2,\dots\dots,n \qquad (3\text{-}23)$$

Calcul de la gamme R :

$$R(n) = max(Z_1, Z_2, \dots, Z_n) - min(Z_1, Z_2, \dots, Z_n) \quad (3\text{-}24)$$

Calcul de l'écart type S :

$$S(n) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (X_i - m)^2} \quad (3\text{-}25)$$

Calcul de la gamme rééchelonnée $R(n)/\,S(n)$ et la moyenne sur toutes les séries de temps partielles de longueur n.

Alors, pour calculer l'exposant de Hurst on ajuste l'équation 3-21 aux données. Cela est matérialisé en traçant le logarithme de $E\left[\frac{R(n)}{S(n)}\right]$. En fonction de log n, la pente de la droite donne H [43].

Les valeurs de l'exposant de Hurst sont comprises entre 0 et 1. Sur la base de la valeur de H, nous pouvons classer n'importe quelle série chronologique dans l'une des trois catégories comme dans le tableau suivant [41] :

Tableau 3-2 Type de série selon la valeur de l'exposant de Hurst.

| | |
|---|---|
| H = 0.5 | Marche aléatoire. |
| H > 0.5 | Une série tendance (persistante). Plus la valeur est proche de 1, plus la tendance est forte. En pratique, cela signifie qu'une valeur élevée est suivie d'une valeur supérieure. |
| H < 0.5 | Une série de retour à la moyenne (anti-persistante). Plus la valeur est proche de 0, plus le processus de retour à la moyenne est fort. En pratique, cela signifie qu'une valeur élevée est suivie d'une valeur faible et vice-versa. |

## 3.7 Saisonnalité

Une première étape à faire dans tout travail d'analyse de données est de tracer des courbes des données. Les graphiques visualisent de nombreuses caractéristiques des données, y compris les observations inhabituelles (un événement externe inattendu), les changements dans le temps et les relations entre les variables pouvant mener à des modèles. Lors de la représentation graphique des séries temporelles, les observations sont tracées en fonction du temps d'observation. On mentionne trois types de motif (courbe) pour bien comprendre la donnée et avoir une prévision plus crédible.

**Tendance** : Une tendance existe quand il y a un accroissement ou une diminution à long terme des données. Il peut y avoir le cas de passer d'une tendance à la hausse à une tendance à la baisse.

**Saisonnière** : La saisonnalité est toujours d'une fréquence fixe et connue. Autrement dit, lorsqu'une série chronologique est affectée par des facteurs saisonniers par exemple la période de l'année ou le jour de la semaine.

**Cyclique** : Un cycle a lieu quand les données présentent des hausses et des baisses qui ne sont pas d'une fréquence fixe.

La Figure 3-8 illustre des exemples fictifs pour montrer les différents motifs de série temporelle. La figure en haut à gauche présente une forte saisonnalité au cours de chaque année, ainsi qu'un fort comportement cyclique avec une période d'environ 6 à 10 ans. Dans la figure en haut à droite, on n'aperçoit pas de saisonnalité, mais une tendance à la baisse évidente. La figure en bas n'a pas de tendance, ni de saisonnalité, ni comportement cyclique. Il existe des fluctuations aléatoires qui ne semblent pas très prévisibles.

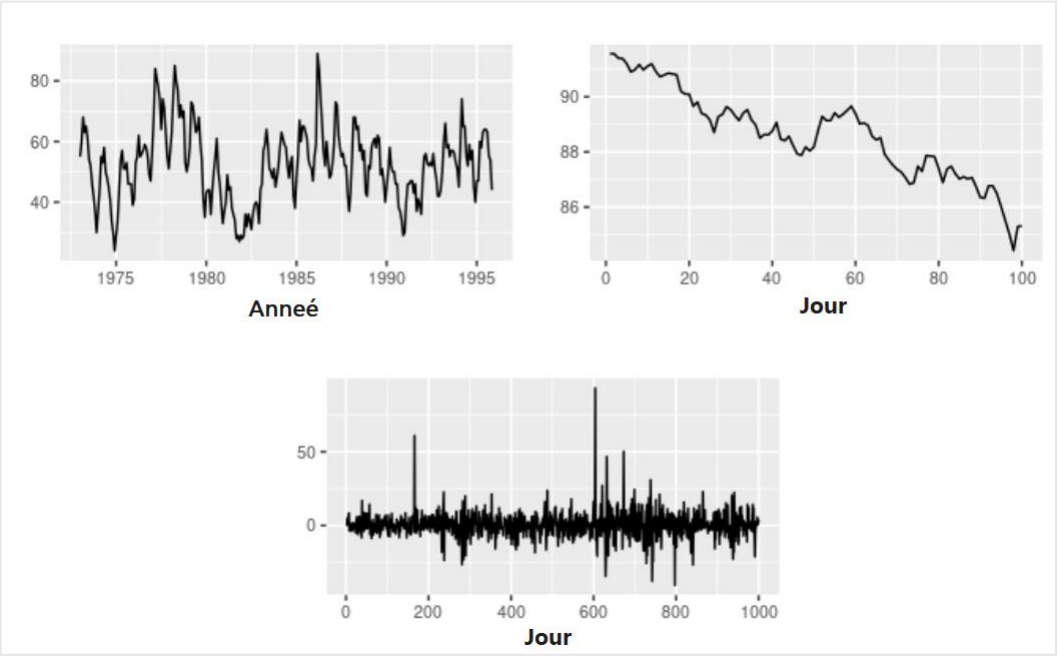



Figure 3-8 Exemple de graphe de série temporelle.

### 3.8 Efficacité et évaluation des modèles

Pour évaluer la précision des modèles proposés on utilise les méthodes d'évaluation suivantes : l'erreur absolue moyenne (MAE), l'erreur quadratique moyenne (RMSE) et l'erreur absolue moyenne en pourcentage (MAPE) [44].

$$MAE = \frac{1}{n}\sum_{t=1}^{n} |A_t - F_t| \qquad (3\text{-}26)$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} (A_t - F_t)^2} \qquad (3\text{-}27)$$

$$MAPE = \frac{1}{n}\sum_{t=1}^{n} \left|\frac{A_t - F_t}{A_t}\right| \times 100\% \qquad (3\text{-}28)$$

Où $F_t$ est la valeur prévue et $A_t$ est la valeur réelle à l'instant ou échantillon t.

En ce qui concerne MAPE les critères d'évaluation sont donnés dans le Tableau 3-3.

Tableau 3-3 Signification de l'erreur MAPE.

| **MAPE (%)** | **Puissance de la prévision** |
|---|---|
| <10 | Prévision très précise |
| 10 - 20 | Bonne prévision |
| 20-50 | Prévision raisonnable |
| >50 | Prévision inexacte |

## 3.9 Conclusion

Ce chapitre permet de bien comprendre la théorie derrière les réseaux de neurones et la régression. Ces deux méthodes ont été choisies pour faire la prévision à court et à moyen terme ciblant les lignes de distribution résidentielles et hybrides (résidentiel et commercial).

Le chapitre suivant montre comment les lignes de distribution sont catégorisées pour permettre de faire une description du réseau de la ville de Baie-Comeau. Ensuite, voir les résultats acquis avec nos modèles de prévision choisis.

# Chapitre 4 - Étude de prévision

## 4.1 Introduction

Dans ce chapitre, nous présenterons nos travaux ainsi que les résultats obtenus pour la prévision de la consommation pour le réseau de distribution de la ville de Baie-Comeau.

Pour ce faire, nous commençons par présenter en détail les différentes bases de données qui ont été mises à notre disposition. Ensuite, nous montrons comment elles ont été collectées, filtrées et organisées.

Par la suite on présente comment ces bases de données ont été utilisées pour le développement de modèles avec les deux méthodes choisies et expliquées dans le chapitre précédent, à savoir les réseaux de neurones (ANN) et la régression linéaire. Ceci en utilisant les logiciels MATLAB et Python. Enfin, dans ce chapitre, on va présenter et discuter les résultats obtenus pour une prévision de la consommation à moyen terme et à court terme.

## 4.2 Description de la base de données

Dans le but d'avoir une bonne base de données, il est nécessaire de récolter un maximum d'informations quantitatives avec différents paramètres dans un espace-temps le plus large possible. Dans le cas de notre étude, nous avons utilisé différentes bases de données dont les paramètres météorologiques sont liés au fonctionnement du réseau électrique de la ville. Ces bases de données sont issues du site web *Simeb* [45]. Ce site offre des informations horaires

concernant la température sèche, la température humide, la vitesse du vent et l'irradiation solaire.

Nous avons voulu au préalable avoir des informations pour chaque quart d'heure pour tous les paramètres météorologiques. En fait, c'est pour correspondre à la fréquence des relevés de données de la ville, car la ville fait des extractions de données chaque quinze (15) minutes.

On a trouvé la résolution au problème de relevé de paramètres (correspondance de temps) en utilisant l'extrapolation des données avec le logiciel MATLAB et avec la libraire de Python (*scipy.interpolate*).

Le site web *Simeb* propose des données bien détaillées sur différentes zones au nord du Canada avec un historique d'une dizaine d'années pour la ville de Baie-Comeau. Cependant, dans notre travail, nous avons choisi de prendre en considération uniquement les données partant du premier jour de l'année 2020 au 31 Décembre de la même année. Le choix de cette période est lié à la base de données qui est fournie par la ville de Baie-Comeau.

La Figure 4-1 est une représentation graphique permettant de visualiser les nombreuses caractéristiques des données climatiques. On voit que les températures sèches et humides ainsi que les irradiations suivent le motif saisonnier, par contre le vent a un comportement totalement aléatoire sans le caractère saisonnier ou comportement cyclique.

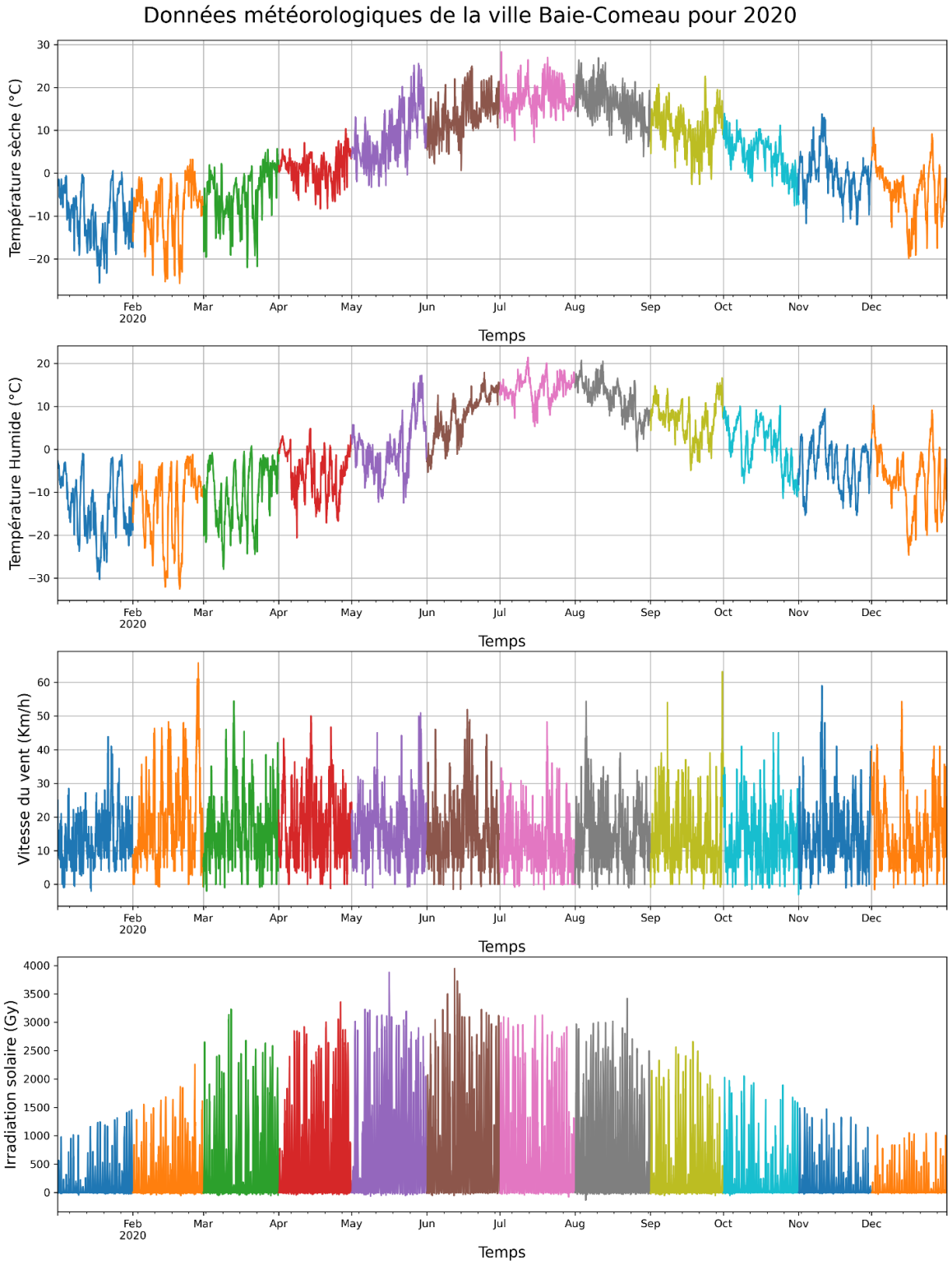



Figure 4-1 Données météorologiques de la ville Baie-Comeau 2020.

En ce qui concerne les données réelles de consommation de la ville, nous avons réalisé une analyse et un tri sur ces données brutes afin de faciliter le traitement. Nous notons ici que les logiciels que nous avons utilisés pour le traitement de données sont MATLAB, Excel et Python (*Anaconda*). Ces trois outils permettent de bien gérer et de classifier de grands nombres de données numériques d'une façon efficace. Car les données réelles étaient sous forme de texte (.txt), elles ont été converties pour être classifiées sur Excel (.xlsx), ainsi chargées et utilisées sur MATLAB et Python sous forme de vecteur et de matrice (.mat).

Après cette analyse, on identifie dans le réseau de la ville huit lignes de distribution distinctes qui alimentent la ville. Chaque ligne alimente en électricité un ou plusieurs types de consommateur par catégorie comme suit : résidentielle, commerciale, industrielle et autre.

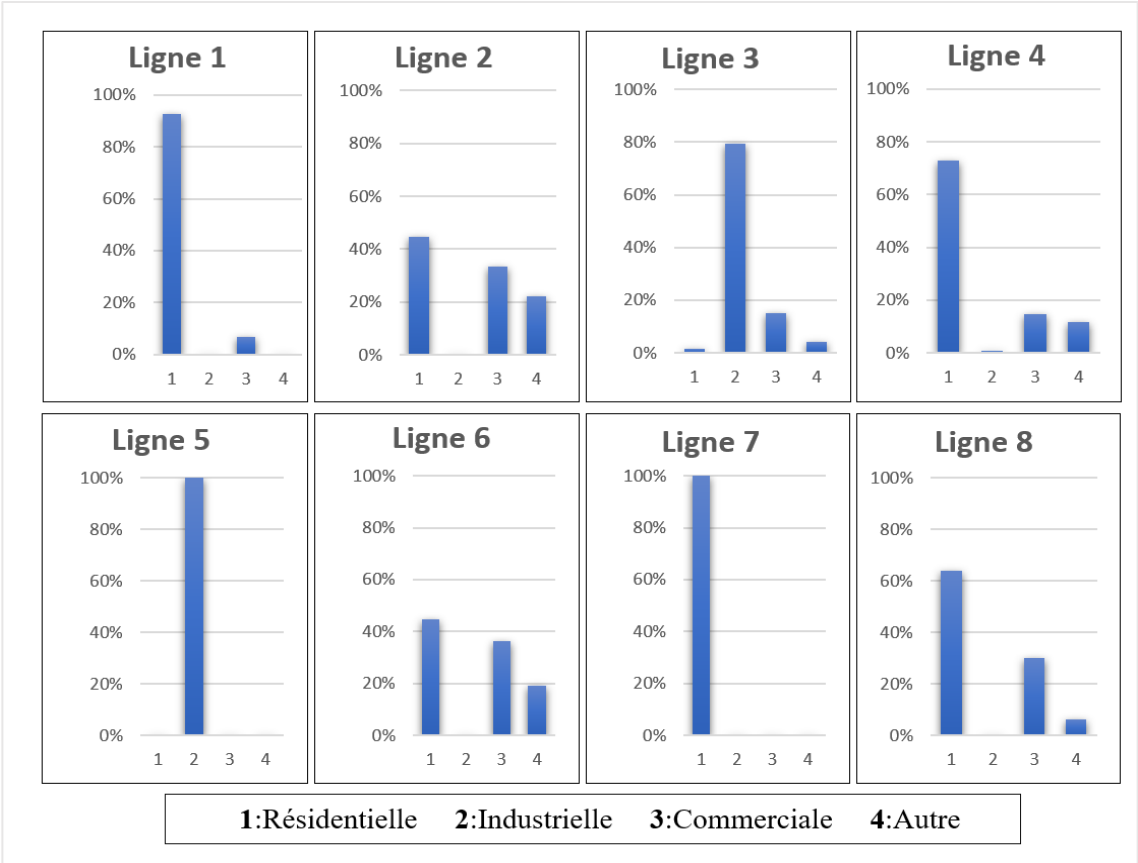



Figure 4-2 Les huit lignes de distribution catégorisées par type de clients.

Nous pouvons voir ainsi chaque ligne de distribution avec sa vocation qui la caractérise. Nous prenons l'exemple de la ligne 5 qui représente une distribution absolument industrielle. Cependant, les autres lignes font un mélange de toutes les catégories de consommateurs déjà citées auparavant, mais avec des pourcentages bien différents. Nous pouvons catégoriser les lignes 1, 4, 7 et 8 comme des distributions résidentielles. Les lignes 3 et 5 sont des distributions industrielles et elles sont principalement liées au centre de traitement de données (minage de cryptomonnaie). Les lignes 2 et 6 sont hybrides entre commerciales et résidentielles.

La ville fait des extractions de données de manière périodique durant toute l'année avec 96 prélèvements quotidiens. Ces derniers concernent la consommation électrique ainsi que tous les paramètres liés aux réseaux électriques pour les huit lignes de distribution. La Figure 4-3, ci-dessous, illustre un graphe de série temporelle suivant la consommation en Mégawatts pour chaque ligne de distribution durant l'année 2020.

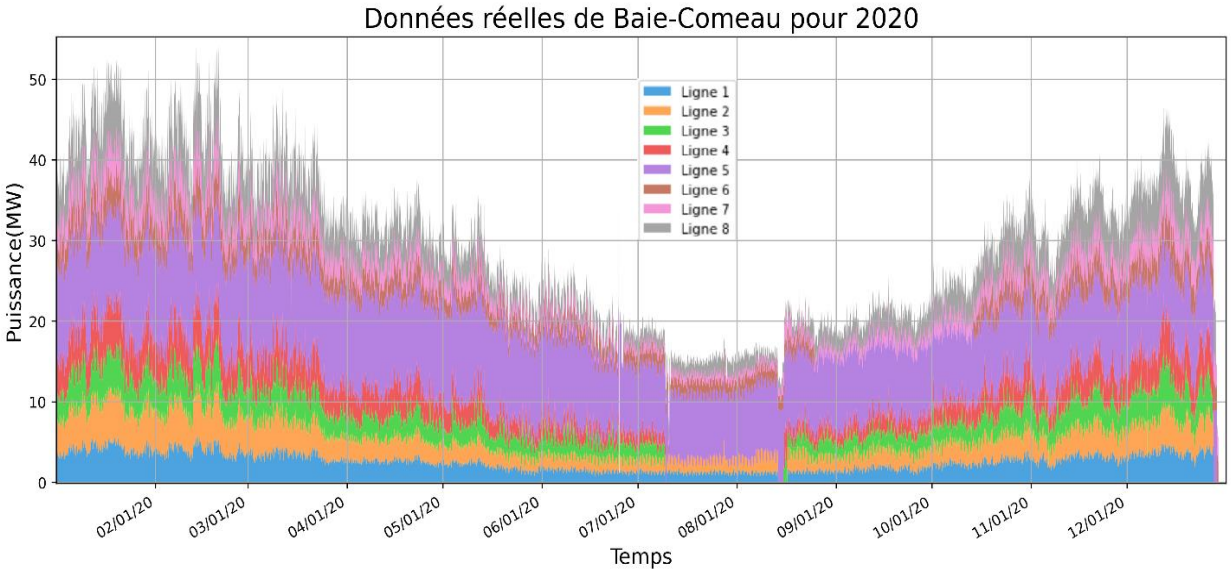



Figure 4-3 Les données réelles durant l'année 2020 de la ville de Baie-Comeau.

Dans le but de distinguer la consommation quotidienne de l'énergie totale enregistrée pour chaque type de client, nous avons pris un exemple de prélèvement d'une seule journée du mois de novembre 2020. Ce prélèvement nous permet de voir de combien consomme chaque type de client du total enregistré durant cette journée-là.

La puissance totale était de 29,40 MW. D'après la Figure 4-4, nous remarquons que la ville de Baie-Comeau a plutôt une vocation industrielle. L'énergie consommée est donc d'une grande part industrielle. On peut voir aussi la consommation de chaque ligne dans la Figure 4-5.

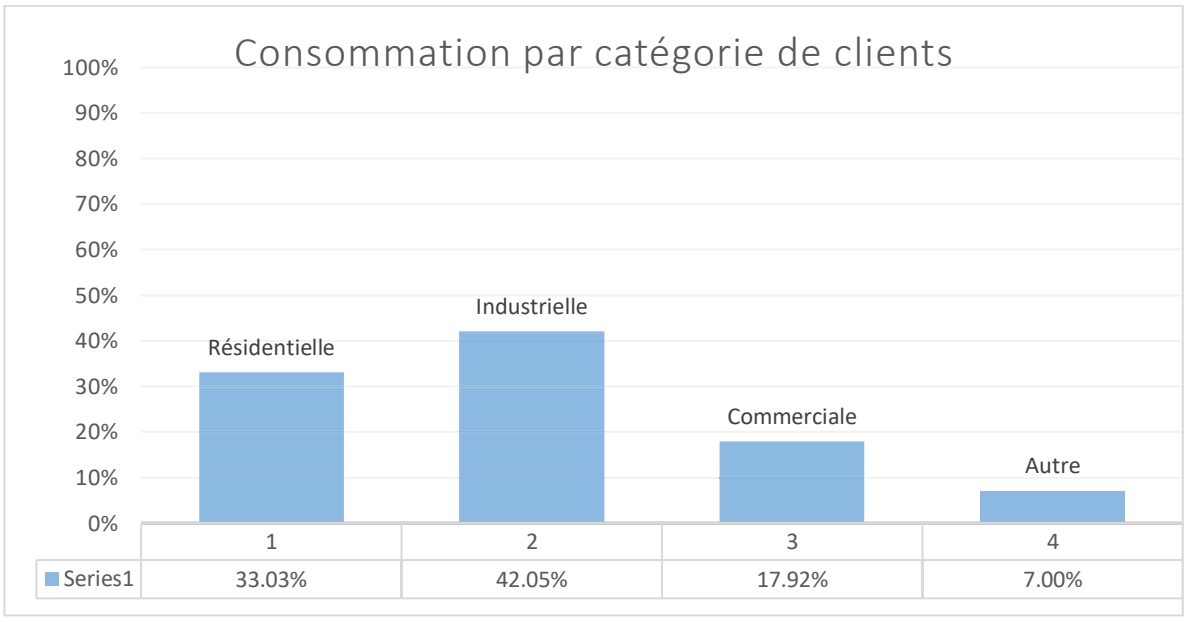



Figure 4-4 La consommation par catégorie de clients du total lors d'un prélèvement en Novembre 2020.

