UMONS
Université de Mons

Maxime GLOESENER

Projet Streaming Data Analysis

Analyse d'une série chronologique

POLYTECH MONS

14/01/2022

ACADÉMIE UNIVERSITAIRE WALLONIE-BRUXELLES

Table des matières

1. Introduction

Ce rapport a pour objectif d'expliquer le raisonnement qui a été suivi dans le cadre du projet réalisé pour le cours de Streaming Data Analysis.

Le but de ce projet est d'analyser une série chronologique grâce aux différents outils qui ont été vus en cours.

Ce rapport se divise en plusieurs parties :

- Traitement des données et visualisation ;
- Analyse descriptive des données ;
- Transformations des données pour rendre la série stationnaire ;
- Application de la méthode de Box et Jenkins pour réaliser une modélisation SARIMA de la série ;
- Calcul des prédictions sur un jeu de données de test ;
- Analyse critique des résultats obtenus ;
- Ecriture de l'équation théorique du modèle SARIMA choisi ;
- Autres méthodes de modélisation SARIMA.

Le raisonnement pour chaque étape sera expliqué et des représentations graphiques viendront illustrer nos propos.

Il est important de noter que le projet a été réalisé dans un jupyter notebook. Ce type de format permet d'ajouter des commentaires au format markdown entre chaque cellule pour expliquer ce qui est réalisé. De ce fait, la lecture du code est facilitée car chaque étape est expliquée, ce rapport vient s'ajouter au code pour expliciter certains aspects théoriques non abordés dans le code.

Le dossier déposé sur moodle contient le fichier Excel avec les données utilisés lors de ce projet, le code python sous format jupyter notebook, un dossier avec des images reprenant la plupart des illustrations présentes dans ce rapport et ce rapport.

2. Traitement des données

Le jeu de données utilisé pour ce projet provient du célèbre site Kaggle et peut être retrouvé en suivant ce lien : https://www.kaggle.com/saurav9786/time-series-data?select=TractorSales.csv

Le nom du dataset est « TractorsSales.csv ».

A première vue, les données ne sont pas sous un format adéquat à l'analyse que nous voulons en faire. Nous allons donc commencer par un prétraitement de celles-ci.

Voici à quoi ressemblent les données initiales :

Month-Year	Number of Tractor Sold
3-Jan	141
3-Feb	157
3-Mar	185
3-Apr	199
3-May	203
3-Jun	189
3-Jul	207

Figure 1 Données initiales

Les données correspondent à la vente de tracteurs dans une entreprise entre 2003 et 2014.

Après un traitement des données, nous avons des données bien ordonnées dans un dataframe pour pouvoir appliquer tous nos outils d'analyse dessus.

	Vente-Tracteurs
2003-01-01	141
2003-02-01	157
2003-03-01	185
2003-04-01	199
2003-05-01	203
2003-06-01	189
2003-07-01	207
2003-08-01	207
2003-09-01	171
2003-10-01	150

Figure 2 Données obtenues après le traitement

3. Visualisation des données

Maintenant que nous avons traité les données, nous pouvons les visualiser.

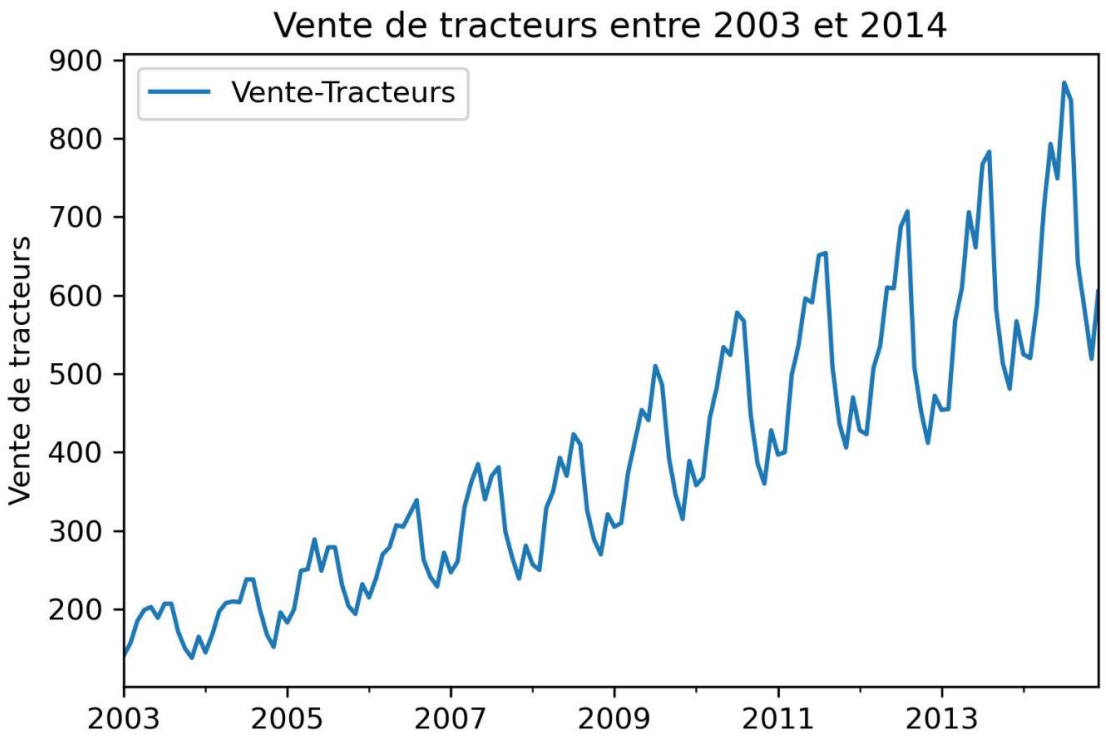

Figure 3 Visualisation des données

Voici l'évolution de la vente de tracteurs entre 2013 et 2014 pour une entreprise.