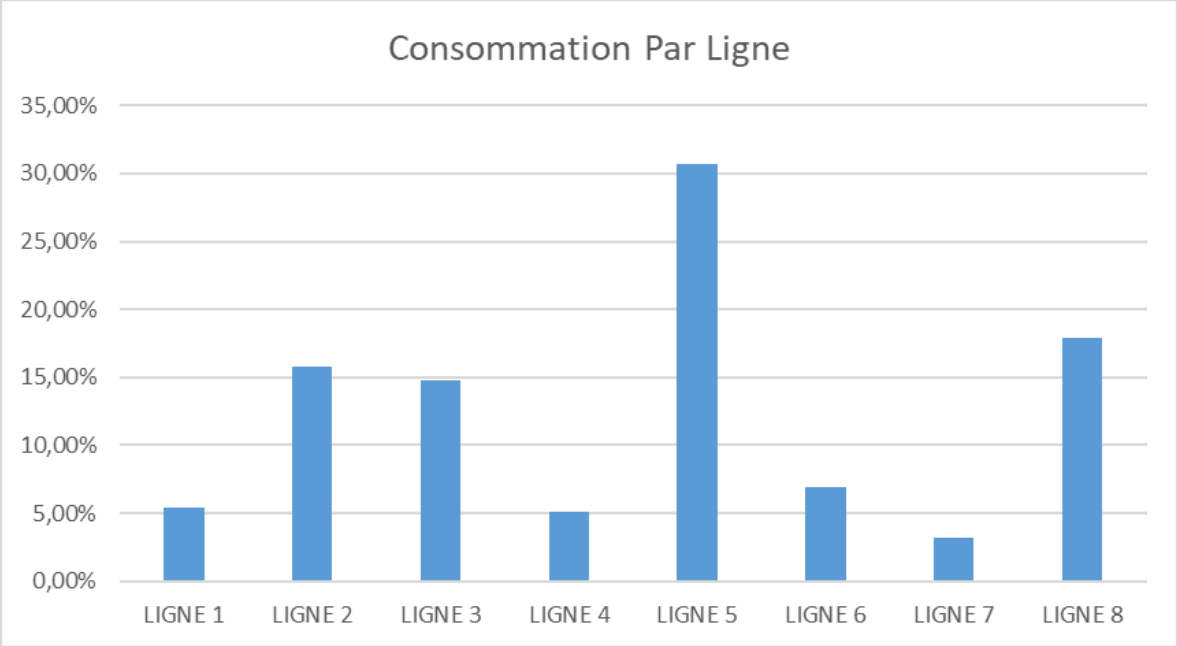



Figure 4-5 La consommation par ligne lors d'un prélèvement en Novembre 2020.

## 4.3 Étude de corrélation

Dans notre cas, la demande en puissance est souvent connue pour être sensible aux changements météorologiques, étant donné que tout au long de l'année la fluctuation météorologique affecte la demande en puissance. Cependant, pour évaluer cette relation entre les variables météorologiques et la consommation nous sommes appelés à utiliser la corrélation. Cette étude de corrélation joue un rôle très important pour l'efficacité de la prévision et le bon fonctionnement du modèle. Pour déterminer les paramètres influençant sur la demande en puissance, on cherche à trouver le coefficient de corrélation R qui nous permet de déterminer la force de la corrélation entre la grandeur de la météo (température sèche et humide, vent et ensoleillement), et la consommation de chaque ligne, car elle influe directement sur le comportement des catégories des consommateurs industriels, résidentiels ou commerciaux. Le coefficient R est calculé sous Python en se basant sur la relation 3-20.

Les calculs ont été réalisés avec l'utilisation des bibliothèques suivantes : *Seaborn, Pandas, Matplotlib.pyplot* et *Numpy*.

Les figures suivantes montrent comment les quatre facteurs météorologiques déjà cités plus haut influent sur la consommation de chacune des huit lignes de distribution, et ceci durant chaque saison de l'année 2020. La figure 4-6 concerne la période d'hiver, en absence des données du mois de décembre 2019, elle débute du premier Janvier 2020 jusqu'au 19 Mars 2020. Puis la figure 4-7 représente celle du printemps qui commence le 20 Mars 2020 jusqu'au 19 Juin. Ensuite, la figure 4-8 correspond à la saison d'été qui commence du 20 Juin au 22 Septembre 2020. Finalement la figure 4-9 représente la saison d'automne avec quelques jours d'hiver  qui commence du 23 Septembre au 31 Décembre 2020.

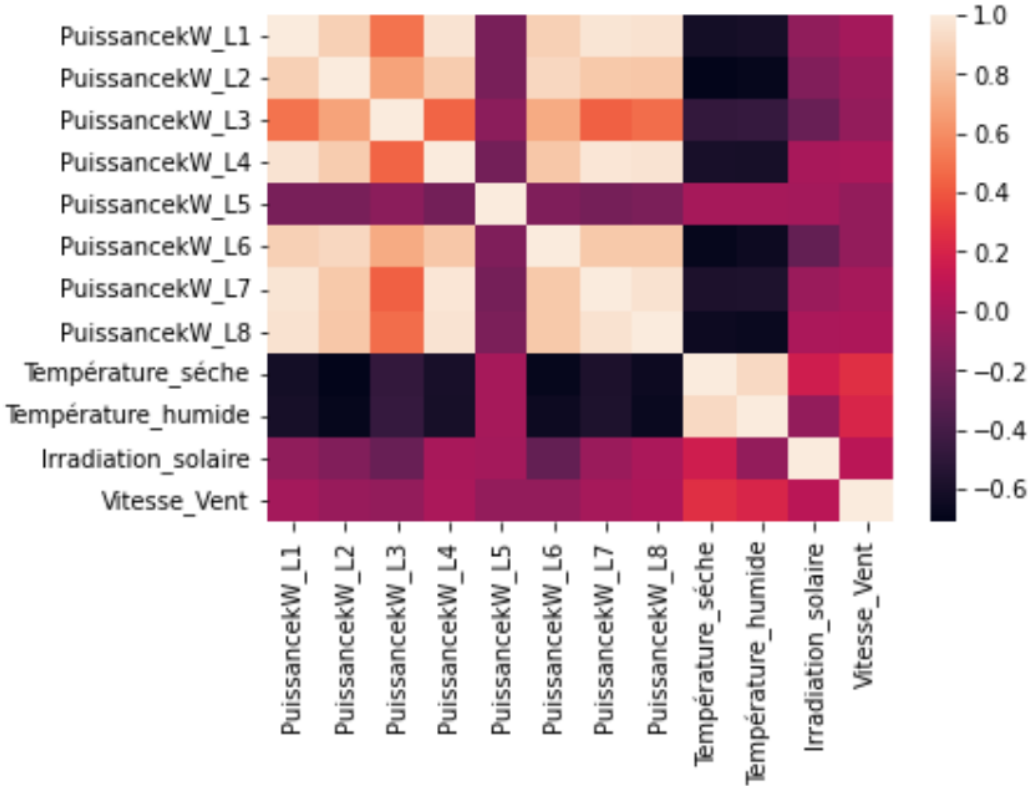



Figure 4-6 La matrice de corrélation pour l'hiver 2020.

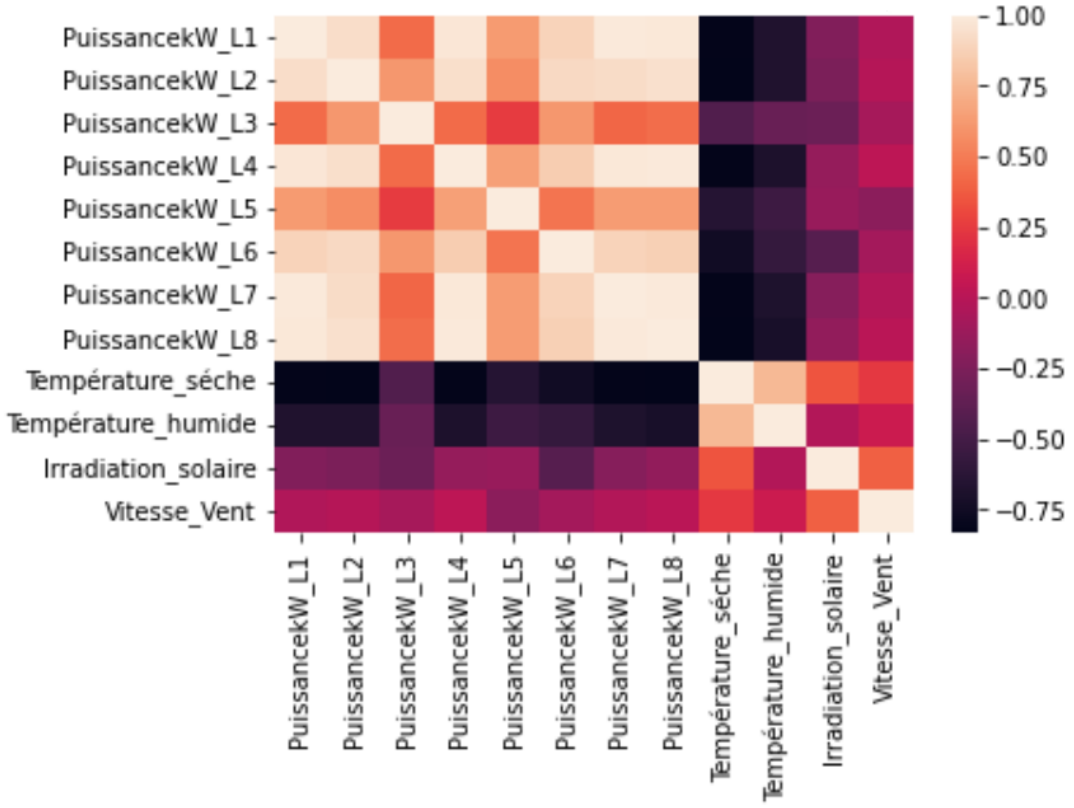



Figure 4-7 La matrice de corrélation pour le printemps 2020.

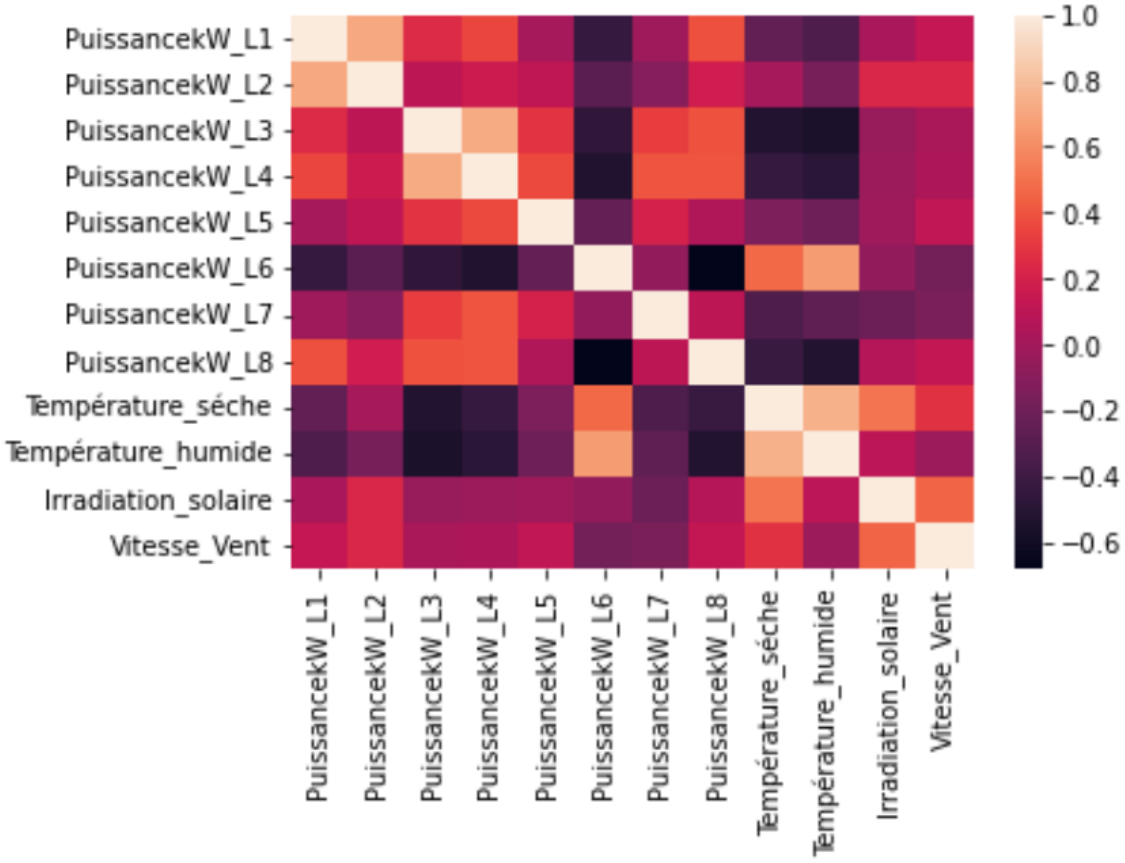



Figure 4-8 La matrice de corrélation pour l'été 2020.

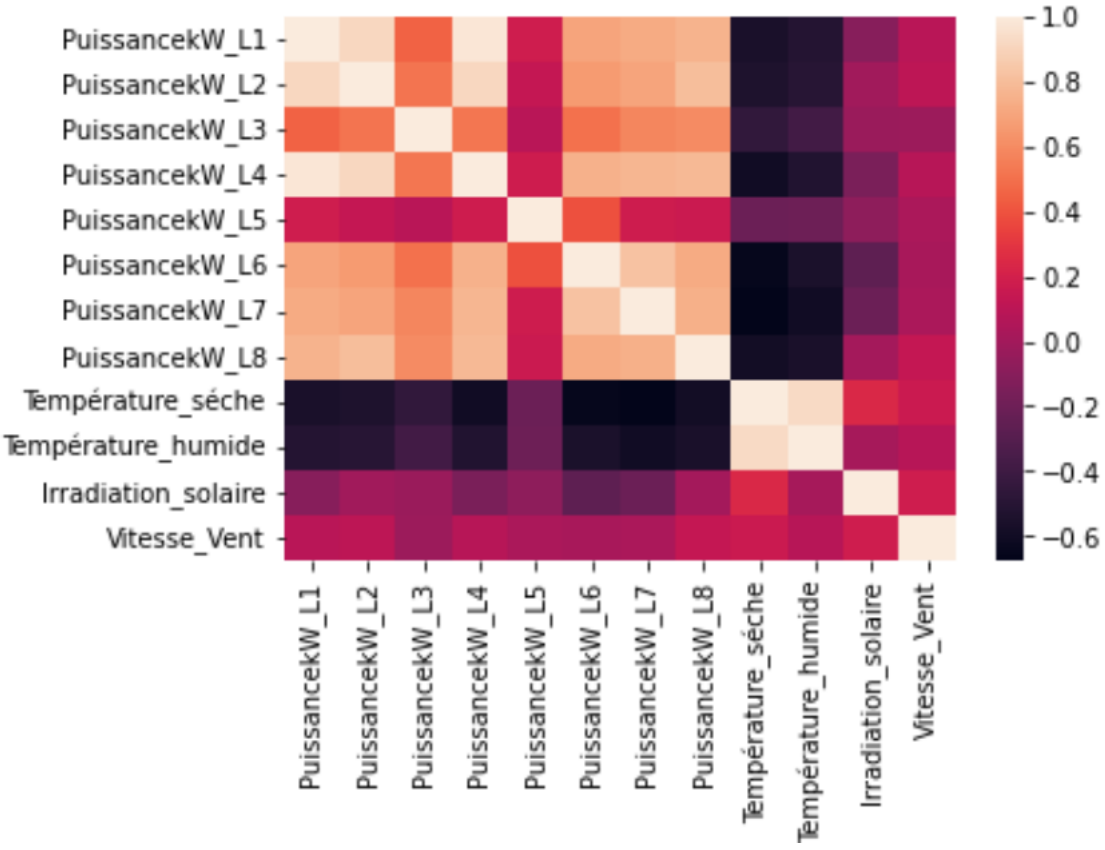



Figure 4-9 La matrice de corrélation pour l'automne 2020.

Nous remarquons que les lignes résidentielles et hybrides ont une bonne corrélation avec la température sèche et humide (R=-0.6 et -0.7). Par ailleurs, il y a une corrélation trop faible avec la vitesse du vent et l'irradiation. Cependant, la ligne 5 a une indépendance totale avec les conditions météorologiques, ce qui est expliqué par la consommation des centres de cryptomonnaie qui ont une alimentation fixe, et peu importe la saison ou les conditions climatiques.

Toutes les autres lignes résidentielles et hybrides ont une bonne corrélation avec la température sèche et humide quelle que soit la durée de temps d'un seul mois ou bien pour toutes les saisons. La figure 4-10 montre bien comment la température sèche influe sur la consommation totale de la ville tout au long de l'année 2020.

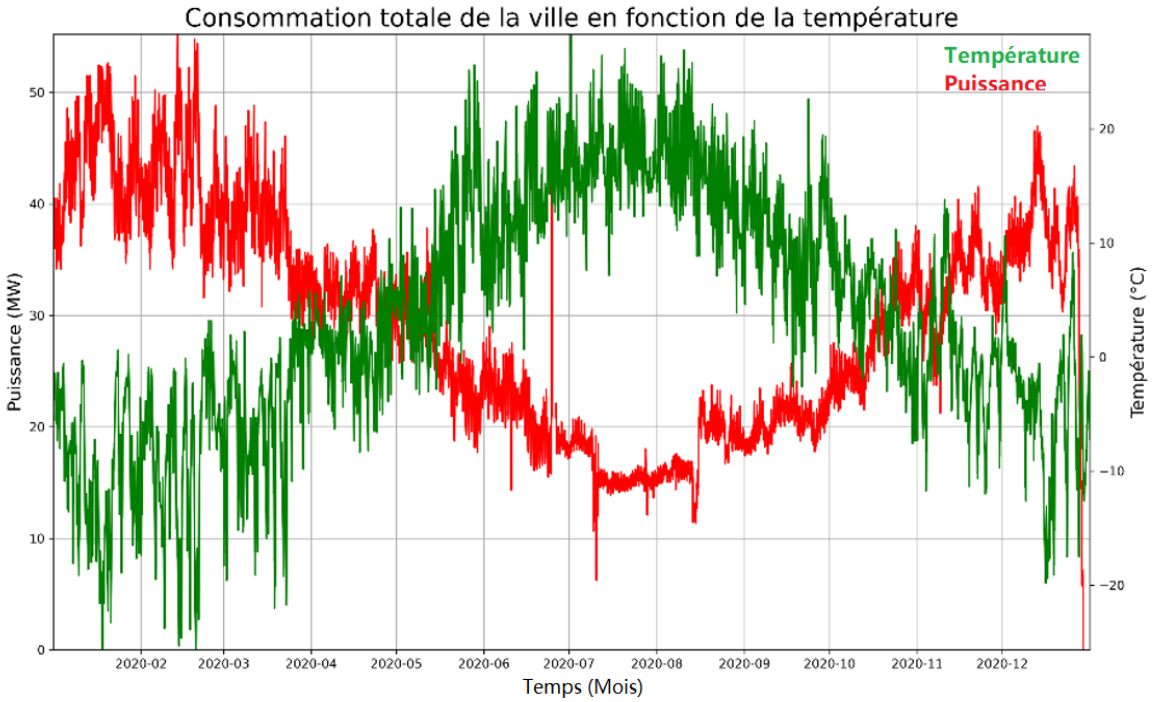



Figure 4-10 La consommation totale de la ville comparée à l'évolution de la température en 2020.

Donc, dans notre étude, on va regarder la température qui va être le facteur influenceur principal. Suivant les directives du cahier de charges, la ville a besoin d'une estimation de la consommation à court et à moyen terme pour aider à la prise de décision pour le bon fonctionnement du réseau.

Les consommateurs résidentiels, en particulier, font l'objet d'une enquête car ils partagent une grande partie de l'énergie totale consommée et ont des composantes irrégulières dans leurs courbes de consommation (lignes à caractères résidentiels et hybrides). En revanche la ligne industrielle 5 a une consommation constante durant toute l'année car la température n'a aucun effet sur elle donc elle ne va pas figurer dans notre étude. Cette ligne pourrait être utilisée par le distributeur comme moyen pour moduler la puissance lors des périodes critiques. Par exemple, en réduisant le nombre d'unités de calcul utilisées pour le minage.

## 4.4 Étude de prédictibilité

Avant de faire l'étude de la prévision des séries chronologiques, il faut tout d'abord calculer le coefficient de Hurst [46]. Ce dernier est une mesure de la mémoire dans une série chronologique et est utilisé pour classer la série comme retour à la moyenne, tendance ou marche aléatoire (Tableau 3-2).

Le coefficient de Hurst est calculé grâce à la bibliothèque Python Hurst [41]. Les données étudiées sont celles de la consommation annuelle pour toutes les lignes.

Cette implémentation est basée sur l'estimation du taux de comportement diffusif en fonction de la variance du logarithme (Log) des valeurs numériques, qui dans notre cas est la consommation annuelle.

On peut définir 'x' comme le logarithme de la consommation 'C' comme suit [41]:

$$x_t = \log(C_t) \qquad (4\text{-}1)$$

La variance pour un décalage arbitraire (notée τ) peut être exprimée comme suit :

$$\mathrm{Var}(\tau) = \langle |x_{t+\tau} - x_t|^2 \rangle \qquad (4\text{-}2)$$

Si la consommation suit un mouvement de marche aléatoire, la variance varierait linéairement avec le retard τ [46] :

$$\mathrm{Var}(\tau) \sim \tau \qquad (4\text{-}3)$$

Dans le cas où la consommation ne suit pas une marche aléatoire, la variance pour un décalage donné n'est plus proportionnelle au décalage lui-même, mais acquiert plutôt un exposant anormal. La nouvelle relation se présente comme suit [46]:

$$\mathrm{Var}(\tau) \sim \tau^{2H} \qquad (4\text{-}4)$$

Où H représente l'exposant de Hurst.

La bibliothèque de Python nous permet d'ajuster le nombre de retard. Le nombre de retard par défaut est 20. D'après [41] si le nombre de retards est trop élevé, les résultats pourraient être inexacts. Le Tableau 4-1 montre les résultats obtenus.

Tableau 4-1 Résultats du coefficient de Hurst.

| Retard | Ligne 1 | Ligne 2 | Ligne 3 | Ligne 4 | Ligne 5 | Ligne 6 | Ligne 7 | Ligne 8 |
|---|---|---|---|---|---|---|---|---|
| 10 | 0.7023 | 0.4705 | 0.4580 | 0.5255 | 0.2559 | 0.6905 | 0.5520 | 0.6702 |
| 20 | 0.6756 | 0.5163 | 0.3856 | 0.5410 | 0.2300 | 0.6828 | 0.5750 | 0.6491 |
| 40 | 0.4976 | 0.4793 | 0.4020 | 0.4073 | 0.2529 | 0.5829 | 0.4336 | 0.4956 |
| 60 | 0.3336 | 0.4078 | 0.3822 | 0.2889 | 0.2688 | 0.4712 | 0.2896 | 0.3580 |
| 80 | 0.3178 | 0.3514 | 0.3007 | 0.2656 | 0.2780 | 0.3938 | 0.2800 | 0.3358 |
| 100 | 0.2350 | 0.2684 | 0.1935 | 0.2133 | 0.2767 | 0.2650 | 0.2023 | 0.2637 |
| 200 | 0.1944 | 0.2412 | 0.1566 | 0.1720 | 0.2294 | 0.2120 | 0.1741 | 0.2282 |

Pour toutes les séries temporelles des lignes de consommation, lorsque nous augmentons le décalage, la valeur de l'exposant diminue vers 0, ce qui indique une série de retour à la moyenne (anti-persistante). Il convient de faire le calcul et l'analyse du coefficient Hurst avant de se lancer dans des stratégies de prévision. Les résultats trouvés sont logiques avec nos attentes initiales, ce sont des données avec des valeurs élevées qui sont suivies d'une valeur faible et vice-versa. Cela peut s'expliquer par le changement journalier (jour/nuit), et l'influence des facteurs externes sur la variation des données.

## 4.5 Prévision de la demande en puissance à moyen terme

### *4.5.1 Régression linéaire*

Vu que la ville de Baie-Comeau cherche à optimiser son fonctionnement pendant l'hiver, notre étude est consacrée sur les deux premiers mois de l'année 2020, à savoir Janvier et Février. En effet, comme illustré dans la Figure 4-10, c'est durant cette période de l'année où il y a les pics de consommation et on observe des températures basses extrêmes. En outre, au début de cette saison hivernale, il y a un bon coefficient de corrélation à l'égard de toute ligne qu'on veut étudier.

La Figure 4-11 montre l'équation de la régression linéaire entre les deux variables température et consommation. Nous avons obtenu ces résultats en entraînant le modèle sur un mois; l'historique d'un mois de température et de puissance pour la ligne 1 pour le mois de Janvier.

Pour entraîner le modèle de la régression, il faut tout d'abord extraire les équations linéaires, cubique et quadratique et de choisir ensuite quelle équation est la plus convenable avec les données de la température et la puissance en MW pour la période d'analyse.

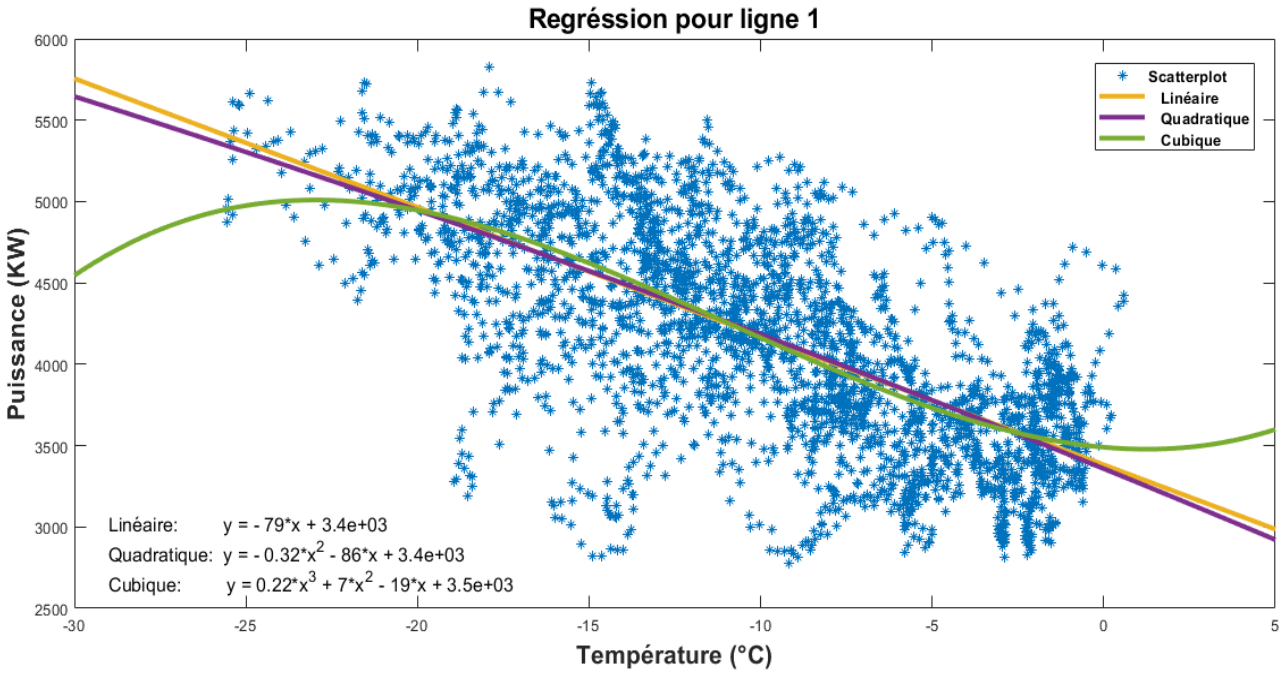



Figure 4-11 L'équation de régression avec scatter plot pour la ligne 1.

Pour faire la prévision de la consommation d'un mois, on utilise les équations extraites de la régression soit linéaire, quadratique ou cubique. Dans notre cas, pour faire la prévision, on utilise l'équation linéaire car elle donne de meilleurs résultats par rapport aux autres équations polynomiales.

Le principe de fonctionnement du modèle est d'introduire comme entrée les données concernant les températures du même mois d'entraînement (p. ex., janvier 2020) dans l'équation choisie f(u). Ceci afin d'avoir un mois de la puissance estimée. Cette dernière est soustraite à la puissance mesurée ou réelle. La représentation dans la figure ci-dessous illustre cette méthodologie.

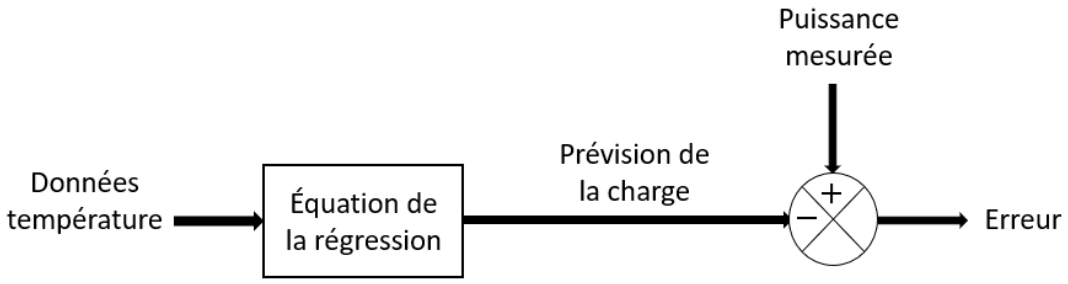



Figure 4-12 La méthodologie de la régression linéaire.

Les résultats obtenus du modèle Simulink sont ceux des lignes de distribution 1, 4 et 7. Ils sont présentés ci-dessous.

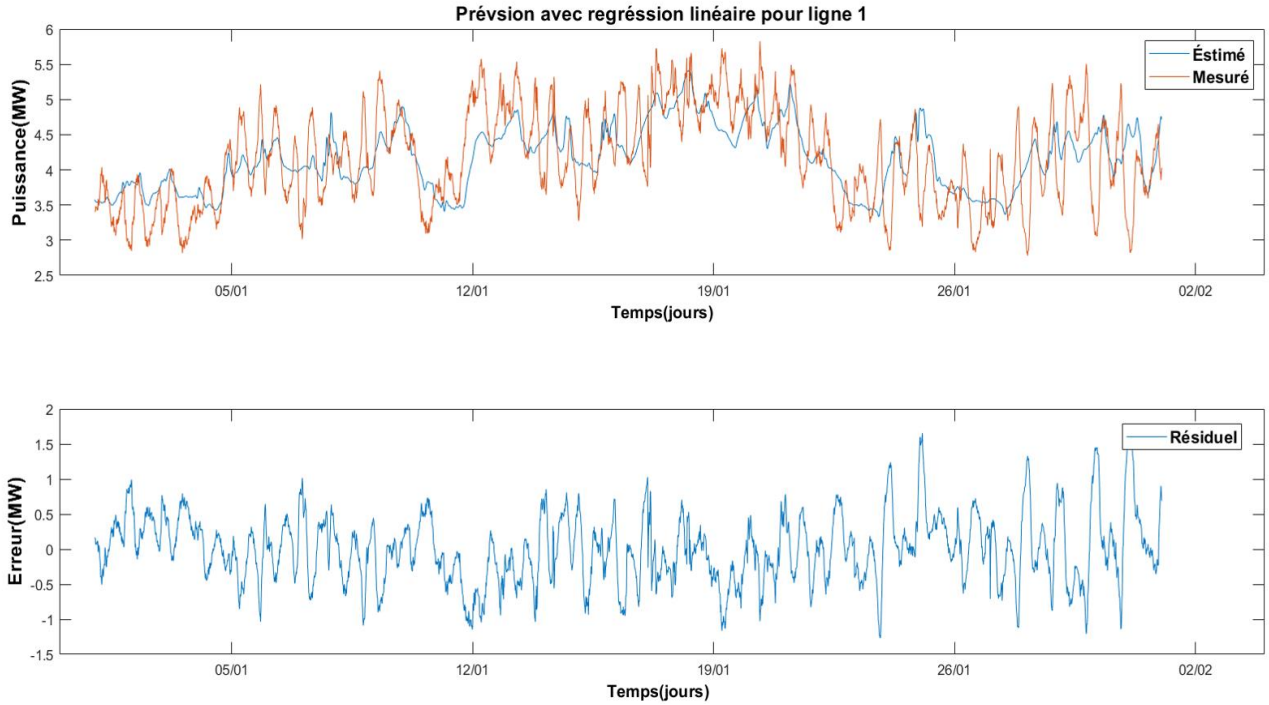



Figure 4-13 Résultats de prévision avec le modèle de régression linéaire pour la ligne 1.

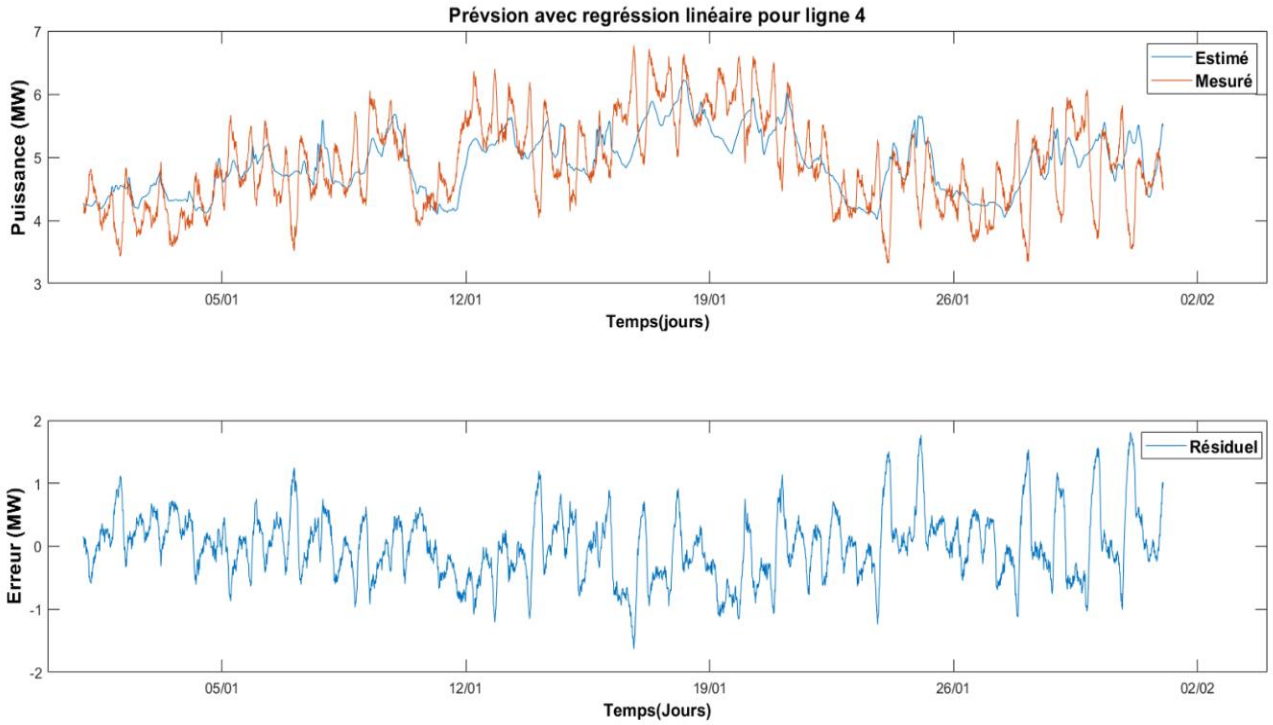



Figure 4-14 Résultats de prévision avec modèle de la régression linéaire pour la ligne 4.

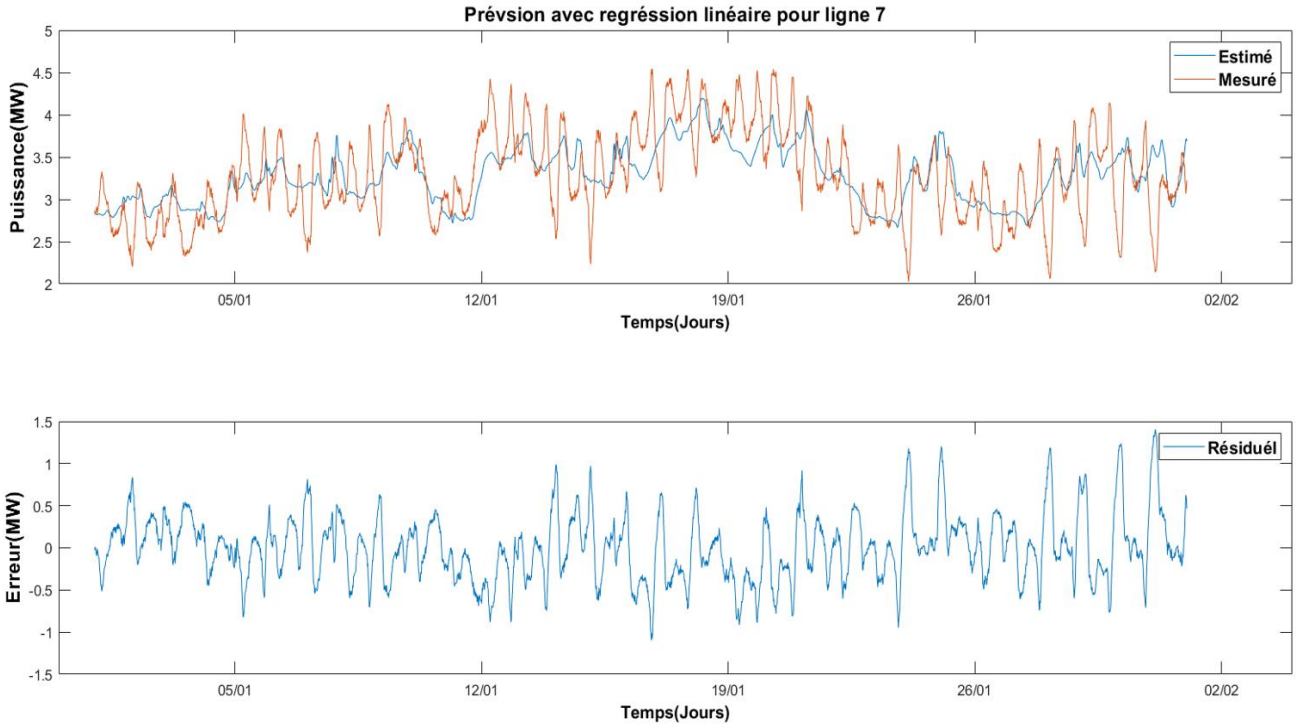



Figure 4-15 Résultats de prévision avec modèle de la régression linéaire pour la ligne 7.

Les résultats avec la régression linéaire montrent une prévision qui suit la moyenne de ce qui est enregistré réellement. La régression linéaire n'a pas pu détecter les pics de consommation pour toutes les lignes résidentielles et hybrides, l'erreur MAPE est entre 30% et 40% et ceci est une prévision raisonnable, mais assez précise car ça donne juste une idée à quoi va ressembler la consommation dans les semaines à venir. Si la prévision de la météo est fiable, elle peut indiquer par exemple quand la consommation va dépasser un seuil donné par exemple 3.5 MW pour la ligne 7.

La section suivante porte sur les résultats obtenus avec les réseaux de neurones artificiels ensuite nous on va discuter les résultats et les comparer avec ceux de la régression.

### *4.5.2 Les réseaux de neurones artificiels*

Comme déjà cité précédemment, les ANN sont une procédure systématique pour définir quantitativement les demandes en puissance futures. Alors, un modèle de prévision de demande en puissance est développé à l'aide d'un réseau neuronal multicouche avec un algorithme d'apprentissage de rétropropagation modifié de manière appropriée. Le modèle peut être créé pour produire une prévision à moyen terme de la demande en puissance. À cet effet, un outil d'ajustement neuronal est utilisé à partir de la boîte à outils de réseau de neurones du MATLAB (*NNtool*).

La méthode d'entraînement est effectuée de manière supervisée, c'est-à-dire que pour chaque vecteur d'entrée, la sortie souhaitée est donnée. Les poids sont ajustés pour minimiser une fonction d'erreur qui mesure l'écart entre la sortie souhaitée et la sortie calculée par le réseau.

Cette méthode d'entraînement est appliquée avec la rétropropagation du gradient conjugué mis à l'échelle (*trainscg* : *Scaled conjugate gradient backpropagation*). *Trainscg* est une fonction d'apprentissage du réseau qui met à jour les valeurs de pondération et de biais selon la méthode du gradient conjugué mis à l'échelle. On peut aussi créer un réseau standard qui utilise *trainscg* avec *feedforwardnet* ou *cascadeforwardnet*. Dans notre cas nous avons utilisé un réseau *feedforwardnet* car il est efficace avec une structure simple. Les réseaux *feedforward* sont constitués d'une série de couches. La première couche a une connexion depuis l'entrée du réseau. Chaque couche suivante a une connexion à partir de la couche précédente. Le réseau contient une ou plusieurs couches cachées, et la couche finale produit la sortie du réseau.

La méthodologie pour la prévision avec les ANN est présentée dans la figure 4-16. Une étape de prétraitement des données est nécessaire. Elle consiste en effet à préparer les données pour identifier les entrées et les sorties du modèle. Les données d'entrée sont l'historique de la température sèche et la demande en puissance de consommation qui vont être utilisées par la suite par le modèle afin de donner une prévision de la demande en puissance comme sortie. Cette prévision est comparée avec la demande en puissance réelle pour obtenir l'erreur.

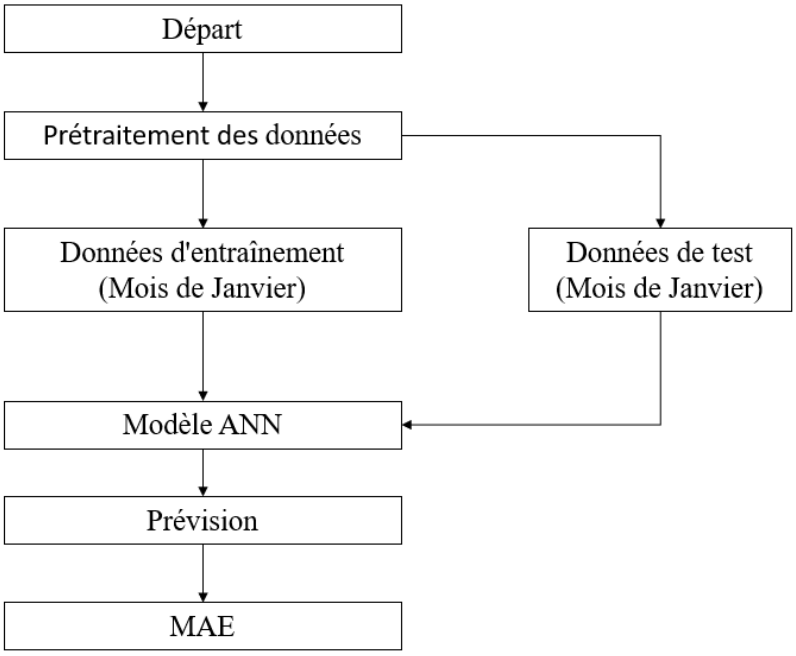



Figure 4-16 Méthodologie pour l'entraînement et test du modèle avec réseau de neurones.

Concernant le modèle *feedforward*, nous commençons premièrement par faire entrer les données de la température du mois de Janvier de l'année 2020, ainsi que les données de demande en puissance en MW de la ligne qu'on veut prédire, de la même période. Ces données sont chargées sous forme d'une matrice X de [195*2880] où 195 représente le nombre d'entrées (*input*) et 2880 représente le nombre de données en 30 jours (96 prélèvements journaliers en 30 jours).

Deuxièmement le nombre de neurones de la couche cachée dépend de la non-linéarité du problème [22]. D'après les essais que nous avons réalisés, 5 couches cachées se sont avérées suffisantes pour le problème actuel. En effet, cela a permis de donner les meilleurs résultats qui ne présentent pas un grand écart par rapport à la courbe de demande en puissance réelle.

Sur MATLAB la fonction *Fitnet (net=fitnet(5,'trainscg'))* a été utilisée pour définir le nombre de couches cachées.

Enfin, dans le but d'avoir le vecteur de sortie qui représente la prévision de la demande en puissance, il faut entraîner le système. La couche de sortie possède un seul neurone qui représente la demande mensuelle en puissance avec la matrice Y [1*2880]. La simulation a arrêté après 57 itérations, avec 1 intervalle (*epoch*). Généralement, l'erreur diminue après plusieurs époques d'apprentissage, mais peut commencer à augmenter sur l'ensemble de données de validation lorsque le réseau commence à s'adapter avec les données d'apprentissage. Dans la configuration par défaut, la formation s'arrête après six augmentations consécutives de l'erreur de validation, et les meilleures performances sont tirées de l'époque avec l'erreur de validation la plus faible.

Ces valeurs de sortie sont comparées graphiquement avec les valeurs réelles enregistrées, et pour voir l'erreur on calcule MAPE.