4. Analyse descriptive

La première étape de visualisation est très importante car elle nous permet déjà de réaliser une première analyse de la série chronologique que nous étudions.

En effet, à vu d'œil, on peut remarquer que la série a une forte tendance et qu'il y a une saisonnalité. De plus, on peut aussi identifier l'augmentation dans l'amplitude des pics avec le temps. Cette augmentation d'amplitude est synonyme d'une variance non stationnaire et nous permet de dire que c'est un modèle multiplicatif. Il faudra par la suite appliquer une transformation logarithmique sur le modèle pour stabiliser la variance et rendre le modèle additif lors de l'analyse.

Pour confirmer nos observations, nous pouvons réaliser une décomposition de la série en utilisant la fonction seasonal_decompose(). Cela va nous permettre de mieux visualiser la tendance, la saisonnalité et les résidus.

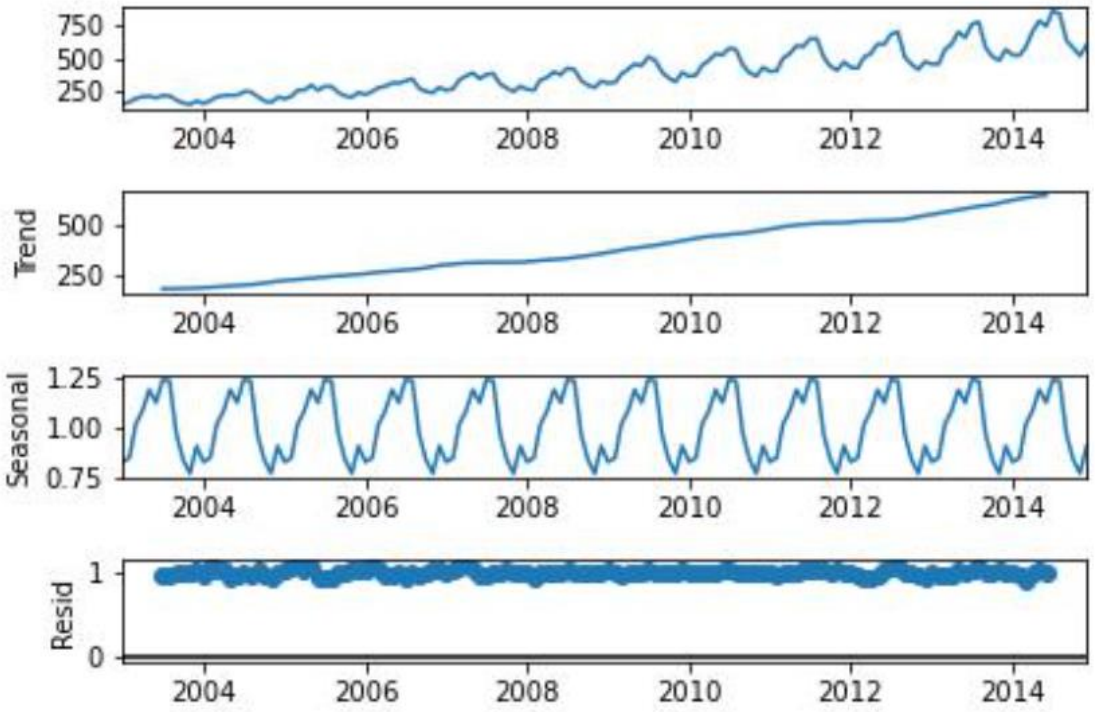

Figure 4 Décomposition de la série

Comme attendu, la décomposition de la série nous montre une tendance linéaire croissante, une saisonnalité (avec s = 12 mois) et une non-stationnarité.

5. Transformations sur la série initiale

Nous savons que pour pouvoir analyser les autocorrélations et autocorrélations partielles de la série, nous devons la rendre stationnaire étant donné que tous les outils vus en cours se basent sur des séries stationnaires en moyenne, en variance et en covariance. Nous allons donc appliquer différentes transformations à la série avec pour objectif de la rendre stationnaire.

Comme nous travaillons avec un modèle multiplicatif, la première étape est de le rendre additif. Pour cela, comme nous l'avons vu en cours, nous allons appliquer une transformation logarithmique sur la série. Cela a pour objectif de stabiliser la variance et donc par conséquent, l'amplitude entre les pics ne devrait plus augmenter mais se stabiliser et devrait devenir constante.