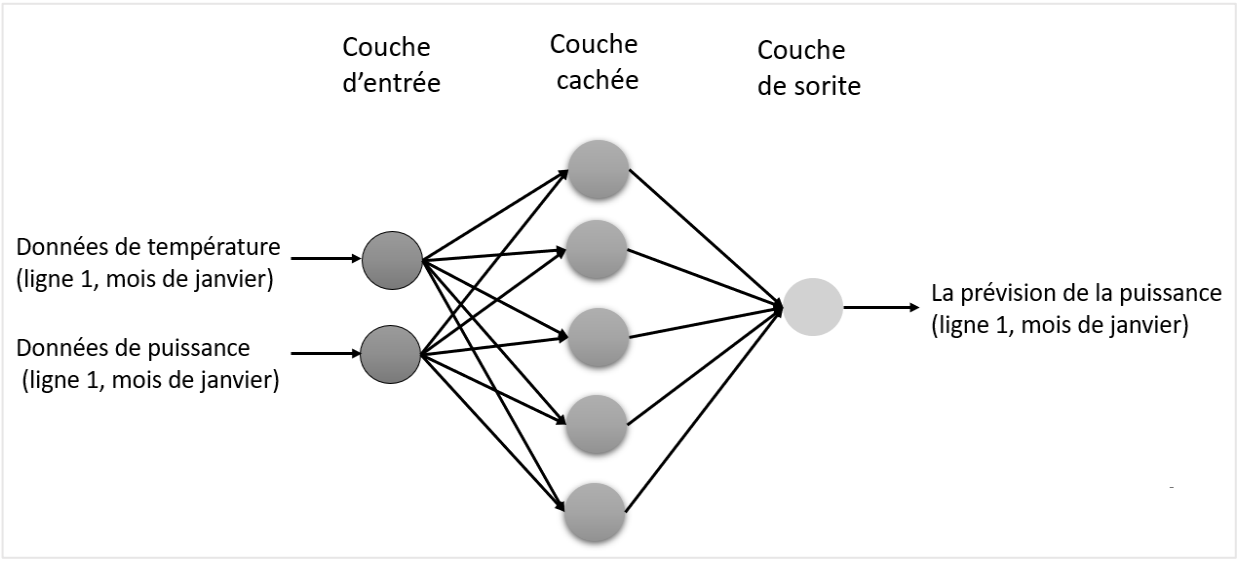



Figure 4-17 Caractéristique du modèle de réseaux de neurones.

Les résultats pour la prévision du mois de janvier 2020 pour la ligne 1 sont représentés dans la figure 4-18. Par ailleurs, les résultats des prévisions des autres lignes ont été obtenus en suivant la même procédure, mais avec des données d'entraînements et des tests propres à chaque ligne. On peut voir aussi les résultats des lignes 4 et 7 dans les figures 4-19 et 4-20 :

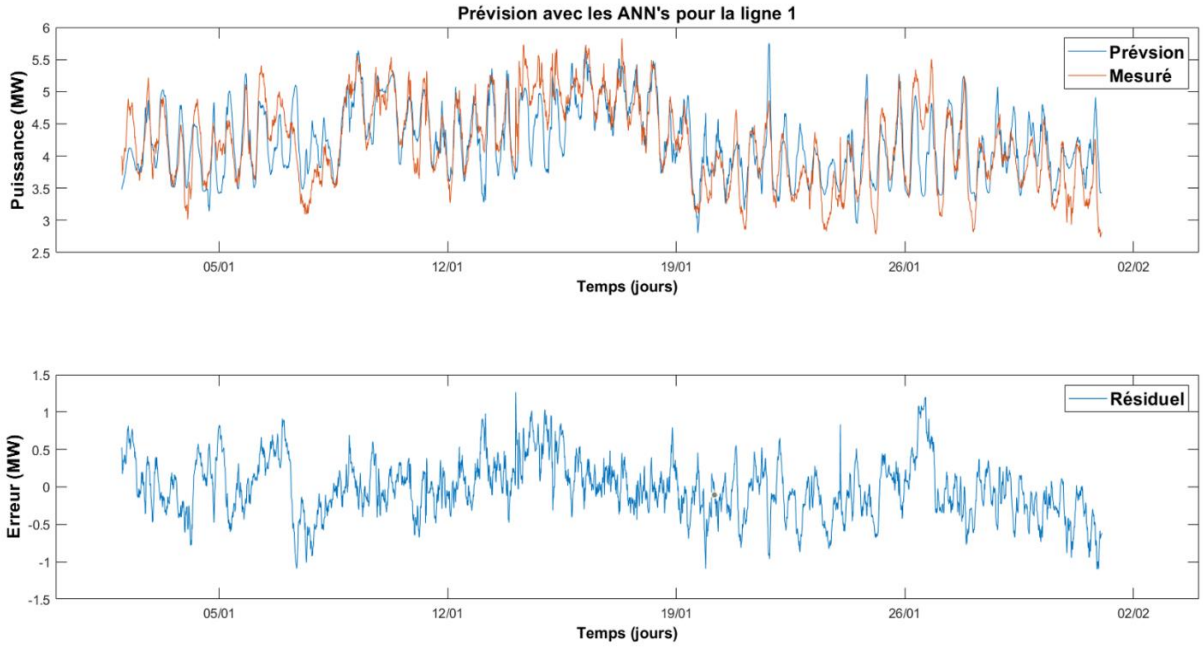



Figure 4-18 Résultats de la prévision avec les réseaux de neurones artificiels pour la ligne 1.

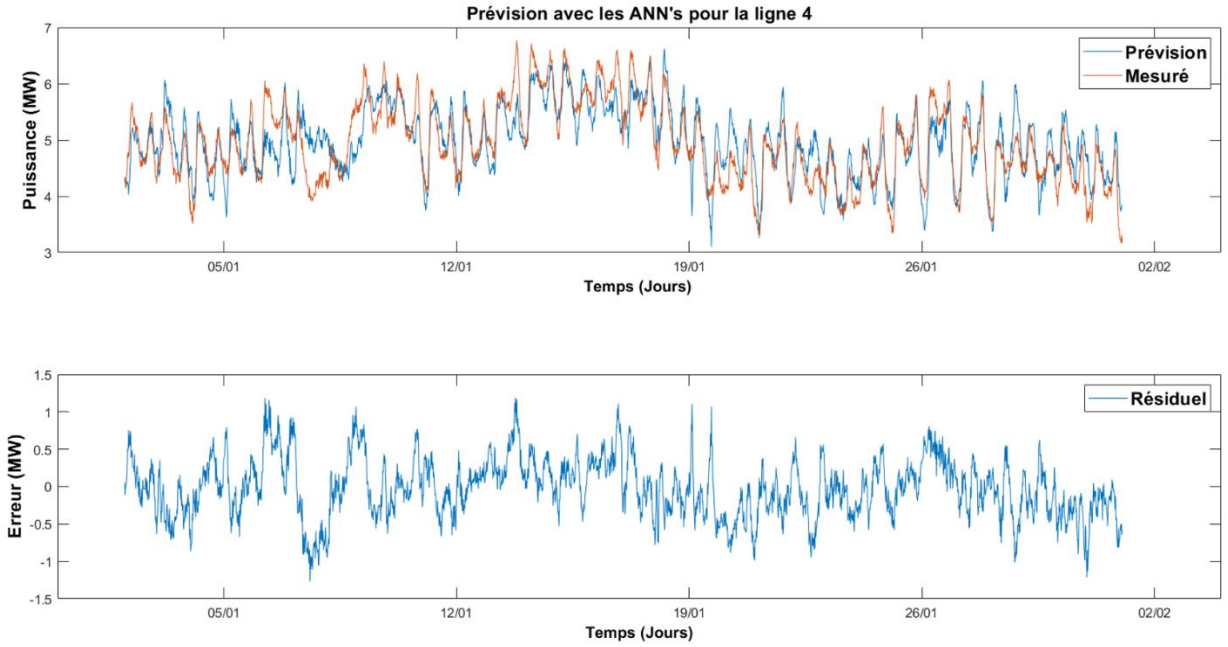



Figure 4-19 Résultats de la prévision avec les réseaux de neurones artificiels pour la ligne 4.

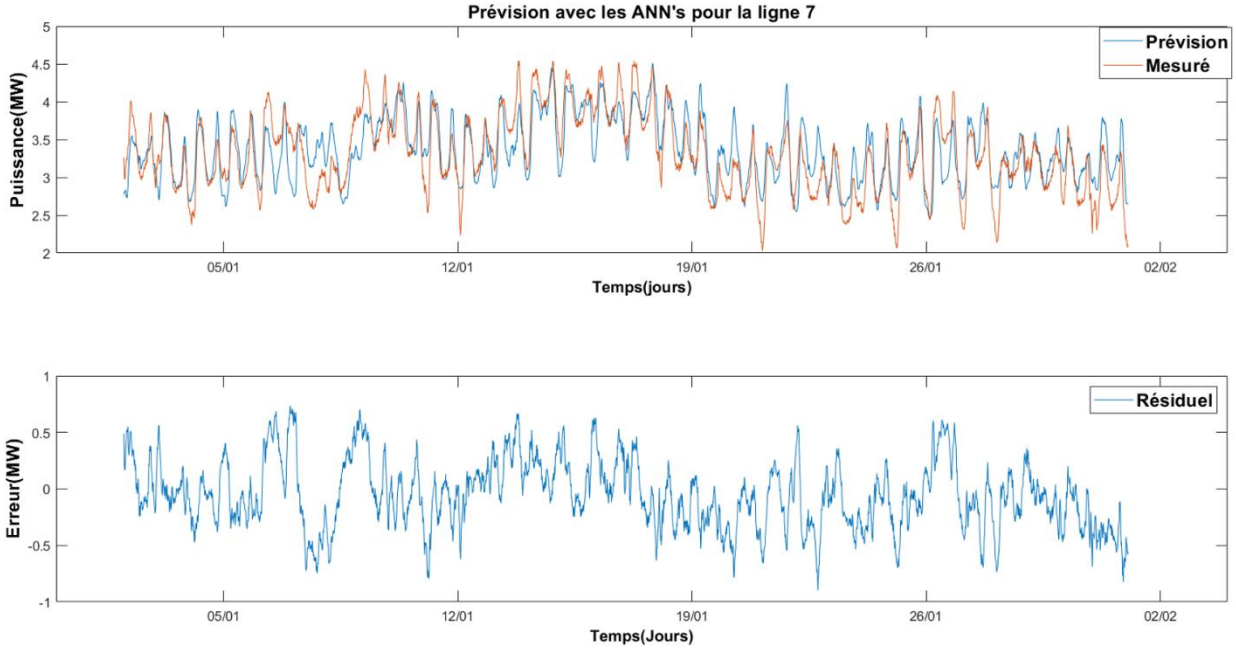



Figure 4-20 Résultats de la prévision avec les réseaux de neurones artificiels pour la ligne 7.

On voit que la prévision suit la trajectoire de la moyenne de la consommation enregistrée. Par ailleurs, on remarque que la prévision peut détecter les pics de consommation tout au long du mois contrairement à la régression linéaire. L'erreur MAPE a été aussi diminuée, ce qui montre l'efficacité des réseaux de neurones artificiels. Le tableau 4-2 montre tous les résultats d'erreur MAPE des huit lignes pour le mois de janvier 2020. Pour toutes les lignes, les réseaux de neurones donnent une erreur inférieure à celle de la régression, sauf la ligne 2 et la ligne 6, qui ont une erreur presque égale ou supérieure à la régression linéaire. Cela s'explique par la catégorie de consommateurs de ces deux lignes qui sont constituées d'un mélange de résidentiel, commercial et autre. Contrairement aux autres lignes qui n'ont qu'un seul type de consommateurs dominants.

Tableau 4-2 Résultats d'erreur comparative.

| Les lignes de distribution | Erreur MAPE (%) mois de Janvier | |
|---|---|---|
| | Régression linéaire | Réseaux de neurones |
| Ligne 1 | 38.4 | 29.8 |
| Ligne 2 | 32.2 | 34.5 |
| Ligne 3 | 60.9 | 33.5 |
| Ligne 4 | 40.5 | 31.6 |
| Ligne 5 | 08.0 | 06.1 |
| Ligne 6 | 23.9 | 22.1 |
| Ligne 7 | 30.4 | 24.7 |
| Ligne 8 | 48.2 | 39.7 |

### *4.5.3 Discussion et conclusion*

A moyen terme, nous avons trouvé que les réseaux de neurones ont montré une meilleure efficacité et précision pour la prévision par rapport à la régression linéaire tout au long du mois de janvier. Nous avons trouvé les mêmes résultats pour le mois de février car ils ont le même coefficient de corrélation et des caractéristiques similaires.

Ces résultats ont été atteints après plusieurs essais par ce qu'il est nécessaire de choisir avec soin certains paramètres pour avoir le modèle le plus performant. Par exemple le choix du nombre optimal de couches cachées et la quantité de données à utiliser pour l'entraînement. Par ailleurs, les réseaux de neurones ont bien fonctionné grâce au taux de données historiques disponibles. Et ont montré un bon résultat quand les lignes ne sont pas hybrides.

## 4.6 Prévision de la demande en puissance à court terme

### *4.6.1 Modèle prévision dynamique*

Comme nous l'avons vu précédemment, le modèle de réseaux de neurones donne de meilleurs résultats en termes de prévision. Nous pouvons donc bien l'adapter pour faire une prévision à court terme dynamique plus précise avec MATLAB.

Le principe de fonctionnement du nouveau modèle est de faire une prévision d'une heure, et quand cette dernière est écoulée, le modèle va l'utiliser comme donnée d'entraînement en plus des données déjà mises comme entrées. Ensuite, le modèle va générer la prochaine nouvelle heure suivante, et ainsi de suite qu'on avance dans le temps jusqu'à faire 24 heures consécutives.

Ainsi, le système se met automatiquement à jour toutes les heures avec de nouvelles données d'entraînement donnant plus de précision au modèle. Ce type de procédure est appelé prévision récursive en plusieurs étapes, comme illustré dans la figure suivante :

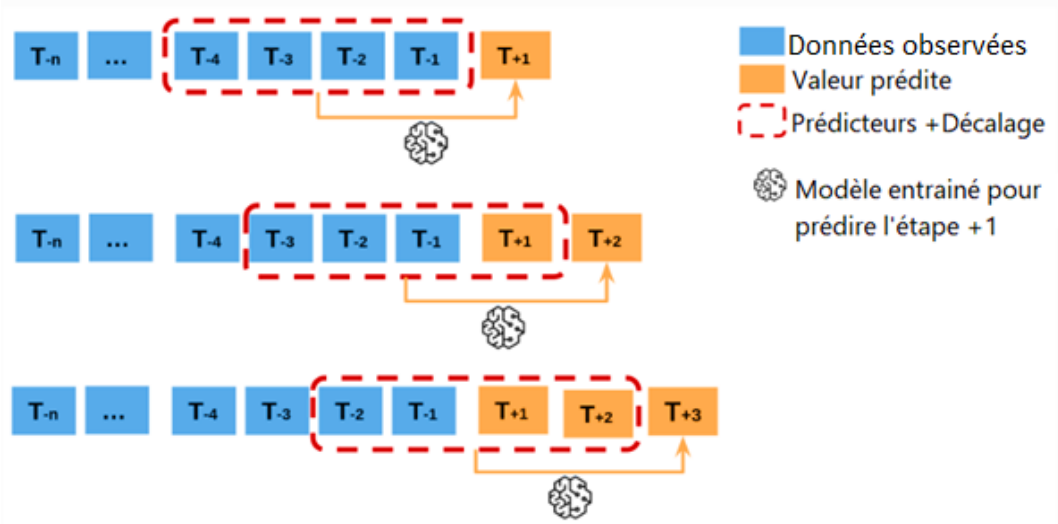



Figure 4-21 Prévision récursive en plusieurs étapes [47].

Le modèle ANN est identique à celui du moyen terme (4.5.2). Il s'agit d'un réseau *feedforwardnet* ajusté avec un entraînement supervisé avec la fonction *trainscg*. Cependant, ces caractéristiques sont différentes car les données d'entrée et de simulation sont différentes. Le réseau contient 28 entrées (*input*) qui sont le vecteur de la température et le vecteur historique de consommation, ces deux derniers correspondent à un mois de données d'entraînement. Il contient 5 couches cachées et 1 neurone de sortie (*output*) qui est la prévision de consommation [Y_est]. L'entraînement s'arrête après 57 itérations, avec 1 intervalle (*epoch*).

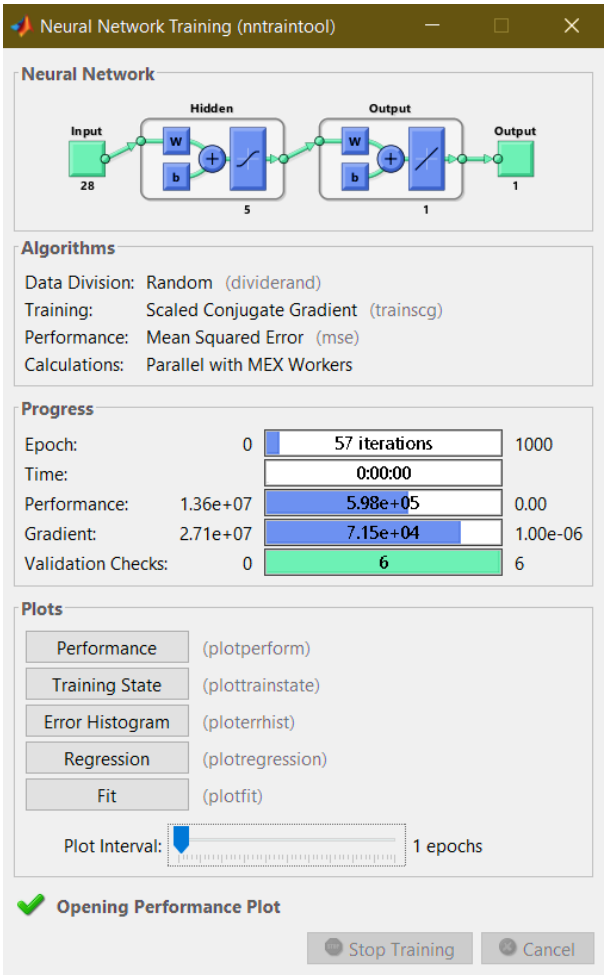



Figure 4-22 Les caractéristiques du modèle réseaux de neurones artificiels.

Les figures suivantes montrent les résultats de prévision des neuf premières heures de décalage. Les figures concernant la 10ème heure jusqu'à la 24ème heure sont illustrées dans l'annexe C.


Figure 4-23 Prévisions avec ANN pour la ligne 1 des trois premières heures de décalages.

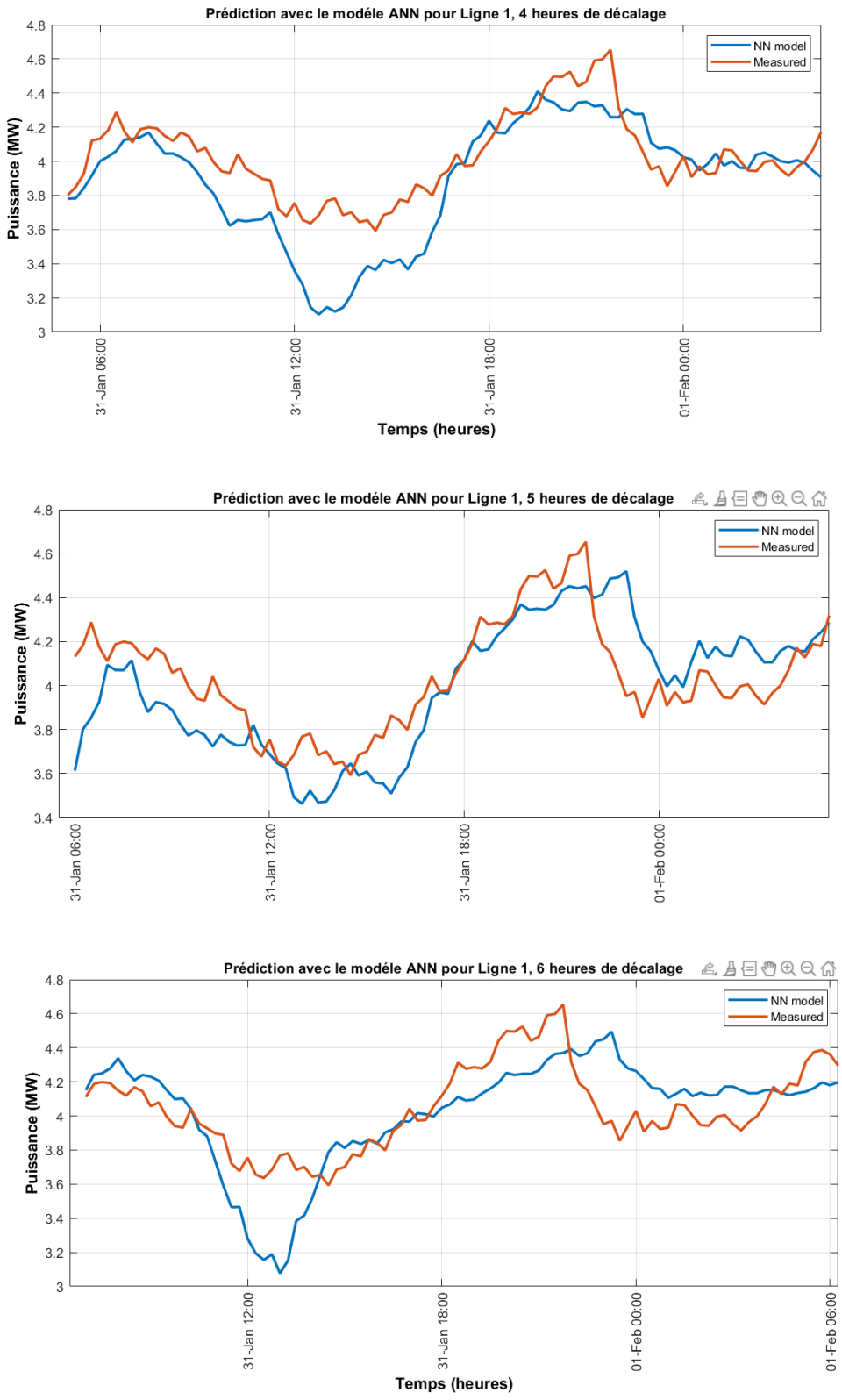



Figure 4-24 Prévisions avec ANN pour la ligne 1 des trois heures suivantes de décalages (4 à 6 heures).

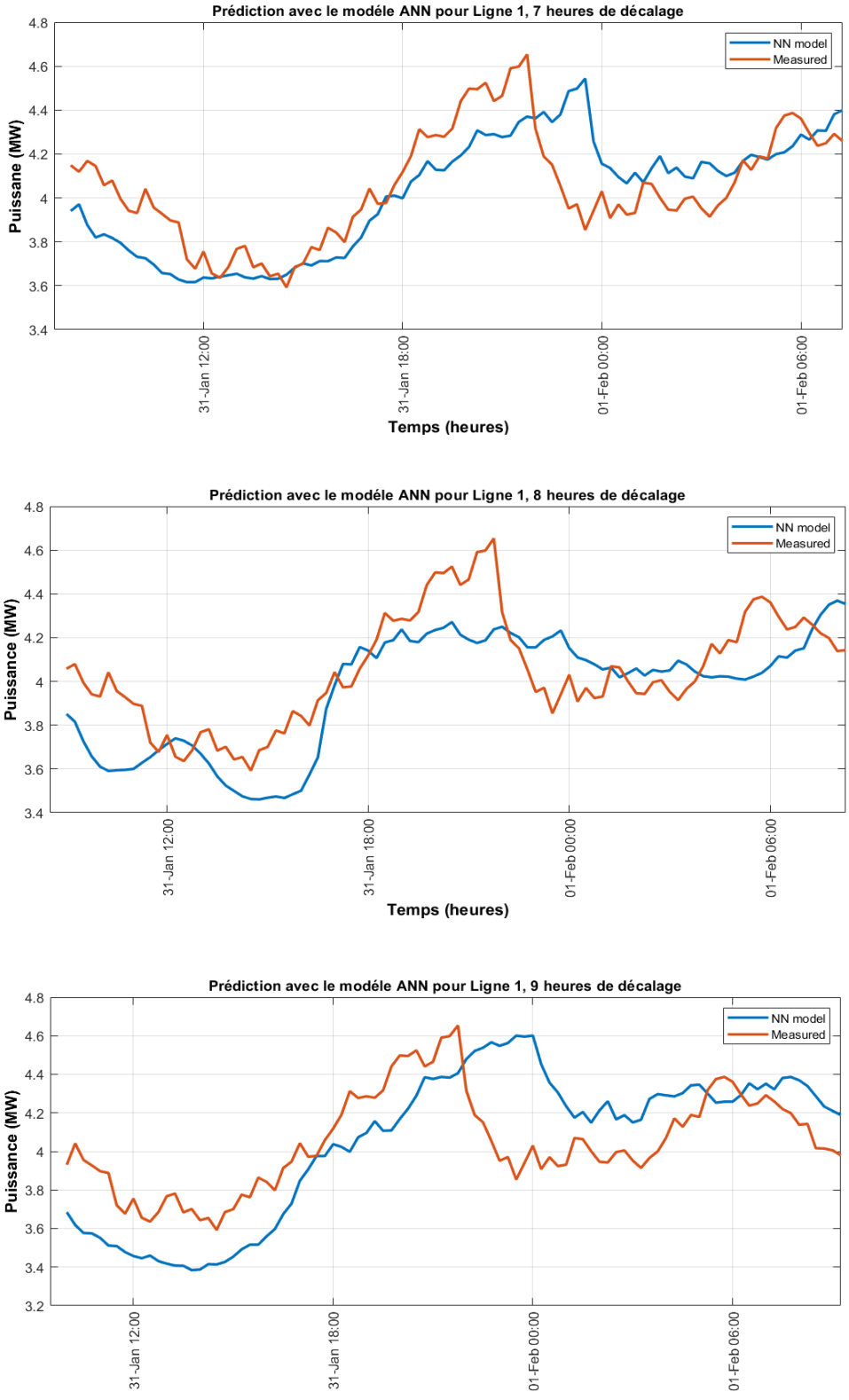



Figure 4-25 Prévision avec ANN pour la ligne 1 des trois heures suivantes de décalages (7 à 9 heures).

Le tableaux 4-3 résume tous les résultats pour toutes les lignes :

Tableau 4-3 Résultats erreur MAPE pour toutes les lignes de distribution.

| | Erreur MAPE (%) pour chaque heure | | | | | | |
|---|---|---|---|---|---|---|---|
| Ligne / Heure | Ligne 1 | Ligne 2 | Ligne 3 | Ligne 4 | Ligne 6 | Ligne 7 | Ligne 8 |
| 1 | 5.01 | 5.09 | 7.63 | 4.97 | 3.53 | 4.80 | 3.43 |
| 2 | 4.94 | 3.88 | 7.96 | 3.50 | 4.18 | 4.88 | 4.63 |
| 3 | 4.21 | 3.10 | 7.94 | 3.6 | 4.09 | 4.78 | 4.67 |
| 4 | 5.00 | 3.92 | 9.37 | 4.1 | 3.90 | 3.25 | 3.67 |
| 5 | 4.30 | 4.73 | 9.54 | 4.04 | 3.62 | 4.30 | 2.72 |
| 6 | 4.68 | 4.04 | 12.46 | 4.06 | 4.59 | 5.02 | 3.85 |
| 7 | 3.83 | 4.73 | 12.66 | 5.83 | 4.76 | 3.84 | 2.42 |
| 8 | 4.43 | 5.81 | 12.43 | 4.96 | 6.29 | 4.29 | 3.06 |
| 9 | 6.10 | 4.40 | 11.90 | 4.95 | 6.26 | 6.67 | 4.92 |
| 10 | 7.28 | 4.62 | 15.45 | 5.04 | 6.77 | 4.12 | 4.39 |
| 11 | 6.22 | 5.40 | 12.52 | 4.86 | 5.32 | 4.86 | 4.88 |
| 12 | 7.06 | 5.45 | 15.51 | 5.36 | 5.93 | 4.34 | 3.81 |
| 13 | 5.45 | 4.73 | 16.01 | 5.17 | 5.97 | 4.12 | 3.19 |
| 14 | 5.47 | 4.33 | 14.29 | 5.29 | 6.06 | 4.71 | 4.26 |
| 15 | 4.75 | 4.81 | 15.90 | 4.35 | 6.12 | 4.72 | 3.57 |
| 16 | 5.90 | 4.77 | 15.84 | 5.84 | 5.79 | 4.77 | 3.01 |
| 17 | 5.05 | 4.59 | 15.93 | 7.32 | 5.69 | 5.08 | 3.78 |
| 18 | 6.05 | 5.66 | 19.36 | 5.80 | 6.99 | 5.09 | 5.61 |
| 19 | 5.93 | 4.35 | 18.89 | 4.97 | 5.62 | 4.38 | 4.55 |
| 20 | 4.17 | 4.54 | 17.30 | 6.31 | 6.86 | 5.80 | 4.99 |
| 21 | 5.65 | 4.19 | 17.08 | 5.93 | 7.05 | 4.50 | 5.44 |
| 22 | 4.77 | 4.78 | 15.42 | 5.20 | 6.16 | 5.19 | 3.95 |
| 23 | 5.23 | 5.07 | 16.80 | 6.53 | 5.65 | 5.32 | 4.64 |
| 24 | 4.9 | 5.47 | 17.05 | 4.97 | 7.98 | 4.13 | 4.87 |

Les résultats obtenus pour toutes les lignes ont une erreur inférieure à 10%, ce qui représente des prévisions précises, et comme on voit dès les premières huit heures, la prévision donne presque les mêmes résultats enregistrés par le réseau. En détectant aussi les pics de consommation. La ligne 8 est celle qui donne une erreur MAPE plus faible que les autres lignes, cela se justifie par la quantité de données disponibles pour l'entraînement car cette ligne représente 18% de la consommation totale de la ville, comme le montre la Figure 4-5. De plus, cette ligne et les autres lignes hybrides (lignes 6, 2 et 4) donnent presque les mêmes résultats, puisqu'elles ont une bonne corrélation avec la température qui aide le modèle à bien fonctionner, contrairement à la ligne 3, qui a une erreur MAPE supérieure à 10% en raison de sa catégorie de consommateur principalement industrielle, qui n'est pas affectée par la température.

### *4.6.2 Prévision de la demande en puissance avec les bibliothèques Python*

Pour exploiter encore plus notre base de données, nous avons réalisé une étude en utilisant des bibliothèques Python consacrées spécialement pour la prévision des séries chronologiques. Il existe plusieurs bibliothèques créées pour la prévision, chacune avec ses propres méthodologies. Nous avons opté premièrement pour *Sklearn* [48] qui est dédiée pour la régression linéaire et deuxièmement pour la bibliothèque *Tensorflow* [49] qui est dédiée pour les réseaux de neurones artificiels. Vu que ces bibliothèques sont déjà bâties et programmées, nous ne pouvons pas effectuer des modifications. Nous pouvons donc charger les données d'entrée, installer les bibliothèques et choisir les périodes d'entraînement. Les bibliothèques importées pour la gestion de données et les représentations graphiques sont (*numpy, matplotlib* et *pandas*).

Nous avons alors effectué une étude de prévision à court terme pour prédire 24 heures, pour voir quels sont les résultats que nous pouvons obtenir.

Pour des quantités différentes de données utilisées pour l'entraînement (1 semaine, 15 jours, 1 mois, 2 mois, 3 mois), nous avons fait la prévision pour la régression linéaire et les réseaux de neurones pour trois journées différentes. Cela est illustré dans les figures qui suivent. Ensuite, le tableau 4-4 montre les résultats d'erreur MAE, et pour toutes les autres lignes voire l'annexe A.

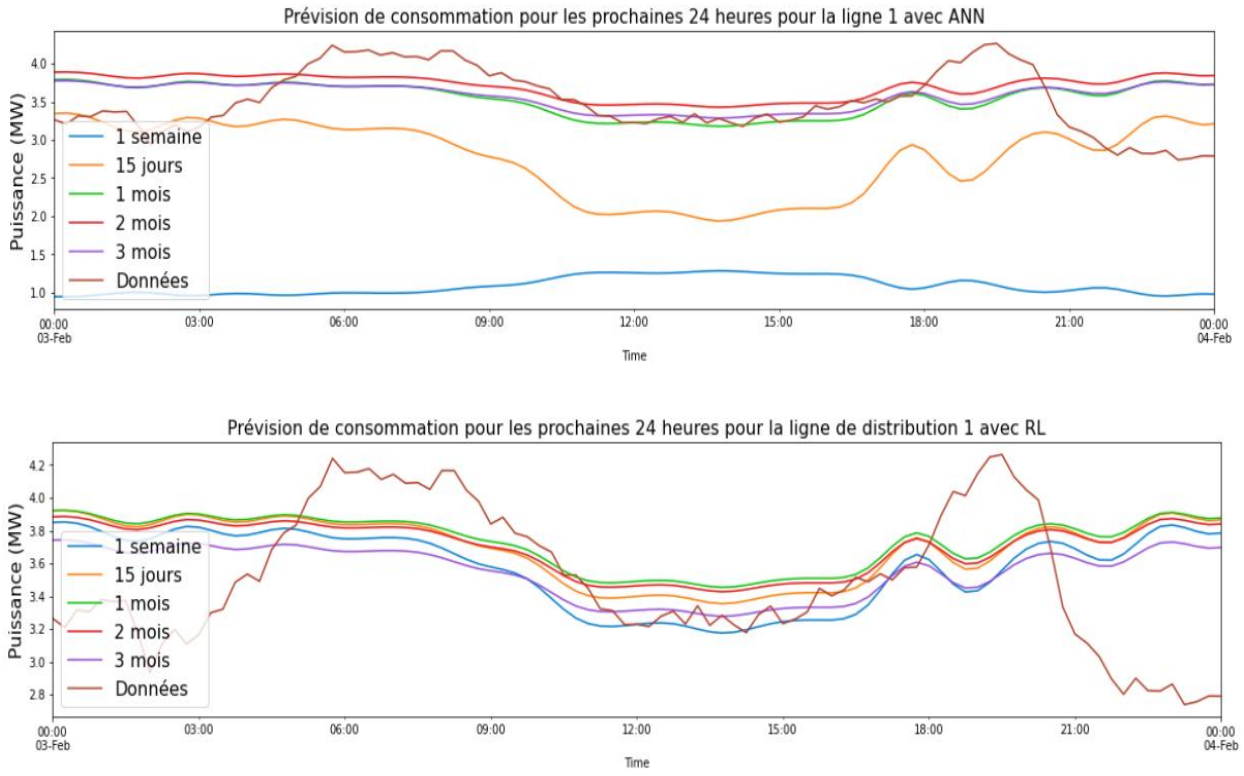



Figure 4-26 Résultats de prévision avec Python au 3 Février 2020 avec ANN et RL.

Tableau 4-4 Erreur MAE pour 24 heures (03 Février 2020).

| | | MAPE (%) pour la Ligne 1 | | |
|---|---|---|---|---|
| Modèle | Quantité de données | 1 FEV (Samedi) | 2 FEV (Dimanche) | 3 FEV (Lundi) |
| ANN | 1 Semaine | 26.57 | 11.67 | 30.28 |
| | 15 Jours | 14.71 | 10.69 | 12.27 |
| | 1 Mois | 5.55 | 8.64 | 11.78 |
| | 2 Mois | 5.08 | 10.01 | 11.47 |
| | 3 Mois | 4.85 | 8.45 | 10.36 |
| RL | 1 Semaine | 5.92 | 9.12 | 11.09 |
| | 15 Jours | 6.01 | 10.35 | 11.37 |
| | 1 Mois | 5.44 | 10.58 | 11.85 |
| | 2 Mois | 5.08 | 10.04 | 11.44 |
| | 3 Mois | 4.81 | 8.38 | 10.32 |

Les études faites avec les bibliothèques Python donnent des résultats de prévision assez précis avec des erreurs MAPE de 5 à 30 % dépendant des quantités de données utilisées pour l'entraînement. Les courbes suivent la moyenne de consommation mais sans pouvoir détecter les pics, que ce soit pour la RL ou les ANN. Ces bibliothèques demeurent une manière d'avoir une idée générale sur le fonctionnement et la gestion des données avec Python. Les résultats obtenus avec certaines librairies ne sont pas entièrement crédibles, parce que souvent on n'a pas la possibilité de les modifier ou de les personnaliser suffisamment vu que ce sont déjà des algorithmes préprogrammés installables. MATLAB, quant à lui, nous permet d'écrire notre propre code ainsi que de créer et gérer notre propre modèle de prévision de la demande en puissance. Néanmoins, il faut citer que les

bibliothèques de Python constituent un excellent outil pour gérer un grand nombre de base de données et facilite les représentations graphiques grâce aux bibliothèques consacrées disponibles et ouvertes sans frais.

## 4.7 Conclusion

Dans ce chapitre nous avons montré comment les bases de données brutes acquises par la ville ont été collectées, traitées et analysées. La disponibilité de ces bases de données météorologiques et de demande de puissance nous a permis de développer des modèles de prévision avec l'utilisation des deux logiciels MATLAB et Python.

Les lignes de distribution à caractères résidentiels et hybrides (résidentiels et commerciaux) ont été notre cible de recherche. Nous avons trouvé que les réseaux de neurones artificiels avec algorithme supervisé *feedforwardnet* est celui qui donne les résultats les plus précis soit à moyen terme ou dynamiquement (heure par heure). Ces résultats montrent les avantages et la performance de l'intelligence artificielle dans le domaine de la prévision des séries temporelles.

# Chapitre 5 - Conclusion

Dans ce projet de recherche, différents modèles de prévision de la demande en puissance ont été développés en se basant sur les algorithmes de régression et ceux des réseaux de neurones. Afin d'atteindre l'objectif de recherche qui a été l'anticipation de la demande de puissance du réseau de distribution de la ville de Baie-Comeau, il fallait faire face aux problèmes liés aux fonctionnements des réseaux lors des périodes de froid extrême.

Pour un fonctionnement optimal des modèles proposés de prédictibilité de données et de corrélation, nous avons étudié quels sont les types de données nécessaires à introduire comme entrées. Nous avons trouvé que la température est le seul facteur météorologique qui a une influence sur la consommation pour les lignes étudiées, et les données sont de nature anti persistantes. Ces données nous ont donc permis de faire l'étude et d'obtenir de bons résultats.

La prévision avec les réseaux de neurones à l'aide du logiciel MATLAB donne des résultats très précis avec des erreurs allant de 20 à 30 % dans le cas de moyen terme et une erreur inférieure à 10% pour le court terme. Le bon fonctionnement de ce modèle réside dans le bon choix des paramètres, par exemple le type d'algorithme d'entraînement, le nombre de couches cachées et le type de données à entraîner. Cela représente par conséquent un inconvénient, car nous devons faire différents essais jusqu'en arriver aux meilleurs résultats. Il n'existe pas vraiment une méthodologie universelle pour le choix de l'architecture du réseau neuronal et chaque cas doit être traité différemment selon la nature des données.

Nous devons mentionner que l'utilisation des bibliothèques Python est facile, mais évidemment l'inconvénient ici est la difficulté de trouver des bibliothèques crédibles, assez flexibles et qui donnent de bons résultats pour notre ensemble de données.

La régression linéaire est une méthode classique et peu coûteuse qui donne des résultats de prévisions acceptables et toujours cohérents. Cependant, dans notre cas d'étude, ce sont des résultats les moins efficaces par rapport aux réseaux de neurones artificiels.

Les pics de surconsommation de l'énergie électrique sont toujours problématiques pour le personnel exploitant. Cette étude servira de base pour la mise en œuvre de stratégies pour l'optimisation de fonctionnement des installations électriques qui réduirait significativement les coûts de maintenance corrective. La prévision à moyen terme aide à planifier des maintenances préventives ainsi que la négociation avec les autres fournisseurs d'électricité, car tout écart par rapport à la demande de charge réelle peut entraîner les surcoûts économiques importants. Quant à la prévision à court terme, elle aidera à anticiper les pics de consommation et à prendre les bonnes décisions pour l'intervention au moment opportun afin d'assurer la sécurité des installations.

**Perspective**

Pour poursuivre ce chemin de recherche, il est conseillé d'analyser d'autres algorithmes afin d'améliorer la précision de la prévision. Nous citons à titre d'exemple l'apprentissage profond.

Il est aussi souhaitable de développer sa propre bibliothèque Python pour faire la prévision, introduire d'autres facteurs calendaires dans les modèles tels que les jours fériés, et explorer d'autres logiciels pour la prévision tel que 'R'.

Il est également conseillé de mettre en place de nouvelles installations météorologiques. Ceci pour l'obtention de données plus précises sur les différents paramètres climatiques tels que les températures, l'humidité, le vent et les irradiations solaires.

# Bibliographie


[1] Régie de l'énergie du Canada, "Profils énergétiques des provinces et territoires – Canada." https://www.cer-rec.gc.ca/fr/donnees-analyse/marches-energetiques/profils-energetiques-provinces-territoires/profils-energetiques-provinces-territoires-canada.html (Accédé le 08/13, 2022).

[2] Wikipédia. "Hydro-Québec," https://fr.wikipedia.org/wiki/Hydro-Qu%C3%A9bec (accédé le 08/26, 2022).

[3] Régie de l'énergie du Canada, "Profils énergétiques des provinces et territoires – Québec." https://www.cer-rec.gc.ca/fr/donnees-analyse/marches-energetiques/profils-energetiques-provinces-territoires/profils-energetiques-provinces-territoires-quebec.html (Accédé le 08/13, 2022).

[4] Gouvernement du Québec, "Géographie du territoire québécois." https://www.quebec.ca/gouv/portrait-quebec/geographie-territoire (Accédé le 08/24, 2022).

[5] Ville de Baie-Comeau, "Rapport financier 2019 consolidé." https://www.ville.baie-comeau.qc.ca/ville/budget-et-finances/ (Accédé le 01/22, 2021).

[6] Ville de Baie-Comeau, "Électricité Baie-Comeau." https://ville.baie-comeau.qc.ca/citoyen/electricite-baie-comeau/ (Accédé le 01/22, 2021).

[7] Wikipédia. "Baie-Comeau." https://fr.wikipedia.org/wiki/Baie-Comeau (Accédé le 21/01, 2021).

[8] T. Senjyu, H. Takara, K. Uezato, and T. Funabashi, "One-hour-ahead load forecasting using neural network," *IEEE Transactions on Power Systems,* vol. 17, no. 1, pp. 113-118, 2002, doi: 10.1109/59.982201.

[9] T. Haida and S. Muto, "Regression based peak load forecasting using a transformation technique," *IEEE Transactions on Power Systems,* vol. 9, no. 4, pp. 1788-1794, 1994, doi: 10.1109/59.331433.

[10] Y. Tan, Y. Chen, Y. Li, and Y. Cao, "Linearizing Power Flow Model: A Hybrid Physical Model-Driven and Data-Driven Approach," *IEEE Transactions on Power Systems,* vol. 35, no. 3, pp. 2475-2478, 2020, doi: 10.1109/TPWRS.2020.2975455.

[11] V. Smil, "A Delphi forecast," *Futures,* vol. 9, no. 6, pp. 474-489, 1977/12/01/ 1977, doi: https://doi.org/10.1016/0016-3287(77)90077-5.

[12] J. Jin *et al.*, "Inductive Relation Prediction Using Analogy Subgraph Embeddings," in *International Conference on Learning Representations*, 2021.

[13] F. Amara, "Modélisation et prévision de la demande d'électricité résidentielle," Thése de doctorat, GEI, Université du Québec à Trois-Rivières, Canada, 2018. [Online]. Available: https://depot-e.uqtr.ca/id/eprint/8537/1/032107433.pdf

[14] R. J. H. a. G. Athanasopoulos, *Forecasting: principles and practice*. Australia: Monash University, 2018. oct 2013.

[15] P. R. Winters, "Forecasting Sales by Exponentially Weighted Moving Averages," *Management Science,* vol. 6, no. 3, pp. 324-342, 1960, doi: 10.1287/mnsc.6.3.324.

[16] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *International Journal of Forecasting,* vol. 20, no. 1, pp. 5-10, 2004/01/01/ 2004, doi: https://doi.org/10.1016/j.ijforecast.2003.09.015.

[17] N. R. Draper and H. Smith, *Applied regression analysis*. Department of Statistics at the University of Wisconsin: John Wiley & Sons, 1998, p. 736.

[18] E. Hossain, I. Khan, F. Un-Noor, S. S. Sikander, and M. S. H. Sunny, "Application of Big Data and Machine Learning in Smart Grid, and Associated Security Concerns: A Review," *IEEE Access,* vol. 7, pp. 13960-13988, 2019, doi: 10.1109/ACCESS.2019.2894819.

[19] S. Kavitha, S. Varuna, and R. Ramya, "A comparative analysis on linear regression and support vector regression," presented at the 2016 Online International Conference on Green Engineering and Technologies (IC-GET), Coimbatore, India., 19-19 Nov. 2016, 19 Novembre 2016. [Online]. Available: https://www.proceedings.com/34341.html.

[20] Q. Ding, "Long-Term Load Forecast Using Decision Tree Method," presented at the 2006 IEEE PES Power Systems Conference and Exposition, Atlanta, GA, USA, 29 Oct.-1 Nov. 2006, 2006.

[21] S. Kumar, S. Mishra, and S. Gupta, "Short Term Load Forecasting Using ANN and Multiple Linear Regression," in *2016 Second International Conference on Computational Intelligence & Communication Technology (CICT)*, 12-13 Feb. 2016 2016, pp. 184-186, doi: 10.1109/CICT.2016.44.