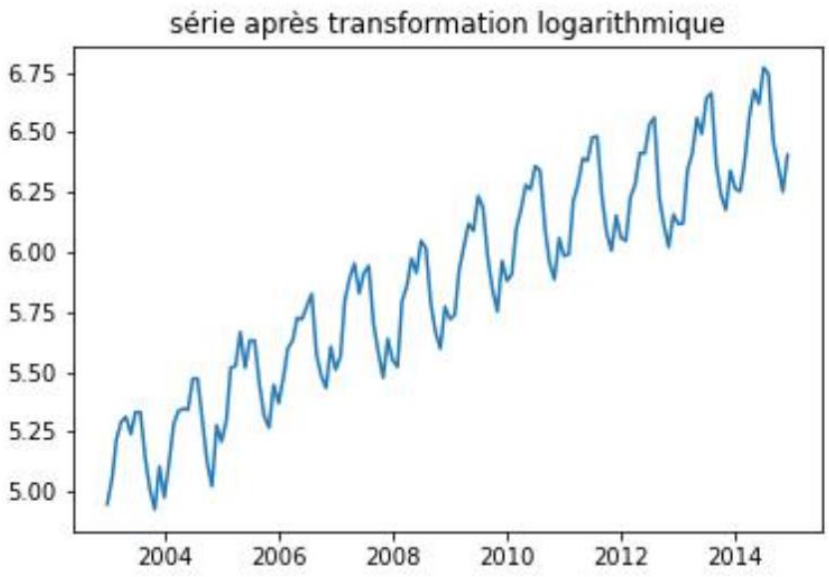

Figure 5 Série après la transformation logarithmique

Voici la série obtenue après la transformation logarithmique. Comme attendu, on voit bien que l'augmentation d'amplitude des pics a bien été stabilisée.

Nous savons qu'un processus X_t est considéré comme stationnaire au sens faible si 3 conditions sont respectées, à savoir :

Au sens faible

$\{X_t\}$ est stationnaire au sens faible ("stationnarité du second ordre") si

$$\left.\begin{array}{l}\mathbb{E}(X_t) = \mu \\ \text{Var}(X_t) = \sigma_x^2 \\ \text{Cov}(X_t, X_{t+k}) = \gamma_k\end{array}\right\} \quad \text{indépendants de } t \quad (\forall t,\ \forall k).$$

Figure 6 Stationnarité faible

Dans le cas de notre série, nous avons pu corriger la non-stationnarité en variance, mais il reste toujours le problème de la tendance qu'il faudrait éliminer. Pour cela, il faut mettre en pratique le premier point de la définition de stationnarité faible et il faut stabiliser la moyenne.

Pour cela, nous allons utiliser un outil vu en cours qui nous permet de stationnariser une série instationnaire en moyenne. Cet outil est la différentiation.

Nous appliquons un filtre différence première à la série.

Voici l'équation de ce type de filtre :

$$y_t = \nabla x_t = x_t - x_{t-1}$$

Figure 7 Filtre différence première

Ce filtre a pour objectif d'éliminer les tendances linéaires.

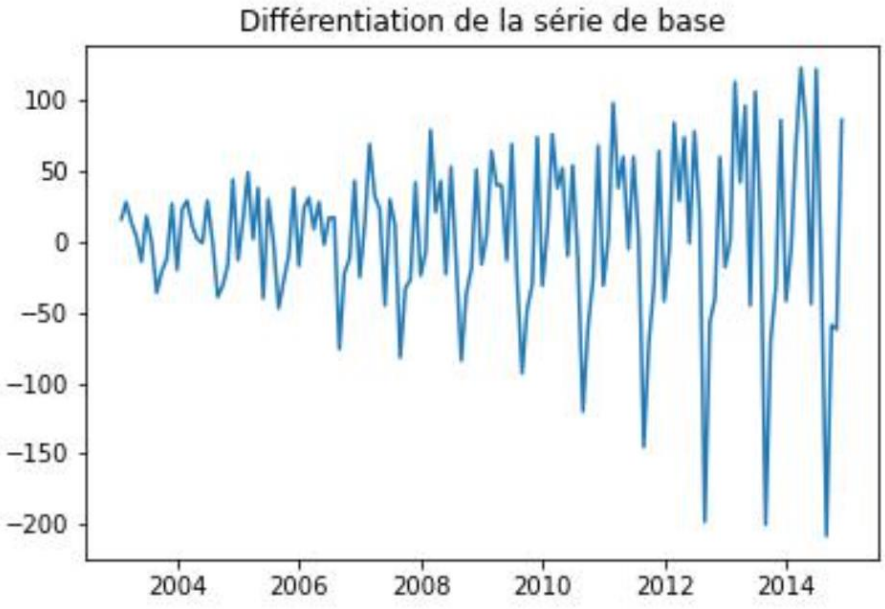

Figure 8 Différentiation de la série de base

Voici la série de base qui a été différentiée une fois. La tendance linéaire a bien été éliminée, cependant, la saisonnalité reste présente et sa variance augmente avec le temps, ce qui n'est pas idéal.

Nous remarquons que grâce à la différentiation, nous avons pu éliminer la tendance. Nous avons donc maintenant une série stable en moyenne mais toujours instationnaire en variance car on voit bien que la variance de la série augmente avec le temps.

L'idée est donc maintenant d'appliquer les deux méthodes vues avant pour stationnariser notre série. Nous allons commencer par appliquer une transformation logarithmique qui sera suivie par une différentiation d'ordre 1.

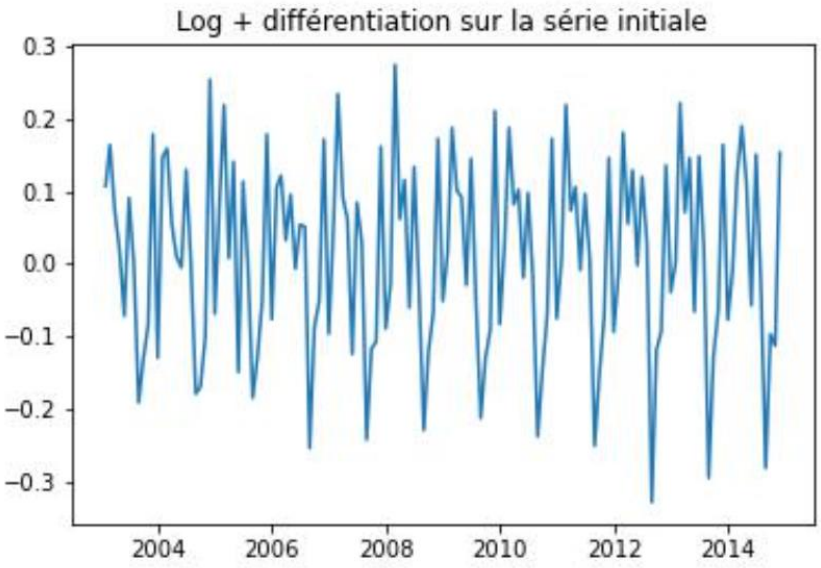

Figure 9 Série log et différentiation

Voici la série obtenue lorsque nous appliquons les deux transformations à la suite (log de la série et filtre D (-1)). Nous remarquons que la série est bien stationnarisée en moyenne, la tendance linéaire a été éliminée et en variance mais nous ne pouvons pas être certain de sa stationnarité à vue d'œil.

Pour analyser la stationnarité de la série, nous allons utiliser le test de Dicky-Fuller. Les résultats de ce test nous donnent :

```
1. ADF :  -2.680467119996607
2. P-Value :  0.07747972836508323
3. Num Of Lags :  14
4. Num Of Observations Used For ADF Regression and Critical Values Calculation : 128
5. Critical Values :
         1% :  -3.4825006939887997
         5% :  -2.884397984161377
         10% :  -2.578960197753906
```

Figure 10 Résultats du test de stationnarité de Dickey-Fuller

Etant donné que la p-value est légèrement supérieure à 0.05, nous ne pouvons pas être sûr avec 95% de confiance que la série est stationnaire. Cela peut s'expliquer en analysant visuellement la série obtenue après les deux transformations.

Nous pouvons identifier une certaine saisonnalité dans la répétition des pics. Pour stationnariser la série, nous allons donc effectuer une différentiation saisonnière (avec s = 12).

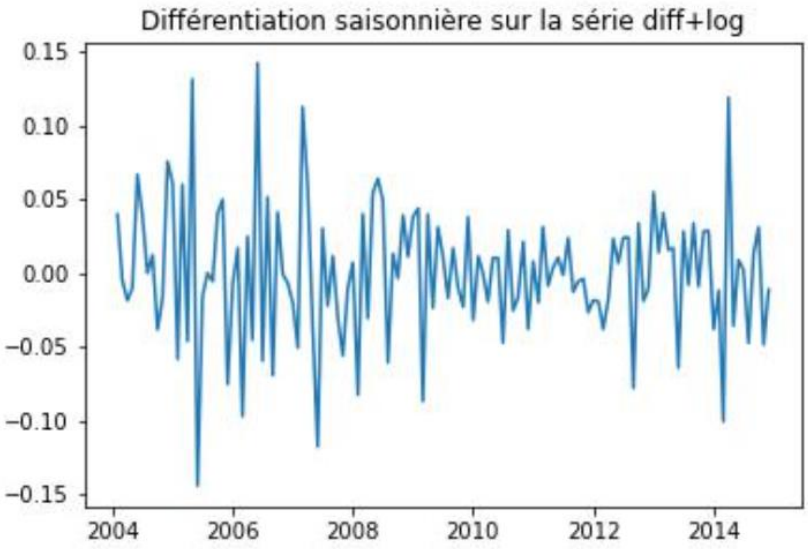

Figure 11 Différentiation saisonnière sur la série différentiée et log

Voici la série obtenue lorsqu'on lui a appliqué la transformation logarithmique suivie d'une différentiation et d'une différentiation saisonnière avec s qui vaut 12. La différentiation saisonnière nous a permis d'éliminer la saisonnalité.

Nous pouvons refaire un test de Dickey-Fuller sur cette série pour s'assurer de sa stationnarité. Celui-ci nous confirme que la série est bien stationnaire car sa p-value est largement inférieure à 0.05.

```
1. ADF :  -4.4809622221427
2. P-Value :  0.0002128161041120004
3. Num Of Lags :  12
4. Num Of Observations Used For ADF Regression and Critical Values Calculation : 118
5. Critical Values :
        1% :  -3.4870216863700767
        5% :  -2.8863625166643136
        10% :  -2.580009026141913
```

Figure 12 Test de stationnarité de Dickey-Fuller

Maintenant que nous avons appliqué des transformations sur la série pour la rendre stationnaire, nous pouvons utiliser les différents outils d'analyse du ACF et PACF pour construire le modèle SARIMA.