[22] A. J. Mehta, H. A. Mehta, T. C. Manjunath, and C. Ardil, "A Multi-layer Artificial Neural Network Architecture Design for Load Forecasting in Power Systems," *International Journal of Electrical and Computer Engineering,* vol. 5, pp. 207-220, 2008.

[23] C. Twumasi and J. Twumasi, "Machine learning algorithms for forecasting and backcasting blood demand data with missing values and outliers: A study of Tema General Hospital of Ghana," (in english), *International Journal of Forecasting,* vol. 38, no. 3, 2021/12/31/ 2021, doi: https://doi.org/10.1016/j.ijforecast.2021.10.008.

[24] P. Schiilkop, C. Burgest, and V. Vapnik, "Extracting support data for a given task," presented at the Proceedings of the 1st international conference on knowledge discovery & data mining, Montréal Québec Canada 1995.

[25] L. Chun-Fu and W. Sheng-De, "Fuzzy support vector machines," *IEEE Transactions on Neural Networks,* vol. 13, no. 2, pp. 464-471, 2002, doi: 10.1109/72.991432.

[26] S. Bo, S. Shi-Ji, and W. Cheng, "A new algorithm of support vector machine based on weighted feature," in *2009 International Conference on Machine Learning and Cybernetics*, Baoding, Hebei, China, IEEE, Ed., 12-15 July 2009 jul, 12 2009, vol. 3: IEEE, pp. 1616-1620, doi: 10.1109/ICMLC.2009.5212256.

[27] H. Zheng and L. Zhang, "The factor analysis of short-term load forecast based on wavelet transform," in *Proceedings. International Conference on Power System Technology*, Kunming, China, I. P. E. S. I. PES), Ed., 13-17 Oct. 2002 2002, vol. 4: IEEE, pp. 1073-1076 vol.2, doi: 10.1109/ICPST.2002.1047565.

[28] J. Wen *et al.*, "Short-term wind power forecasting based on lifting wavelet transform and SVM," in *2012 Power Engineering and Automation Conference*, Wuhan, China, 18-20 Sept. 2012 18-20 September 2012: IEEE, pp. 1-4, doi: 10.1109/PEAM.2012.6612530.

[29] C. Xia, B. Lei, C. Rao, and Z. He, "Research on short-term load forecasting model based on wavelet decomposition and neural network," in *2011 Seventh International Conference on Natural Computation*, china, 26-28 July 2011 2011, vol. 4: IEEE, CAS, pp. 830-834, doi: 10.1109/ICNC.2011.6022226. [Online]. Available: https://ieeexplore.ieee.org/document/6612412

[30] J. Delua. "Supervised vs. Unsupervised Learning: What's the Difference?" IBM. https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning (Accédé le 08/15, 2022).

[31] N. A. Salim, T. K. A. Rahman, M. F. Jamaludin, and M. F. Musa, "Case study of Short Term Load Forecasting for weekends," in *2009 IEEE Student Conference on Research and Development (SCOReD)*, SERDANG, MALAYSIA, U. P. M. FACULTY OF ENGINEERING, Ed., 16-18 Nov. 2009 16 – 18 NOVEMBER 2009: UPM, IEEE, EDS, pp. 332-335, doi: 10.1109/SCORED.2009.5443006.

[32] O. Olabode, O. Amole, T. Ajewole, and I. Okakwu, "Medium-term load forecasting in a nigerian electricity distribution region using regression analysis techniques," in *2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS)*, Ayobo, Nigeria, IEEE, Ed., 18-21 March 2020 18-21 March 2020: IEEE, pp. 1-5, doi: 10.1109/ICMCECS47690.2020.240907.

[33] T. Häring, R. Ahmadiahangar, A. Rosin, T. Korõtko, and H. Biechl, "Accuracy analysis of selected time series and machine learning methods for smart cities based on estonian electricity consumption forecast," in *2020 IEEE 14th International Conference on Compatibility, Power Electronics and Power Engineering (CPE-POWERENG)*, Setúbal, Portugal, I. IEEE, Ed., 8-10 July 2020 08 - 10 July, 2020, vol. 1: Faculdade de Ciências e Tecnologia, Universidade NOVA de Lisboa, Portugal, pp. 425-428, doi: 10.1109/CPE-POWERENG48600.2020.9161690.

[34] P. A. Schirmer, I. Mporas, and I. Potamitis, "Evaluation of regression algorithms in residential energy consumption prediction," in *2019 3rd European Conference on Electrical Engineering and Computer Science (EECS)*, Athens, Greece, 28-30 Dec. 2019 Dec. 28 2019 to Dec. 30 2019: EECS, pp. 22-25, doi: 10.1109/EECS49779.2019.00018.

[35] H. Jia-Jia, "Nonlinear regression analysis of failure probability of power grid equipment based on its condition," in *2016 China International Conference on Electricity Distribution (CICED)*, China, IEEE, Ed., 10-13 Aug. 2016 Aug 2016, pp. 1-5, doi: 10.1109/CICED.2016.7576108.

[36] E. Pekel and S. Kara, "A comprehensive review for artifical neural network application to public transportation," (in English), *Sigma Journal of Engineering and Natural Sciences,* vol. 35, pp. 157-179, 03/01 2017.

[37] D. Abdellah and L. Djamel, "Forecasting the algerian load peak profile using time series model based on backpropagation neural networks," in *4th International Conference on Power Engineering, Energy and Electrical Drives*, Istanbul, Turkey, 13-17 May 2013 13-17 May 2013: IEEE ,IES, pp. 1734-1737, doi: 10.1109/PowerEng.2013.6635879.

[38] J. Delua. "Supervised vs. Unsupervised learning: What's the difference?" IBM Cloud. https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning (Accédé le 03/06, 2022).

[39] M. Parizeau, "Réseaux de neurones " thèse de doctorat, GIF-21140 et GIF-64326, Université Laval, Québec, Canada, 2006.

[40] H. Duong-Ngoc, H. Nguyen-Thanh, and T. Nguyen-Minh, "Short term load forcast using deep learning," in *2019 Innovations in Power and Advanced Computing Technologies (i-PACT)*, China, 22-23 March 2019 22-23 March 2019, vol. 1, pp. 1-5, doi: 10.1109/i-PACT44901.2019.8960036.

[41] E. Lewinson. "Introduction to the Hurst exponent — with code in Python." https://towardsdatascience.com/introduction-to-the-hurst-exponent-with-code-in-python-4da0414ca52e (Accédé le 07/06, 2022).

[42] B. Doucouré, "Proposition, intégration dans un système de gestion de réseau intelligent et validation expérimentale d'une méthode de prédiction pour un système d'énergies renouvelables," Université du Québec à Trois-Rivières, Québec, Canada, 2015. [Online]. Available: http://depot-e.uqtr.ca/7674/1/031014383.pdf

[43] B. Qian and K. Rasheed, "Hurst exponent and financial market predictability," in *IASTED conference on Financial Engineering and Applications*, 2004: Proceedings of the IASTED International Conference Cambridge, MA, pp. 203-209.

[44] Q. Wang, X. Wang, and F. Xia, "Integration of grey model and multiple regression model to predict energy consumption," in *2009 International Conference on Energy and Environment Technology*, Guilin, China, 16-18 Oct. 2009 16-18 October 2009, vol. 1: Computer Socity Press, pp. 194-197, doi: 10.1109/ICEET.2009.53.

[45] Simeb," Simulation énergétique des bâtiments ". https://www.simeb.ca/ (Accédé le 02/01, 2021).

[46] QuantStart. "Basics of statistical mean reversion testing." QUANTSTART. https://www.quantstart.com/articles/Basics-of-Statistical-Mean-Reversion-Testing/ Accédé le 02/06, 2022).

[47] Skforecast Documents, "Recursive multi-step forecasting." https://joaquinamatrodrigo.github.io/skforecast/0.4.3/notebooks/autoregresive-forecaster.html (Accédé le 03/22, 2022).

[48] J. E. O. Joaquín Amat Rodrigo. "Skforecast: time series forecasting with Python and Scikit-learn." https://www.cienciadedatos.net/documentos/py27-time-series-forecasting-python-scikitlearn.html Accédé le 06/08, 2022).

[49] TensorFlow tutorials ,"Time series forecasting." https://www.tensorflow.org/tutorials/structured_data/time_series (Accédé le 01/03, 2022).

# ANNEXE A

La suite des résultats obtenus avec Python pour toutes les lignes, aux chapitres (4.6.2)

Tableaux A.1 Suite des résultats pour les lignes 2, 3 et 4.

| | | MAPE (%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Ligne 2 | | | Ligne 3 | | | Ligne 4 | | |
| Modèle | Quantité | 1 FEV | 2 FEV | 3 FEV | 1 FEV | 2 FEV | 3 FEV | 1 FEV | 2 FEV | 3 FEV |
| ANN | 1Semaine | 28.11 | 13.16 | 33.19 | 1.11 | 73.62 | 48.74 | 27.12 | 106.38 | 38.29 |
| | 15 jours | 22.93 | 13.72 | 20.84 | 0.69 | 19.05 | 31.01 | 8.69 | 15.26 | 26.54 |
| | 1 Mois | 5.28 | 6.20 | 6.67 | 0.83 | 21.95 | 11.81 | 4.33 | 8.04 | 11.03 |
| | 2 Mois | 5.02 | 6.35 | 4.95 | 0.82 | 22.73 | 15.56 | 4.47 | 8.44 | 10.91 |
| | 3 Mois | 4.75 | 3.39 | 7.01 | 0.74 | 20.17 | 12.02 | 4.22 | 7.68 | 11.44 |
| RL | 1Semaine | 14.69 | 14.85 | 18.97 | 23.28 | 19.27 | 15.06 | 4.24 | 7.80 | 10.62 |
| | 15 jours | 4.58 | 4.52 | 4.79 | 23.56 | 20.26 | 11.88 | 4.65 | 8.63 | 10.76 |
| | 1 Mois | 5.28 | 6.13 | 4.90 | 24.37 | 22.00 | 11.81 | 4.75 | 8.99 | 11.08 |
| | 2 Mois | 5.02 | 5.95 | 4.85 | 24.37 | 22.29 | 12.52 | 4.33 | 8.52 | 10.75 |
| | 3 Mois | 4.78 | 3.47 | 4.87 | 20.55 | 20.03 | 12.02 | 4.28 | 7.52 | 10.38 |

Tableau A.2 Suite des résultats pour les lignes 6, 7 et 8.

| | | MAPE (%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Ligne 6 | | | Ligne 7 | | | Ligne 8 | | |
| Modèle | Quantité | 1 FEV | 2 FEV | 3 FEV | 1 FEV | 2 FEV | 3 FEV | 1 FEV | 2 FEV | 3 FEV |
| ANN | 1Semaine | 26.84 | 18.31 | 40.81 | 17.04 | 11.17 | 36.19 | 21.66 | 12.91 | 28.56 |
| | 15 jours | 24.24 | 6.03 | 19.98 | 6.20 | 27.32 | 21.89 | 12.85 | 16.97 | 32.80 |
| | 1 Mois | 15.93 | 7.78 | 4.66 | 4.69 | 10.24 | 10.96 | 3.18 | 6.39 | 9.15 |
| | 2 Mois | 0.25 | 7.70 | 5.35 | 4.43 | 9.77 | 11.24 | 3.02 | 7.87 | 9.11 |
| | 3 Mois | 0.28 | 6.25 | 4.72 | 4.53 | 8.03 | 10.62 | 2.97 | 6.84 | 8.81 |
| RL | 1Semaine | 8.49 | 6.51 | 4.42 | 4.76 | 9.16 | 11.03 | 3.38 | 6.52 | 9.61 |
| | 15 jours | 8.73 | 7.09 | 4.55 | 4.81 | 10.04 | 11.58 | 3.24 | 7.58 | 9.04 |
| | 1 Mois | 8.25 | 7.98 | 5.50 | 4.72 | 10.29 | 12.02 | 3.37 | 8.14 | 9.28 |
| | 2 Mois | 7.65 | 7.59 | 5.27 | 4.48 | 9.50 | 11.26 | 3.06 | 7.72 | 9.09 |
| | 3 Mois | 6.18 | 6.30 | 4.69 | 4.78 | 8.18 | 10.59 | 2.97 | 6.71 | 8.79 |

# ANNEXE B

Les figures ci-dessous montrent le modèle Simulink et ses propriétés pour faire la régression linéaire. Ensuite le script MATLAB utilisé au chapitre (4.5) pour faire la corrélation et la régression.

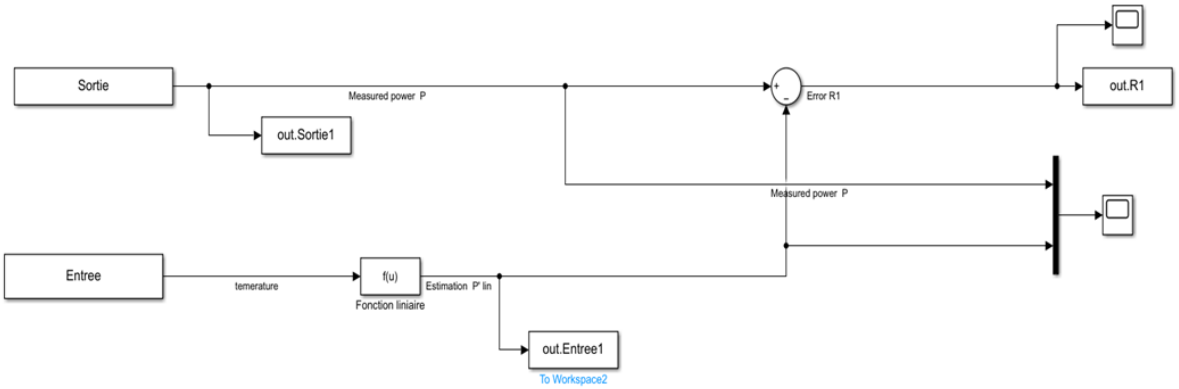
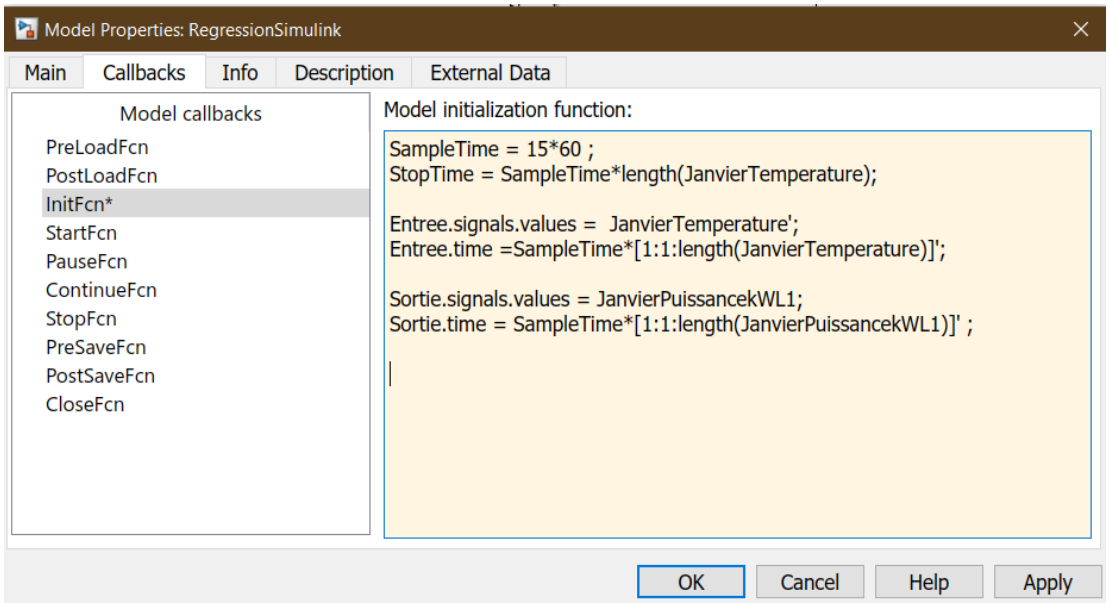



Figure B.1 Le modèle Simulink.




FigureB.2 Les propriétés du modèle Simulink.

Script MATLAB :

```
clc;
clear all;
close all;
warning off;
load('Year2020real.mat')
% Les données météologiques
Temperature_1h=te;
TemperatureD_1h=td;
Vent_1h=vv;
Irradiation_1h=irr;
% Extrapolation de données
Date_start = datenum('2020/01/01 01:00');
Sample_rate = 1/24;
Time_1h =
[Date_start:Sample_rate:Date_start+Sample_rate*(length(te)-1)];
Time_15m = [Date_start-
Sample_rate/2:Sample_rate/4:Date_start+Sample_rate*(length(te)-
1)+Sample_rate/4].';
Temperature_15m = (interp1(Time_1h,Temperature_1h, Time_15m,
'spline','extrap')).';
TemperatureD_15m= (interp1(Time_1h,TemperatureD_1h, Time_15m,
'spline','extrap')).';
Vent_15m= (interp1(Time_1h,Vent_1h, Time_15m,
'spline','extrap')).';
Irradiation_15m= (interp1(Time_1h,Irradiation_1h, Time_15m,
'spline','extrap')).';
%chargement des données des Lignes pour 2020
PuissancekW_L1=BC_PuissancekW(:,1);
PuissancekW_L2=BC_PuissancekW(:,2);
PuissancekW_L3=BC_PuissancekW(:,3);
```

```
PuissancekW_L4=BC_PuissancekW(:,4);
PuissancekW_L5=BC_PuissancekW(:,5);
PuissancekW_L6=BC_PuissancekW(:,6);
PuissancekW_L7=BC_PuissancekW(:,7);
PuissancekW_L8=BC_PuissancekW(:,8);
%les données par mois
%janvier
JanvierPuissancekWL1=PuissancekW_L1(1:2975);
JanvierPuissancekWL2=PuissancekW_L2(1:2975);
JanvierPuissancekWL3=PuissancekW_L3(1:2975);
JanvierPuissancekWL4=PuissancekW_L4(1:2975);
JanvierPuissancekWL5=PuissancekW_L5(1:2975);
JanvierPuissancekWL6=PuissancekW_L6(1:2975);
JanvierPuissancekWL7=PuissancekW_L7(1:2975);
JanvierPuissancekWL8=PuissancekW_L8(1:2975);
JanvierTemperature=Temperature_15m(1:2975);
JanvierTemperatureD=TemperatureD_15m(1:2975);
JanvierVent=Vent_15m(1:2975);
JanvierIrradiation=Irradiation_15m(1:2975);
TimeDD=Time_15m(1:2976);
%fevrier
FevrierPuissancekWL1=PuissancekW_L1(2976:5759);
FevrierPuissancekWL2=PuissancekW_L2(2976:5759);
FevrierPuissancekWL3=PuissancekW_L3(2976:5759);
FevrierPuissancekWL4=PuissancekW_L4(2976:5759);
FevrierPuissancekWL5=PuissancekW_L5(2976:5759);
FevrierPuissancekWL6=PuissancekW_L6(2976:5759);
FevrierPuissancekWL7=PuissancekW_L7(2976:5759);
FevrierPuissancekWL8=PuissancekW_L8(2976:5759);
FevrierTemperature=Temperature_15m(2976:5759);
FevrierTemperatureD=TemperatureD_15m(2976:5759);
FevrierVent=Vent_15m(2976:5759);
FevrierIrradiation=Irradiation_15m(2976:5759);
%Mars
MarsPuissancekWL1=PuissancekW_L1(5760:8735);
MarsPuissancekWL2=PuissancekW_L2(5760:8735);
MarsPuissancekWL3=PuissancekW_L3(5760:8735);
MarsPuissancekWL4=PuissancekW_L4(5760:8735);
MarsPuissancekWL5=PuissancekW_L5(5760:8735);
MarsPuissancekWL6=PuissancekW_L6(5760:8735);
MarsPuissancekWL7=PuissancekW_L7(5760:8735);
MarsPuissancekWL8=PuissancekW_L8(5760:8735);
MarsTemperature=Temperature_15m(5760:8735);
MarsTemperatureD=TemperatureD_15m(5760:8735);
MarsVent=Vent_15m(5760:8735);
MarsIrradiation=Irradiation_15m(5760:8735);
%Avril
AvrilPuissancekWL1=PuissancekW_L1(8736:11615);
AvrilPuissancekWL2=PuissancekW_L2(8736:11615);
AvrilPuissancekWL3=PuissancekW_L3(8736:11615);
AvrilPuissancekWL4=PuissancekW_L4(8736:11615);
AvrilPuissancekWL5=PuissancekW_L5(8736:11615);
```

```
AvrilPuissancekWL6=PuissancekW_L6(8736:11615);
AvrilPuissancekWL7=PuissancekW_L7(8736:11615);
AvrilPuissancekWL8=PuissancekW_L8(8736:11615);
AvrilTemperature=Temperature_15m(8736:11615);
AvrilTemperatureD=TemperatureD_15m(8736:11615);
AvrilVent=Vent_15m(8736:11615);
AvrilIrradiation=Irradiation_15m(8736:11615);
%Mai
MaiPuissancekWL1=PuissancekW_L1(11616:14591);
MaiPuissancekWL2=PuissancekW_L2(11616:14591);
MaiPuissancekWL3=PuissancekW_L3(11616:14591);
MaiPuissancekWL4=PuissancekW_L4(11616:14591);
MaiPuissancekWL5=PuissancekW_L5(11616:14591);
MaiPuissancekWL6=PuissancekW_L6(11616:14591);
MaiPuissancekWL7=PuissancekW_L7(11616:14591);
MaiPuissancekWL8=PuissancekW_L8(11616:14591);
MaiTemperature=Temperature_15m(11616:14591);
MaiTemperatureD=TemperatureD_15m(11616:14591);
MaiVent=Vent_15m(11616:14591);
MaiIrradiation=Irradiation_15m(11616:14591);
%Juin
JuinPuissancekWL1=PuissancekW_L1(14592:17471);
JuinPuissancekWL2=PuissancekW_L2(14592:17471);
JuinPuissancekWL3=PuissancekW_L3(14592:17471);
JuinPuissancekWL4=PuissancekW_L4(14592:17471);
JuinPuissancekWL5=PuissancekW_L5(14592:17471);
JuinPuissancekWL6=PuissancekW_L6(14592:17471);
JuinPuissancekWL7=PuissancekW_L7(14592:17471);
JuinPuissancekWL8=PuissancekW_L8(14592:17471);
JuinTemperature=Temperature_15m(14592:17471);
JuinTemperatureD=TemperatureD_15m(14592:17471);
JuinVent=Vent_15m(14592:17471);
JuinIrradiation=Irradiation_15m(14592:17471);
%Juillet
JuilletPuissancekWL1=PuissancekW_L1(17472:20447);
JuilletPuissancekWL2=PuissancekW_L2(17472:20447);
JuilletPuissancekWL3=PuissancekW_L3(17472:20447);
JuilletPuissancekWL4=PuissancekW_L4(17472:20447);
JuilletPuissancekWL5=PuissancekW_L5(17472:20447);
JuilletPuissancekWL6=PuissancekW_L6(17472:20447);
JuilletPuissancekWL7=PuissancekW_L7(17472:20447);
JuilletPuissancekWL8=PuissancekW_L8(17472:20447);
JuilletTemperature=Temperature_15m(17472:20447);
JuilletTemperatureD=TemperatureD_15m(17472:20447);
JuilletVent=Vent_15m(17472:20447);
JuilletIrradiation=Irradiation_15m(17472:20447);
%Aout
AoutPuissancekWL1=PuissancekW_L1(20448:23423);
AoutPuissancekWL2=PuissancekW_L2(20448:23423);
AoutPuissancekWL3=PuissancekW_L3(20448:23423);
AoutPuissancekWL4=PuissancekW_L4(20448:23423);
AoutPuissancekWL5=PuissancekW_L5(20448:23423);
```

```
AoutPuissancekWL6=PuissancekW_L6(20448:23423);
AoutPuissancekWL7=PuissancekW_L7(20448:23423);
AoutPuissancekWL8=PuissancekW_L8(20448:23423);
AoutTemperature=Temperature_15m(20448:23423);
AoutTemperatureD=TemperatureD_15m(20448:23423);
AoutVent=Vent_15m(20448:23423);
AoutIrradiation=Irradiation_15m(20448:23423);
%Septembre
SeptembrePuissancekWL1=PuissancekW_L1(23424:26303);
SeptembrePuissancekWL2=PuissancekW_L1(23424:26303);
SeptembrePuissancekWL3=PuissancekW_L1(23424:26303);
SeptembrePuissancekWL4=PuissancekW_L1(23424:26303);
SeptembrePuissancekWL5=PuissancekW_L1(23424:26303);
SeptembrePuissancekWL6=PuissancekW_L1(23424:26303);
SeptembrePuissancekWL7=PuissancekW_L1(23424:26303);
SeptembrePuissancekWL8=PuissancekW_L1(23424:26303);
SeptembreTemperature=Temperature_15m(23424:26303);
SeptembreTemperatureD=TemperatureD_15m(23424:26303);
SeptembreVent=Vent_15m(23424:26303);
SeptembreIrradiation=Irradiation_15m(23424:26303);
%Octobre
OctobrePuissancekWL1=PuissancekW_L1(26304:29279);
OctobrePuissancekWL2=PuissancekW_L2(26304:29279);
OctobrePuissancekWL3=PuissancekW_L3(26304:29279);
OctobrePuissancekWL4=PuissancekW_L4(26304:29279);
OctobrePuissancekWL5=PuissancekW_L5(26304:29279);
OctobrePuissancekWL6=PuissancekW_L6(26304:29279);
OctobrePuissancekWL7=PuissancekW_L7(26304:29279);
OctobrePuissancekWL8=PuissancekW_L8(26304:29279);
OctobreTemperature=Temperature_15m(26304:29279);
OctobreTemperatureD=TemperatureD_15m(26304:29279);
OctobreVent=Vent_15m(26304:29279);
OctobreIrradiation=Irradiation_15m(26304:29279);
%Novembre
NovembrePuissancekWL1=PuissancekW_L1(29280:32159);
NovembrePuissancekWL2=PuissancekW_L2(29280:32159);
NovembrePuissancekWL3=PuissancekW_L3(29280:32159);
NovembrePuissancekWL4=PuissancekW_L4(29280:32159);
NovembrePuissancekWL5=PuissancekW_L5(29280:32159);
NovembrePuissancekWL6=PuissancekW_L6(29280:32159);
NovembrePuissancekWL7=PuissancekW_L7(29280:32159);
NovembrePuissancekWL8=PuissancekW_L8(29280:32159);
NovembreTemperature=Temperature_15m(29280:32159);
NovembreTemperatureD=TemperatureD_15m(29280:32159);
NovembreVent=Vent_15m(29280:32159);
NovembreIrradiation=Irradiation_15m(29280:32159);
%Decembre
DecembrePuissancekWL1=PuissancekW_L1(32160:end);
DecembrePuissancekWL2=PuissancekW_L2(32160:end);
DecembrePuissancekWL3=PuissancekW_L3(32160:end);
DecembrePuissancekWL4=PuissancekW_L4(32160:end);
DecembrePuissancekWL5=PuissancekW_L5(32160:end);
```

```
DecembrePuissancekWL6=PuissancekW_L6(32160:end);
DecembrePuissancekWL7=PuissancekW_L7(32160:end);
DecembrePuissancekWL8=PuissancekW_L8(32160:end);
DecembreTemperature=Temperature_15m(32160:end);
DecembreTemperatureD=TemperatureD_15m(32160:end);
DecembreVent=Vent_15m(32160:end);
DecembreIrradiation=Irradiation_15m(32160:end);
%année
yearPuissancekWL1=PuissancekW_L1(1:end);
yearPuissancekWL2=PuissancekW_L2(1:end);
yearPuissancekWL3=PuissancekW_L3(1:end);
yearPuissancekWL4=PuissancekW_L4(1:end);
yearPuissancekWL5=PuissancekW_L5(1:end);
yearPuissancekWL6=PuissancekW_L6(1:end);
yearPuissancekWL7=PuissancekW_L7(1:end);
yearPuissancekWL8=PuissancekW_L8(1:end);
yearTemperature=Temperature_15m(1:end);
yearTemperatureD=TemperatureD_15m(1:end);
yearVent=Vent_15m(1:end);
yearIrradiation=Irradiation_15m(1:end);
%Correlation et regression
x = JanvierTemperature;
y = JanvierPuissancekWL1;
plot(x, y, '*','displayname','Scatterplot');
xlabel("Température (°C)");
ylabel("Puissance (MW)");
title("Regréssion pour ligne 1" );
out.Entree1=out.Entree1/1000;
out.Sortie1=out.Sortie1/1000;
err = out.Entree1-out.Sortie1;
MAE = mean(abs(err));
figure()
subplot(2,1,1);
plot(TimeDD,out.Entree1)
hold
plot(TimeDD,out.Sortie1)
legend('Estimated','Measured')
title('Forecast With liniare Regression for line
1','FontSize',12,'FontWeight','bold','Color','k');
xlabel('Time
(Days)','FontSize',12,'FontWeight','bold','Color','k');
ylabel('Power (MW)','FontSize',12,'FontWeight','bold','Color','k')
datetick('x','dd/mm', 'keeplimits')
subplot(2,1,2);
plot(TimeDD,err)
legend('Residuals')
title(sprintf('Mean Absolute Percent Error (MAPE): %0.2f%% \nMean
Absolute Error (MAE): %0.2f MWh\n',MAPE, MAE));
xlabel('Time
(Days)','FontSize',12,'FontWeight','bold','Color','k');
ylabel('Error','FontSize',12,'FontWeight','bold','Color','k')
datetick('x','dd/mm', 'keeplimits')
```