6. Analyse ACF / PACF pour la modélisation Box et Jenkins

Voici les graphiques d'autocorrélation et d'autocorrélation partielles obtenus sur la série stationnarisée. Ces graphiques permettent de montrer la corrélation entre la série initiale et la série décalée dans le temps. L'objectif est d'analyser ces résultats pour identifier les coefficients du modèle SARIMA.

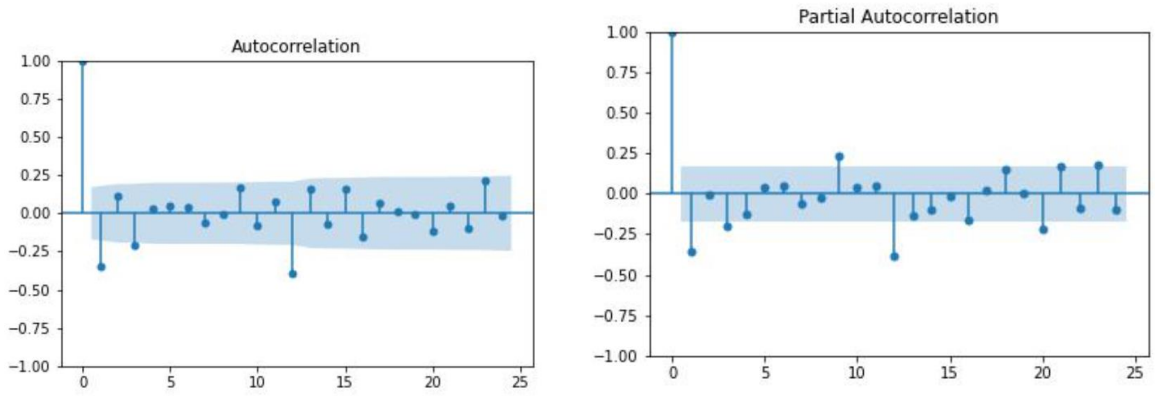

Figure 13 ACF et PACF de la série stationnarisée

Voici les résultats de l'ACF et PACF de la série stationnarisée. Nous savons qu'un modèle SARIMA doit prendre en compte 7 paramètres : (p, d, q) (P, D, Q) m

Avec :

- d = degré de différentiation
- D = ordre différentiation de la saisonnalité
- p = ordre pour les termes AR
- q = ordre pour les termes MA
- Q = ordre pour les termes MA saisonniers
- P = ordre pour les termes AR saisonniers
- m = période

P, D, Q = paramètres de saisonnalité alors que p, d, q sont les paramètres de tendance !

On sait que dans notre cas, m vaut 12. Il nous reste maintenant à identifier les autres termes en analysant les ACF et PACF. Comme nous le savons, l'ACF va nous donner des informations sur les termes MA (q,Q) et la PACF va nous donner des informations sur les termes AR(p,P).

La méthode de Box et Jenkins a pour but de commencer par un modèle simple et d'ensuite augmenter l'ordre des composantes si nécessaire. Cela s'explique par le fait qu'on veut éviter un overfitting sur les données d'entraînement. Si notre modèle est overfitté, il va commencer à prédire les bruits blancs sur les données d'entraînement et cela va, par conséquent, entraîner des erreurs de prévision sur les données de test.

Nous commençons donc par un modèle simple avec d = 1, q = 1, P = 1, Q = 1 et les autres paramètres à 0.

Pourquoi avons-nous choisi ces paramètres ?

En analysant le ACF, cela nous permet d'obtenir des informations sur les composantes MA du modèle et nous pouvons remarquer un batônnet significativement en-dehors de l'intervalle de confiance en 1. Cela nous donne donc q=1. De plus, de l'analyse de l'ACF, nous remarquons aussi un batônnet significativement non nul en 12, cela s'expliquer par la saisonnalité de la série. Nous prenons donc un Q qui vaut 1 car Q est la composante de saisonnalité pour un MA.

Maintenant que nous avons identifié les termes MA, nous pouvons analyser la PACF pour identifier les termes AR. Nous remarquons un batônnet significativement différent de zéro en 12, ce qui correspond à la saisonnalité. Nous avons donc P = 1, la composante de saisonnalité pour le cas AR.

Pour finir, comme nous avons différentié la série pour stabiliser la moyenne, nous mettons le terme de différentiation d à 1.

Maintenant que les paramètres ont été choisi, nous pouvons séparer les données en jeu d'entraînement et jeu de test. Nous prenons 132 données dans le jeu d'entraînement et 12 dans le jeu de test ce qui correspond à environ 10% du jeu de données comme vu dans le cours.

Nous pouvons instancier le modèle avec les différents paramètres et l'entrainer sur la série qui a subi la transformation logarithmique.

Après avoir analysé les résultats de ce modèle, comme le veut la méthode de Box et Jenkins, nous testons un modèle d'ordre un peu plus élevé.

Nous passons d'un modèle (0, 1, 1) * (1, 0, 1)12 à un modèle (1, 1, 1) * (1, 0, 1)12 et nous analysons les paramètres de ce modèle. Nous remarquons que le terme MA = q est proche de 0 et donc négligeable, pour éviter de faire de l'overfitting, nous choisissons aux final de conserver le modèle sélectionné au début.