# ANNEXE C

Le script MATLAB suivant est celui utilisé au chapitre (4) pour faire la prévision avec les réseaux de neurones :

```
%Réseau de neurones
clear all;
close all;
warning off;
load('Year2020real.mat')
%Les données météologiques
Temperature_1h=te;
TemperatureD_1h=td;
Vent_1h=vv;
Irradiation_1h=irr;
%Extrapolation de données
Date_start = datenum('2020/01/01 01:00');
Sample_rate = 1/24;
Time_1h
=[Date_start:Sample_rate:Date_start+Sample_rate*(length(te)-1)];
Time_15m = [Date_start-
Sample_rate/2:Sample_rate/4:Date_start+Sample_rate*(length(te)-
1)+Sample_rate/4].';
Temperature_15m = (interp1(Time_1h,Temperature_1h, Time_15m,
'spline','extrap')).';
TemperatureD_15m= (interp1(Time_1h,TemperatureD_1h, Time_15m,
'spline','extrap')).';
Vent_15m= (interp1(Time_1h,Vent_1h, Time_15m,
'spline','extrap')).';
Irradiation_15m= (interp1(Time_1h,Irradiation_1h, Time_15m,
'spline','extrap')).';
% Charger données de puissance
PuissancekW_L1=BC_PuissancekW(:,1);
PuissancekW_L2=BC_PuissancekW(:,2);
PuissancekW_L3=BC_PuissancekW(:,3);
PuissancekW_L4=BC_PuissancekW(:,4);
PuissancekW_L5=BC_PuissancekW(:,5);
PuissancekW_L6=BC_PuissancekW(:,6);
PuissancekW_L7=BC_PuissancekW(:,7);
PuissancekW_L8=BC_PuissancekW(:,8);
%Préparation de données pour l'entraînement
NTests = 24;
for k=1:NTests
clear Y_test Xtest Y_est
clear X Y
H=24*4;
days = 30;
```

```
LD =  24*4*days;
d=4*k;
X(1,:) = Temperature_15m(1+d:LD+d);
X(2,:) = TemperatureD_15m(1+d:LD+d);
X(3,:) = Vent_15m(1+d:LD+d);
X(4,:) = Irradiation_15m(1+d:LD+d);
X(4+k,:) =  PuissancekW_L3(1+d-k:LD+d-k)';
Y(1,:) = PuissancekW_L3(1+d+H:LD+d+H)';
%définition de nombre de couche caché
net = fitnet(5,'trainscg');
net.divideParam.trainInd = 1:720;
net.divideParam.valInd = 721:1080;
net.divideParam.testInd = 1081:1441;
%entraînement
net = train(net,X,Y,'useParallel','yes','showResources','yes');
clear Y_test Xtest Y_est
Y_test =  PuissancekW_L3(1+d+H:LD+d+H)';
Xtest(1,:) = Temperature_15m(1+d+H:LD+d+H);
Xtest(2,:) = TemperatureD_15m(1+d+H:LD+d+H);
Xtest(3,:) = Vent_15m(1+d+H:LD+d+H);
Xtest(4,:) = Irradiation_15m(1+d+H:LD+d+H);
Xtest(4+k,:) =  PuissancekW_L3(1+d+H-k:LD+d+H-k)';
Y_est = net(Xtest);
Y_est=Y_est(LD+1-H:LD);
Y_test=Y_test(LD+1-H:LD);
Time_test = Time_15m(1+d+H:LD+d+H)';
FuturTime(k,:)=Time_test(LD+1-H:LD);
T_Start = FuturTime(k,1);
T_End = FuturTime(k,end);
Y_est=Y_est/1000;
Y_test=Y_test/1000;
Erreur(k,:) = Y_test-Y_est;
%calcule d'erreur RMSE pour ANN
ANN_RMSE(k,:)=sqrt(mean((Y_test-Y_est).^2));
%calcule de Root Mean Square Error
ANN_MAPE(k,:) = mean(100 * abs(Y_test-Y_est) ./ Y_est);
%Calcule Mean Absolute Percent Error
ANN_MSE(k,:)= immse(Y_test,Y_est);
%calcule Mean Square Error for
figure()
x0=100; y0=300; width=1000; height=450;
set(gcf,'position',[x0,y0,width,height])
plot(FuturTime(k,3:end),Y_est(3:end), 'linewidth',2)
hold
plot(FuturTime(k,3:end), Y_test(3:end), 'linewidth',2)
xlim([T_Start T_End]);
legend('NN model','Measured')
text_title=['Prédiction avec le modèle ANN pour Ligne 7,
',num2str(k), ' heures de décalage' ];
xlabel('Temps
(heures)','FontSize',12,'FontWeight','bold','Color','k');
ylabel('Power (MW)','FontSize',12,'FontWeight','bold','Color','k')
```

```
grid on;
datetick('x','dd-mmm hh:MM', 'keeplimits')
set(gca,'XTickLabelRotation',90);
title(text_title);
end
xdata = [1:1:H]/4;
figure()
plot(xdata,100*abs(mean(Erreur)), 'linewidth',2)
x0=100; y0=300; width=1000; height=350;
set(gcf,'position',[x0,y0,width,height])
xlabel('Horizon
(Heures)','FontSize',12,'FontWeight','bold','Color','k')
ylabel('Erreur [%]','FontSize',12,'FontWeight','bold','Color','k')
title('MAE','FontSize',12,'FontWeight','bold','Color','k')
ylim([ 0 50])
grid on
legend('ANN')
```

Les figures suivantes montrent la suite de la boucle jusqu'à 24 heures de décalage des résultats obtenus avec ANN au (4.6.1) :

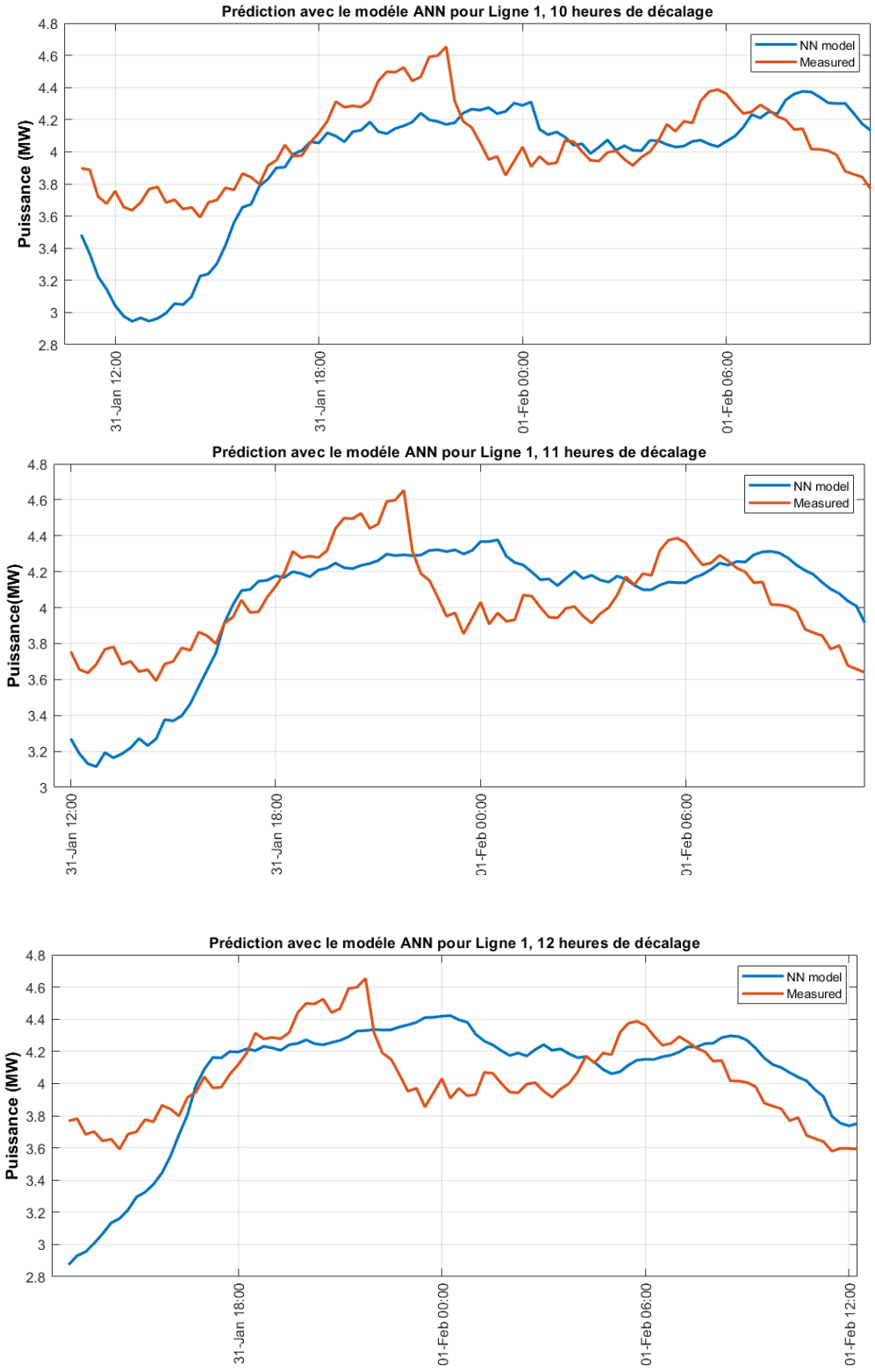



Figure C.1 Les prévisions Avec ANN pour la ligne 1, heure par heure (10 à 12 heures).

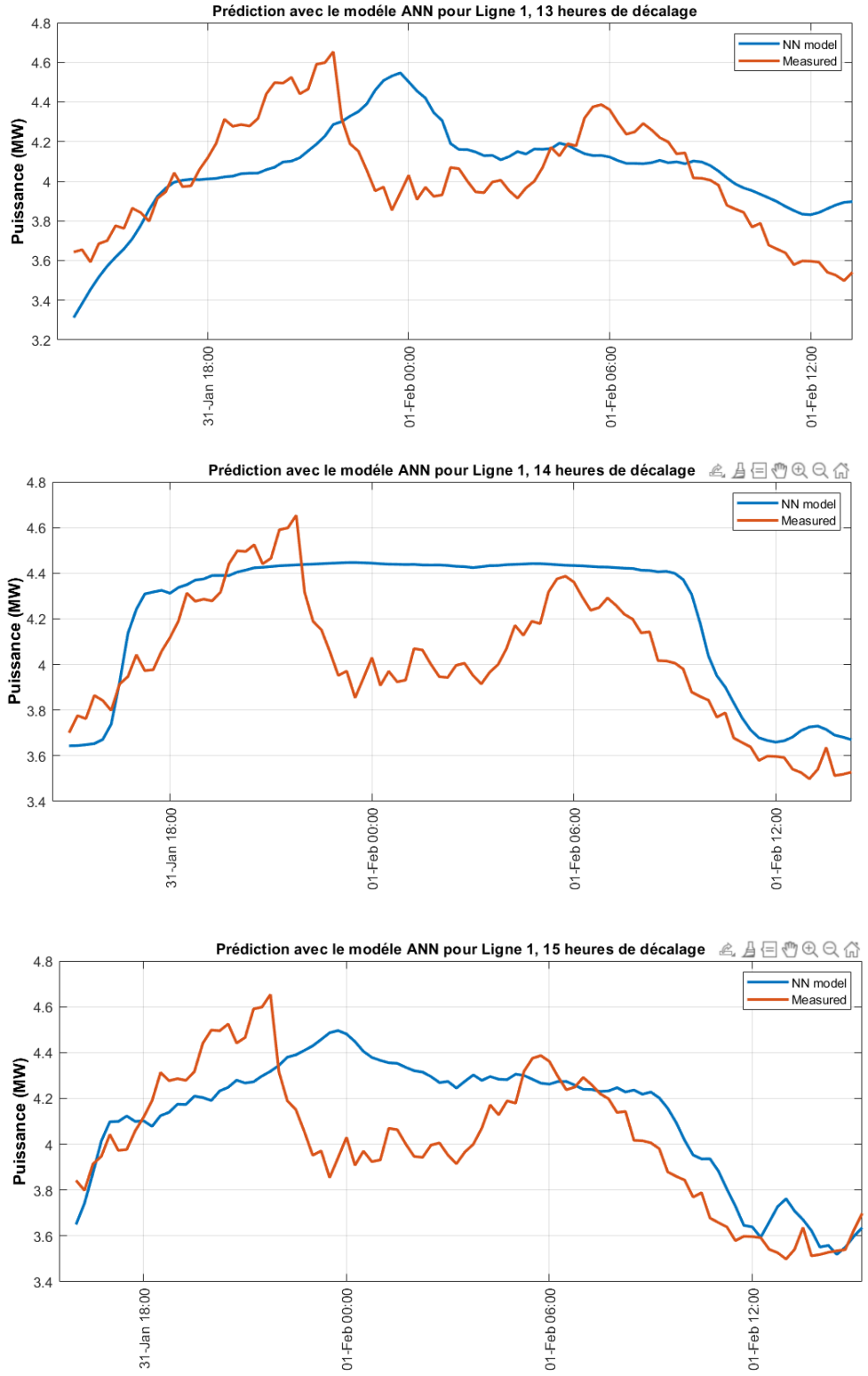



Figure C.2 Les prévisions Avec ANN pour la ligne 1, heure par heure (13 à 15 heures).

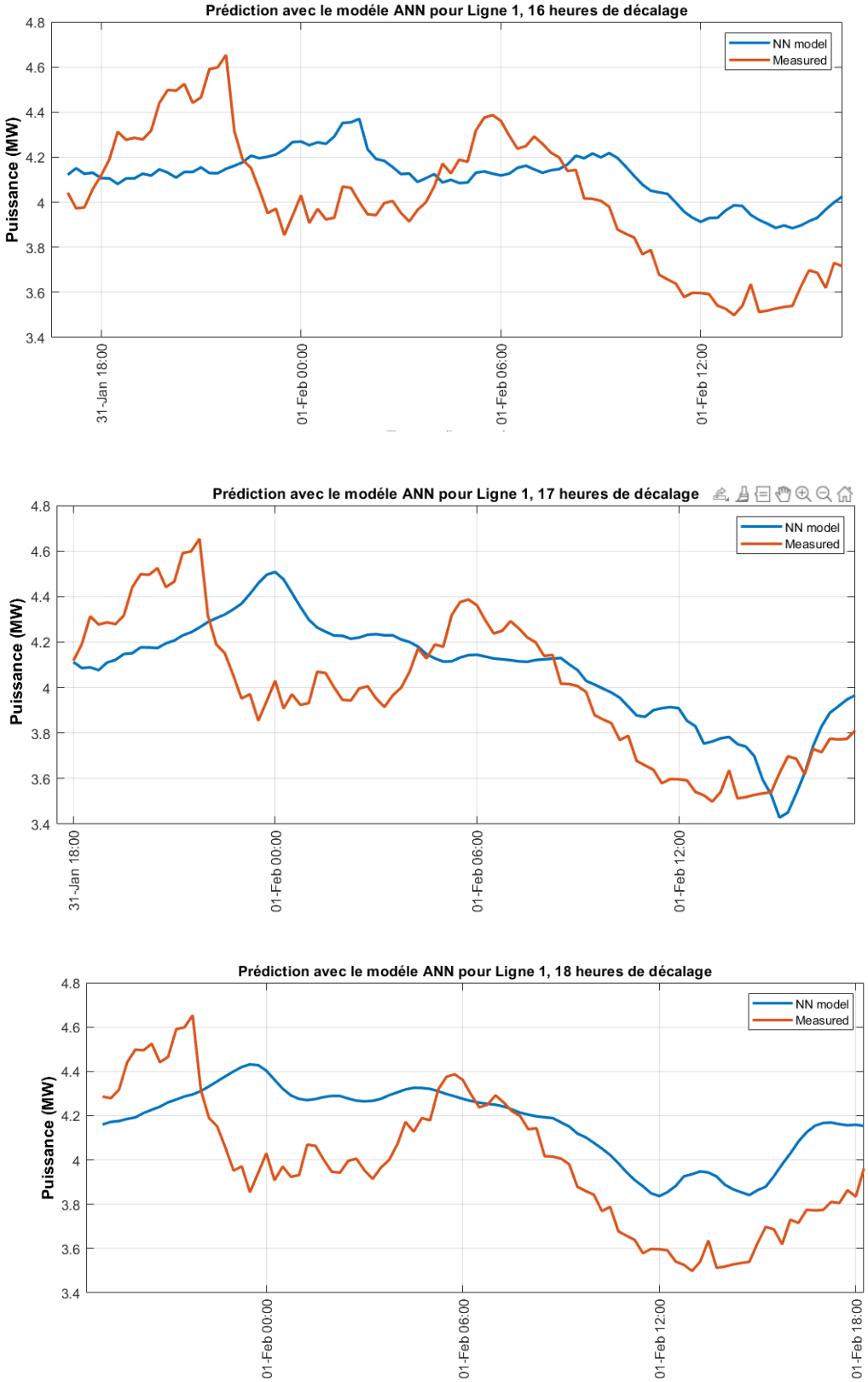



Figure C.3 Les prévisions Avec ANN pour la ligne 1, heure par heure (16 à 18 heures).

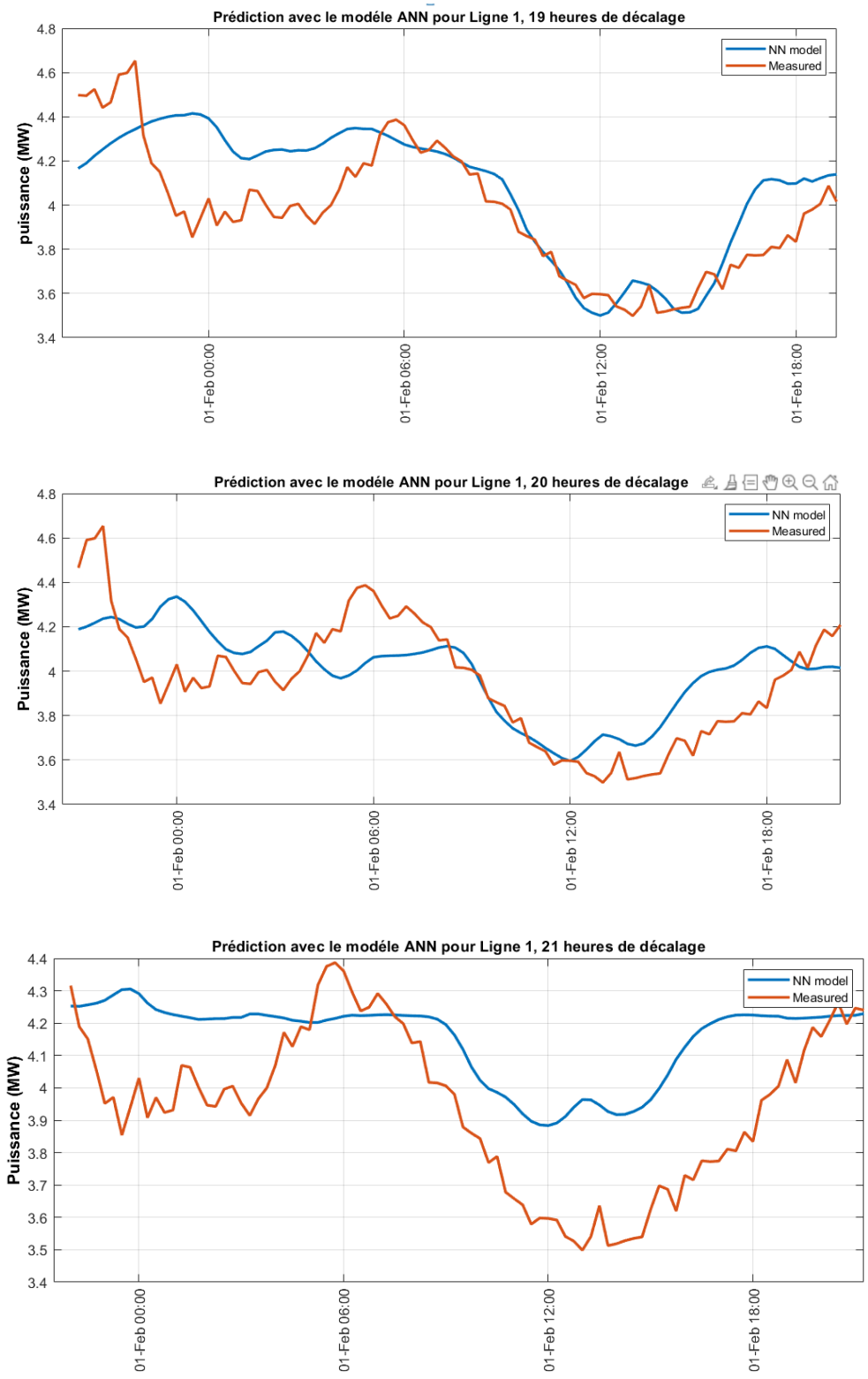



Figure C.4 Les prévisions Avec ANN pour la ligne 1, heure par heure (19 à 21 heures).

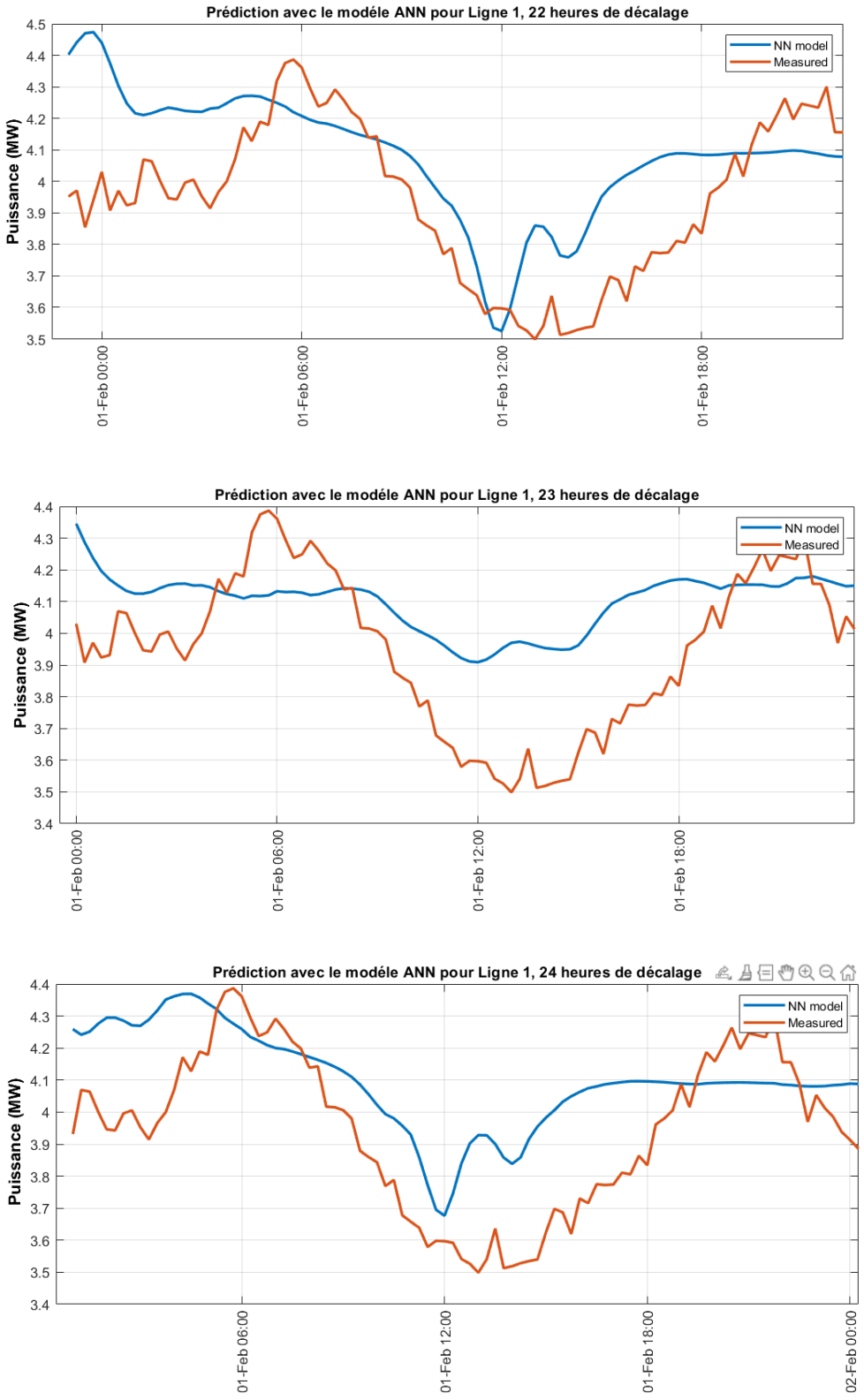



Figure C.5 Les prévisions Avec ANN pour la ligne 1, heure par heure (22 à 24 heures).

# ANNEXE D

Le script Pyhthon pour la prévision avec les réseaux de neurones pour la ligne 1 (1 Février) :

```
1. import numpy as np
2. import matplotlib.pyplot as plt
3. import pandas as pd
4. import datetime as dt
5. import requests
6. from datetime import datetime
7. from scipy.interpolate import interp1d
8. from tensorflow import keras
9. from sklearn.metrics import*
10.
11. data=pd.read_excel('DataSetBC.xlsx', index_col='Time',parse_dates=True)
12. temp_fcast = data['2020-02-01 00:00:00':'2020-02-02 00:00:00']['Tseche']
13. index_future_dates=data['2020-02-01 00:00:00':'2020-02-02
    00:00:00'].index
14.
15. ### Ligne 1
16. # 1 Semaine L1
17. S1L1 = pd.DataFrame({'puissancel1':(data['2020-01-01 00:00:00':'2020-01-
    07 00:00:00']['PuissancekW_L1']/1000),'temperature':(data['2020-01-01
    00:00:00':'2020-01-07 00:00:00']['Tseche'])})
18. PowerS1_L1=S1L1['puissancel1'].values.reshape(-1,1)
19. TempS1=S1L1['temperature'].values.reshape(-1,1)
20. # 15 jours L1
21. D15L1 = pd.DataFrame({'puissancel1':(data['2020-01-01 00:00:00':'2020-01-
    15 00:00:00']['PuissancekW_L1']/1000),'temperature':(data['2020-01-01
    00:00:00':'2020-01-15 00:00:00']['Tseche'])})
22. PowerD15_L1=D15L1['puissancel1'].values.reshape(-1,1)
23. TempD15=D15L1['temperature'].values.reshape(-1,1)
24. # 1 mois   L1
25. M1L1 = pd.DataFrame({'puissancel1':(data['2020-
    01']['PuissancekW_L1']/1000),'temperature':(data['2020-01']['Tseche'])})
26. PowerM1_L1=M1L1['puissancel1'].values.reshape(-1,1)
27. TempM1=M1L1['temperature'].values.reshape(-1,1)
28. # 2 mois   L1
29. M2L1 = pd.DataFrame({'puissancel1':(data['2020-01':'2020-
    02']['PuissancekW_L1']/1000),'temperature':(data['2020-01':'2020-
    02']['Tseche'])})
30. PowerM2_L1=M2L1['puissancel1'].values.reshape(-1,1)
31. TempM2=M2L1['temperature'].values.reshape(-1,1)
32. # 3 mois   L1
33. M3L1 = pd.DataFrame({'puissancel1':(data['2020-01':'2020-
    03']['PuissancekW_L1']/1000),'temperature':(data['2020-01':'2020-
    03']['Tseche'])})
34. PowerM3_L1=M3L1['puissancel1'].values.reshape(-1,1)
```

```
35. TempM3=M3L1['temperature'].values.reshape(-1,1)
36.
37. #ligne1
38. # 1 Semaine L1
39. modelS1L1 = keras.Sequential()
40. modelS1L1.add(keras.layers.Dense(units = 1, activation = 'linear',
    input_shape=[1]))
41. modelS1L1.compile(loss='mse', optimizer="adam")
42. modelS1L1.fit( TempS1, PowerS1_L1, epochs=100, verbose=1 )
43. y_predictedS1L1 = modelS1L1.predict(temp_fcast)
44. y_predictedS1L1 = np.concatenate(y_predictedS1L1)
45. dANNS1L1 =
    pd.DataFrame({'Datefuture':(index_future_dates),'PuissanceANNS1L1':(y_pre
    dictedS1L1)})
46. dANNS1L1.set_index('Datefuture',inplace=True)
47. # 15 jours L1
48. modelD15L1 = keras.Sequential()
49. modelD15L1.add(keras.layers.Dense(units = 1, activation = 'linear',
    input_shape=[1]))
50. modelD15L1.compile(loss='mse', optimizer="adam")
51. modelD15L1.fit( TempD15, PowerD15_L1, epochs=100, verbose=1 )
52. y_predictedD15L1 = modelD15L1.predict(temp_fcast)
53. y_predictedD15L1 = np.concatenate(y_predictedD15L1)
54. dANND15L1 =
    pd.DataFrame({'Datefuture':(index_future_dates),'PuissanceANND15L1':(y_pr
    edictedD15L1)})
55. dANND15L1.set_index('Datefuture',inplace=True)
56. # 1 mois L1
57. modelM1L1 = keras.Sequential()
58. modelM1L1.add(keras.layers.Dense(units = 1, activation = 'linear',
    input_shape=[1]))
59. modelM1L1.compile(loss='mse', optimizer="adam")
60. modelM1L1.fit( TempM1, PowerM1_L1, epochs=100, verbose=1 )
61. y_predictedM1L1 = modelM1L1.predict(temp_fcast)
62. y_predictedM1L1 = np.concatenate(y_predictedM1L1)
63. dANNM1L1 =
    pd.DataFrame({'Datefuture':(index_future_dates),'PuissanceANNM1L1':(y_pre
    dictedM1L1)})
64. dANNM1L1.set_index('Datefuture',inplace=True)
65. # 2 mois L1
66. modelM2L1 = keras.Sequential()
67. modelM2L1.add(keras.layers.Dense(units = 1, activation = 'linear',
    input_shape=[1]))
68. modelM2L1.compile(loss='mse', optimizer="adam")
69. modelM2L1.fit( TempM2, PowerM2_L1, epochs=100, verbose=1 )
70. y_predictedM2L1 = modelM2L1.predict(temp_fcast)
71. y_predictedM2L1 = np.concatenate(y_predictedM2L1)
72. dANNM2L1 =
    pd.DataFrame({'Datefuture':(index_future_dates),'PuissanceANNM2L1':(y_pre
    dictedM2L1)})
73. dANNM2L1.set_index('Datefuture',inplace=True)
74. # 3 mois L1
75. modelM3L1 = keras.Sequential()
76. modelM3L1.add(keras.layers.Dense(units = 1, activation = 'linear',
    input_shape=[1]))
```

```
modelM3L1.compile(loss='mse', optimizer="adam")
modelM3L1.fit( TempM3, PowerM3_L1, epochs=100, verbose=1 )
y_predictedM3L1 = modelM3L1.predict(temp_fcast)
y_predictedM3L1 = np.concatenate(y_predictedM3L1)
dANNM3L1 =
   pd.DataFrame({'Datefuture':(index_future_dates),'PuissanceANNM3L1':(y_pre
   dictedM3L1)})
dANNM3L1.set_index('Datefuture',inplace=True)

plt.figure()
plt.figure(figsize=(20,9))
plt.subplot(2,1,1)
dANNS1L1['PuissanceANNS1L1'].plot()
dANND15L1['PuissanceANND15L1'].plot()
dANNM1L1['PuissanceANNM1L1'].plot()
dANNM2L1['PuissanceANNM2L1'].plot()
dANNM3L1['PuissanceANNM3L1'].plot()
(data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000).plot()
plt.legend(('1 semaine', '15 jours','1 mois','2 mois','3
   mois','Données'), fontsize=16)
plt.ylabel('Puissance (MW)', fontsize=16)
plt.title('Prévision de consommation pour les prochaines 24 heures pour
   la ligne 1 avec ANN', fontsize=16)
plt.subplot(2,1,2)
temp_fcast.plot()
plt.ylabel('Température (°C)', fontsize=16)
plt.xlabel('Temps (Heure)', fontsize=16)
plt.show()

#Ligne 1
print('MAE_1_Week_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dANNS1L1['PuissanceANNS1L1'])))
print('MAE_15_days_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dANND15L1['PuissanceANND15L1'])))
print('MAE_1_month_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dANNM1L1['PuissanceANNM1L1'])))
print('MAE_2_month_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dANNM2L1['PuissanceANNM2L1'])))
print('MAE_3_month_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dANNM3L1['PuissanceANNM3L1'])))
(data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000).mean()

```

Le script Pyhthon pour la prévision avec la régression linéaire pour la ligne 1 (1 février) :

```
1. import numpy as np
2. import matplotlib.pyplot as plt
3. import pandas as pd
4. import datetime as dt
5. import requests
6. from datetime import datetime
7. from statsmodels.tsa.arima_model import ARMA
8. from scipy.interpolate import interp1d
9. from sklearn.datasets import make_regression
10. from sklearn.linear_model import LinearRegression
11. from sklearn import datasets, linear_model
12. from sklearn.metrics import*
13. 
14. data=pd.read_excel('DataSetBC.xlsx', index_col='Time',parse_dates=True)
15. 
16. temp_fcast = data['2020-02-01 00:00:00':'2020-02-02
    00:00:00']['Tseche'].values.reshape(-1,1)
17. index_future_dates=data['2020-02-01 00:00:00':'2020-02-02
    00:00:00'].index
18. 
19. ### Ligne 1
20. # 1 Semaine L1
21. S1L1 = pd.DataFrame({'puissancel1':(data['2020-01-01 00:00:00':'2020-01-
    07 00:00:00']['PuissancekW_L1']/1000),'temperature':(data['2020-01-01
    00:00:00':'2020-01-07 00:00:00']['Tseche'])})
22. PowerS1_L1=S1L1['puissancel1'].values.reshape(-1,1)
23. TempS1=S1L1['temperature'].values.reshape(-1,1)
24. # 15 jours L1
25. D15L1 = pd.DataFrame({'puissancel1':(data['2020-01-01 00:00:00':'2020-01-
    15 00:00:00']['PuissancekW_L1']/1000),'temperature':(data['2020-01-01
    00:00:00':'2020-01-15 00:00:00']['Tseche'])})
26. PowerD15_L1=D15L1['puissancel1'].values.reshape(-1,1)
27. TempD15=D15L1['temperature'].values.reshape(-1,1)
28. # 1 mois   L1
29. M1L1 = pd.DataFrame({'puissancel1':(data['2020-
    01']['PuissancekW_L1']/1000),'temperature':(data['2020-01']['Tseche'])})
30. PowerM1_L1=M1L1['puissancel1'].values.reshape(-1,1)
31. TempM1=M1L1['temperature'].values.reshape(-1,1)
32. # 2 mois   L1
33. M2L1 = pd.DataFrame({'puissancel1':(data['2020-01':'2020-
    02']['PuissancekW_L1']/1000),'temperature':(data['2020-01':'2020-
    02']['Tseche'])})
34. PowerM2_L1=M2L1['puissancel1'].values.reshape(-1,1)
35. TempM2=M2L1['temperature'].values.reshape(-1,1)
36. # 3 mois   L1
```

```
M3L1 = pd.DataFrame({'puissancel1':(data['2020-01':'2020-03']['PuissancekW_L1']/1000),'temperature':(data['2020-01':'2020-03']['Tseche'])})
PowerM3_L1=M3L1['puissancel1'].values.reshape(-1,1)
TempM3=M3L1['temperature'].values.reshape(-1,1)

###Ligne 1
# 1 Semaine L1
regressorS1L1 = LinearRegression()
regressorS1L1.fit(TempS1, PowerS1_L1)
y_predS1L1 = regressorS1L1.predict(temp_fcast)
y_predS1L1 = np.concatenate(y_predS1L1)
dRLS1L1 = pd.DataFrame({'Datefuture':(index_future_dates),'puissanceRLS1L1':(y_predS1L1)})
dRLS1L1.set_index('Datefuture',inplace=True)
# 15 jours L1
regressorD15L1 = LinearRegression()
regressorD15L1.fit(TempD15, PowerD15_L1)
y_predD15L1 = regressorD15L1.predict(temp_fcast)
y_predD15L1 = np.concatenate(y_predD15L1)
dRLD15L1 = pd.DataFrame({'Datefuture':(index_future_dates),'puissanceRLD15L1':(y_predD15L1)})
dRLD15L1.set_index('Datefuture',inplace=True)
# 1 mois   L1
regressorM1L1 = LinearRegression()
regressorM1L1.fit(TempM1, PowerM1_L1)
y_predM1L1 = regressorM1L1.predict(temp_fcast)
y_predM1L1 = np.concatenate(y_predM1L1)
dRLM1L1 = pd.DataFrame({'Datefuture':(index_future_dates),'puissanceRLM1L1':(y_predM1L1)})
dRLM1L1.set_index('Datefuture',inplace=True)
# 2 mois   L1
regressorM2L1 = LinearRegression()
regressorM2L1.fit(TempM2, PowerM2_L1)
y_predM2L1 = regressorM2L1.predict(temp_fcast)
y_predM2L1 = np.concatenate(y_predM2L1)
dRLM2L1 = pd.DataFrame({'Datefuture':(index_future_dates),'puissanceRLM2L1':(y_predM2L1)})
dRLM2L1.set_index('Datefuture',inplace=True)
# 3 mois   L1
regressorM3L1 = LinearRegression()
regressorM3L1.fit(TempM3, PowerM3_L1)
y_predM3L1 = regressorM3L1.predict(temp_fcast)
y_predM3L1 = np.concatenate(y_predM3L1)
dRLM3L1 = pd.DataFrame({'Datefuture':(index_future_dates),'puissanceRLM3L1':(y_predM3L1)})
dRLM3L1.set_index('Datefuture',inplace=True)

plt.figure()
plt.figure(figsize=(20,9))
```

```
plt.subplot(2,1,1)
dRLS1L1['puissanceRLS1L1'].plot()
dRLD15L1['puissanceRLD15L1'].plot()
dRLM1L1['puissanceRLM1L1'].plot()
dRLM2L1['puissanceRLM2L1'].plot()
dRLM3L1['puissanceRLM3L1'].plot()
(data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000).plot()
plt.legend(('1 semaine', '15 jours','1 mois','2 mois','3
   mois','Données'), fontsize=16)
plt.ylabel('Puissance (MW)', fontsize=16)
plt.title('Prévision de consommation pour les prochaines 24 heures pour
   la ligne de distribution 1 avec RL', fontsize=16)
plt.subplot(2,1,2)
data['2020-02-01 00:00:00':'2020-02-02 00:00:00']['Tseche'].plot()
plt.ylabel('Température (°C)', fontsize=16)
plt.xlabel('Temps (Heure)', fontsize=16)
plt.show()

#Ligne 1
print('MAE_1_Week_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dRLS1L1['puissanceRLS1L1'])))
print('MAE_15_days_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dRLD15L1['puissanceRLD15L1'])))
print('MAE_1_month_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dRLM1L1['puissanceRLM1L1'])))
 print('MAE_2_month_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dRLM2L1['puissanceRLM2L1'])))
 print('MAE_3_month_L1=',(mape((data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000),dRLM3L1['puissanceRLM3L1'])))
 (data['2020-02-01 00:00:00':'2020-02-02
   00:00:00']['PuissancekW_L1']/1000).mean()

```