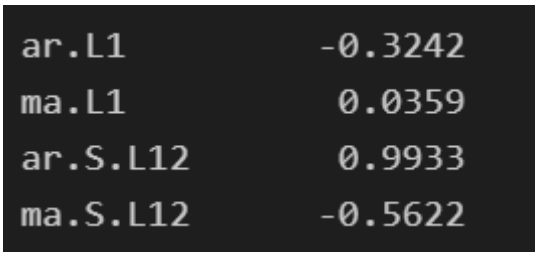

Figure 14 Paramètres modèle 2

7. Evaluation des résultats

Pour étudier la qualité d'un modèle, plusieurs possibilités s'offrent à nous :

- Établir un graphique des résidus en fonction du temps t et vérifier qu'il n'y a pas de tendance ni de variation de la variance ;
- Vérifier la normalité des résidus et faire un test de Shapiro-Wilks ;
- Etudier la non-corrélation des résidus ;
- Examiner les corrélogrammes (ACF et PACF) des résidus.

Nous allons donc réaliser ces différentes techniques pour vérifier la qualité du modèle SARIMA.

Premièrement, nous pouvons utiliser la fonction plot_diagnostic() qui va nous apporter des informations intéressantes sur les résidus :

- Les résidus standardisés varient dans un domaine [-2, 2] et ne présentent pas de tendance marquée, ni de variation de la variance ;
- La distribution suit une loi normale, on peut le voir sur l'histogramme et sur le diagramme quantile-quantile ;
- L'autocorrélation du résidu est bien un bruit blanc comme voulu.

Les principes d'évaluation d'un modèle cités au-dessus sont donc vérifiés et permettent de nous conforter dans la qualité du modèle.

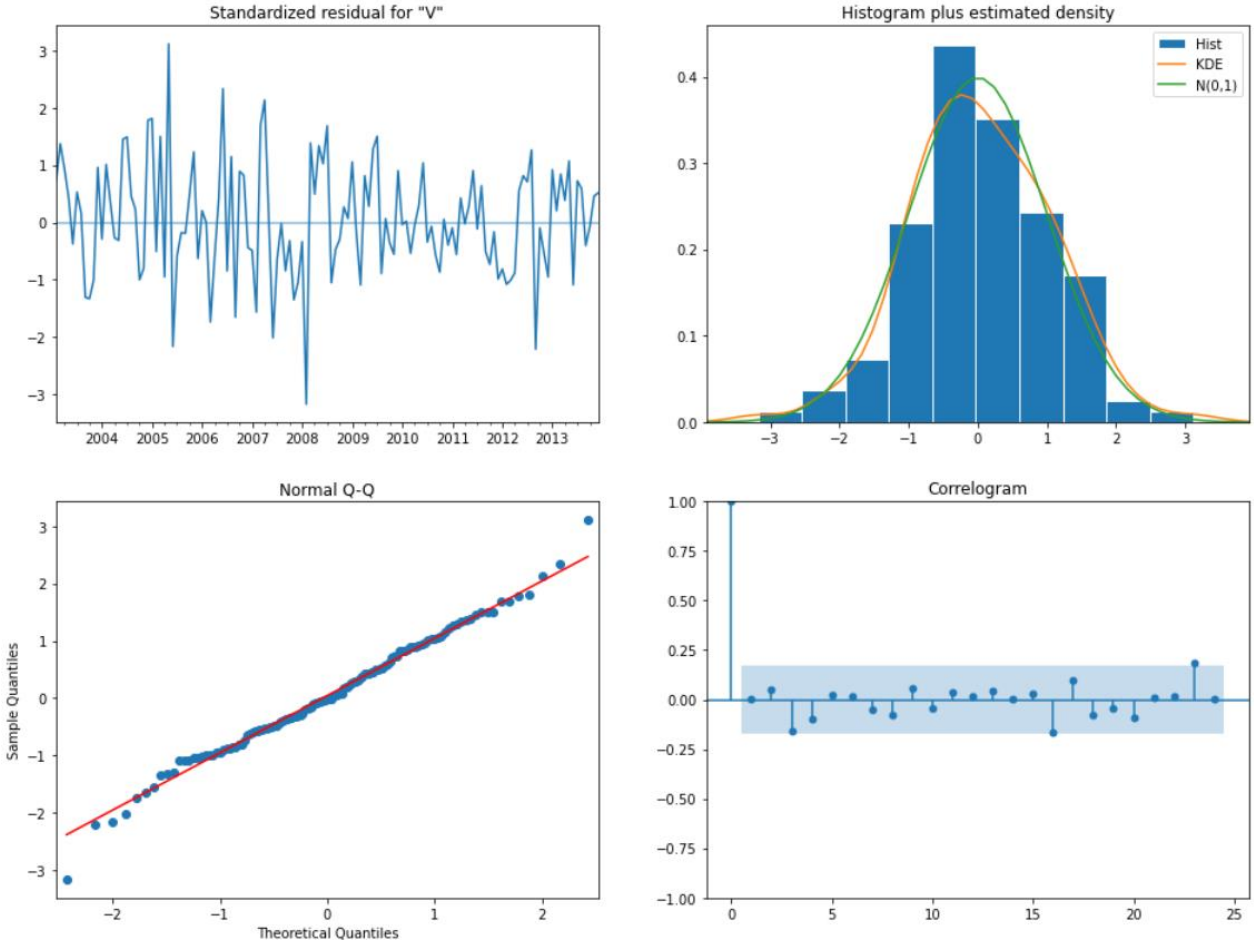

Figure 15 Etude des résidus

En plus de cela, nous pouvons vérifier les hypothèses avec des tests statistiques.

- Test de Ljung-Box : vérifie la non-corrélation entre les résidus. La p-value est supérieure à 0.05, ce qui confirme la non-corrélation des résidus ;

- Test de Shapiro : vérifie la normalité des résidus. De même, la p-value est inférieure à 0.05, on peut donc accepter l'hypothèse de normalité des résidus.

Pour évaluer la qualité d'un modèle, nous utilisons aussi des indicateurs comme la MSE qui est calculée sur la série log et sur la série initiale ou encore la MAE qui est calculée sur la série initiale.

8. Prédictions du modèle

Dans cette partie, nous allons étudier les prédictions du modèle.

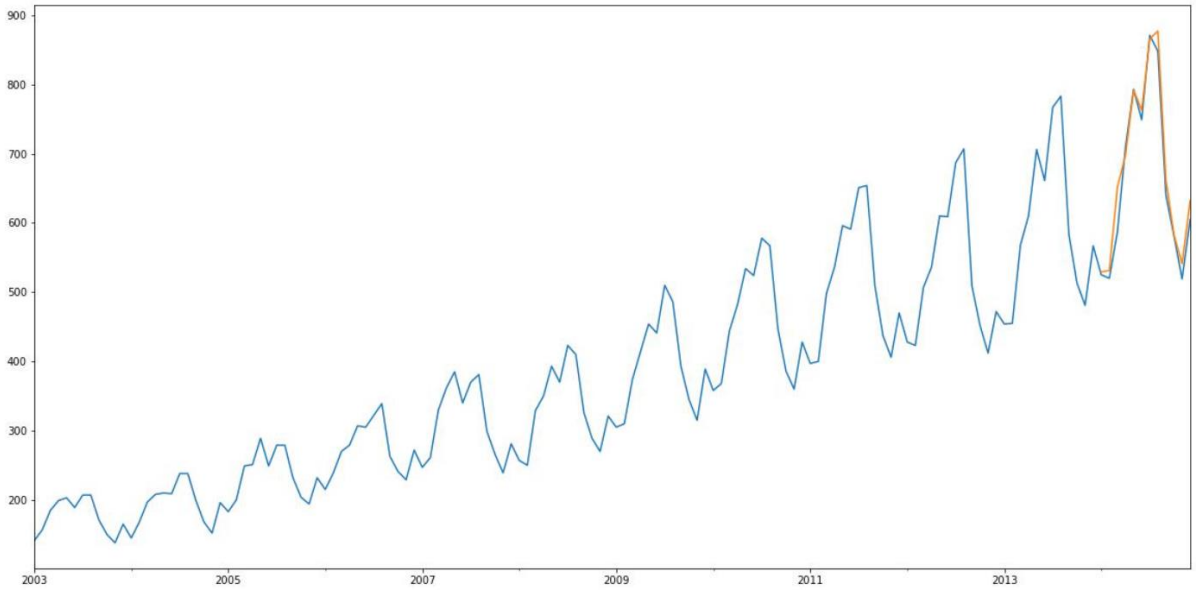

Figure 16 Prédictions du modèle sur les données de test

Voici les prédictions obtenues sur les données de test.

Etant donné que nous avons réalisé le modèle sur la série qui a subi la transformation logarithmique, pour repasser aux « vraies données », nous devons inverser cette transformation et par conséquent, prendre l'exponentielle de la prédiction obtenue. Cela va entraîner un biais dans la prédiction qui sera expliqué dans l'analyse critique des résultats obtenus.

9. Analyse critique des résultats

Un principe important qui a été vu en cours, est que les prévisions optimales au sens des moindres carrés le sont pour la série transformée et, en général, elles ne sont pas optimales pour la série initiale.

Dans notre cas, comme nous faisons des prédictions sur la série transformée logarithmiquement, pour avoir les prédictions sur la série initiale, il ne suffit pas de prendre l'exponentielle des prédictions sur la série transformée, mais il faut prendre en compte un facteur d'ajustement de biais.

Pour une série transformée logarithme :

La prévision optimale de la série initiale n'est pas $\exp[\widehat{X}_t(l)]$ mais :

$$\exp[\widehat{X}_t(l) + \frac{1}{2} var(e_t(l))]$$

Ce principe est expliqué clairement dans cet article : https://otexts.com/fpp2/transformations.html qui nous a permis de bien comprendre la raison de ce biais. Cela s'explique par le fait que la transformation logarithmique n'est pas linéaire.

En effet : E[exp(y)] != exp(E[y]). Ce qui donne par l'inégalité de Jensen : E[exp(y)] >= exp(E[y]). La moyenne d'une distribution log normal n'est donc pas simplement exp(μ) mais exp(μ+ σ^2 /2). On doit prendre en compte un facteur d'ajustement de biais.

A cause de la non-linéarité de la transformation logarithmique, lorsque nous prenons la transformation inverse, au lieu d'obtenir la moyenne de la prédiction, nous obtenons en général, la médiane. C'est pourquoi il faut corriger la prévision en ajoutant le terme ½ * variance de l'erreur de prédiction.

Dans la plupart des exemples lus dans la littérature, les gens ne prennent pas en compte ce biais soit par simplification (ils considèrent ce biais comme négligeable), soit car ils ne sont pas au courant.

Dans le cadre de notre projet nous avons essayé de prendre en compte ce biais d'ajustement mais cela rendait les prédictions moins bonnes. La qualité des prédictions a été estimée en utilisant la RMSE. Alors que lorsque nous faisions une soustraction de la variance de l'erreur de prévision, cela améliorait les prédictions. Après de longues recherches dans la littérature, nous n'avons pas trouvé d'explication convaincantes pour expliquer ce phénomène. Nous avons donc décidé de prendre simplement l'exponentielle des prédictions sur la série log pour obtenir les prédictions sur la série initiale car elle donnait de meilleurs résultats que les prédictions optimales qui tiennent compte du biais. Nous sommes cependant conscients de l'existence de ce biais mais nous n'avons pas réussi à améliorer la qualité des prédictions en le prenant en compte.

Pour améliorer la qualité de la prédiction, il serait intéressant de réussir à prendre en compte le biais de la transformation inverse sans détériorer les prédictions mais au contraire, en les améliorant !

Une autre piste d'amélioration du modèle est peut-être, de prendre des termes d'ordre plus élevé. Nous sommes restés sur des ordres 1 pour éviter un maximum l'overfitting et garder un modèle simple mais nous avons vu dans la littérature que certaines personnes pour des jeux de données simples (par exemple airPassengers) prenaient des modèles d'ordre beaucoup plus élevé. Nous pensons que cela risque de complexifier le modèle et de potentiellement amener de l'overfitting mais c'est une piste à explorer pour améliorer la qualité des prédictions.

10. Equation théorique

Nous avons un modèle SARIMA (0,1 ,1) * (1, 0, 1)12.

Les paramètres sont donc d = 1, q = 1, P = 1, Q = 1, s = 12

Son équation théorique est donc :

$$(1 - \Phi_p B^{12}) * (1 - B) * X_t = (1-\theta B) * (1 - \Theta B^{12}) * a_t$$

Les valeurs des différents paramètres sont connues car elles nous sont données lorsqu'on fit notre modèle. Si voulu, on pourrait remplacer les paramètres par leur valeur et expliciter l'équation en remplaçant les opérateurs de retard B par des X_{t-k}. (il suffit d'utiliser la relation B Xt = Xt-1 et de remplacer les paramètres par leur valeur si on veut obtenir l'équation explicite du modèle)

Voici les paramètres du modèle :

```
ma.L1      -0.307515
ar.S.L12    0.993371
ma.S.L12   -0.559736
sigma2      0.001304
dtype: float64
```

Figure 17 Paramètres modèle SARIMA

11. Autres méthodes de modélisation SARIMA

Dans cette partie, nous allons voir d'autres méthodes qui peuvent être utilisées pour identifier les paramètres optimaux d'un modèle SARIMA. Ces méthodes ont été réalisées dans le cadre du projet par curiosité et sont disponibles dans le notebook. Nous n'allons pas rentrer dans les détails dans ce rapport étant donné que ce n'est pas le but de ce projet et nous allons simplement expliquer les grands principes de ces méthodes.

La première méthode s'appelle « Grid Search » et consiste simplement à tester tous les paramètres possibles pour un modèle dans un intervalle donné. Dans notre cas, les différents paramètres se trouvent dans un intervalle [0, 2] et on va itérer sur toutes les possibilités possibles.

Cette technique utilise la AIC pour identifier la qualité d'un modèle et garder le meilleur modèle après les itérations. Etant donné que c'est une méthode itérative qui teste toutes les combinaisons possibles, elle prend environ 10 secondes à passer sur toutes les combinaisons.

La deuxième méthode qui peut être utilisée est la méthode autoarima. Cette méthode est encore plus simple que la première car c'est un package en python qui trouve les meilleurs paramètres pour nous. Après avoir lu pas mal de documentation sur cette méthode, on notera qu'en général, il est conseillé d'utiliser cette méthode au début d'un projet pour avoir rapidement un avis sur le modèle SARIMA et de pouvoir, par la suite, ajuster certains paramètres ou bien partir sur d'autres méthodes de prédictions qui pourraient être plus efficaces en fonction de la série chronologique que nous avons à modéliser. Cette méthode prend quant à elle, environ 7 secondes pour identifier les paramètres optimaux.

Ces deux méthodes ne donnent pas le même résultat. Cela est assez logique étant donné que la Grid Search utilise seulement la valeur AIC pour évaluer la qualité d'un modèle alors que la méthode autoarima utilise une combinaison de AIC et BIC pour évaluer la qualité du modèle.

La grid search nous donne un SARIMA (1, 1, 0) * (0, 1, 1)12 alors que la méthode autoarima nous donne un SARIMA (0, 1, 1) * (0, 1, 1)12.

12. Conclusion

En conclusion, nous avons pu mettre en pratique les différents outils vus en cours pour modéliser un SARIMA et analyser une série chronologique. Le modèle peut être utilisé pour prédire les ventes futures de tracteurs avec une certaine précision à court terme. Assez logiquement, plus on s'éloigne dans le temps, plus l'erreur de prédiction augmentera et la prédiction perdra donc en précision.