Projet - Séries Temporelles

Lilya-Nada Khelid, Alexis Solar, César Denerier, Antoine de Oliveira

Année 2023-2024

Table des matières

Introduction

L'électricité est un pilier fondamental de notre société moderne, alimentant nos foyers, nos entreprises et nos infrastructures essentielles. Comprendre les schémas de consommation d'électricité est crucial pour garantir la stabilité du réseau et planifier efficacement la production.

Dans le cadre de ce projet, nous nous pencherons sur une série temporelle particulière provenant de la base de données du Réseau de transport d'électricité (RTE). Cette série temporelle documente la consommation d'électricité en France métropolitaine (hors Corse) sur une période de 10 ans, du 1er janvier 2012 au 31 décembre 2021, totalisant ainsi 3653 valeurs.

L'objectif principal de notre projet est de développer un modèle de prédiction afin d'estimer la consommation d'électricité pour l'année 2022. Cette démarche nécessitera une approche systématique, et c'est là que l'utilisation du langage de programmation Python se révèle inestimable. En suivant une méthodologie bien définie, nous explorerons et présenterons la base de données RTE, effectuerons un traitement minutieux des données en utilisant diverses techniques adaptées à notre série temporelle, puis nous nous appuierons sur le modèle le plus adapté pour élaborer des prédictions significatives.

Le projet se déroulera en plusieurs étapes, comprenant la présentation détaillée de la base de données, l'analyse de la série temporelle, et enfin, l'analyse des résultats en fonction du modèle de prédiction. En combinant analyse de données et programmation informatique, notre approche holistique vise à fournir des résultats précis et fiables pour anticiper la consommation d'électricité en France pour l'année 2022. Ce projet offre ainsi une opportunité captivante d'explorer les nuances de la modélisation des séries temporelles tout en répondant à des enjeux pratiques du secteur de l'énergie.

1 Présentation des données

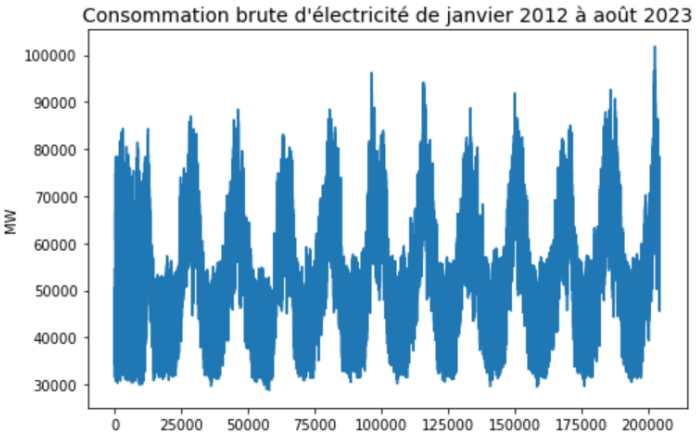
Référence du jeu de données utilisées :
- https://odre.opendatasoft.com *(voir bibliographie pour le lien complet)*

1.1 Données brutes :

Initialement, notre jeu de données incluait la consommation d'électricité et de gaz. Par conséquent, nous avons modifié cette base pour ne conserver que les variables "Date - Heure" et "Consommation brute d'électricité (MW) - RTE".

	Date - Heure	Consommation brute électricité (MW) - RTE
0	2023-08-31T23:30:00+02:00	42710
1	2023-08-31T23:00:00+02:00	43562
2	2023-08-31T22:30:00+02:00	44426
3	2023-08-31T22:00:00+02:00	43430
4	2023-08-31T21:30:00+02:00	43281
...	...	...
204523	2012-01-01T02:00:00+01:00	55531
204524	2012-01-01T01:30:00+01:00	56075
204525	2012-01-01T01:00:00+01:00	56230
204526	2012-01-01T00:30:00+01:00	58314
204527	2012-01-01T00:00:00+01:00	59610

[204528 rows x 2 columns]



Cette base initiale comprenait des enregistrements toutes les demi-heures sur une période de 11 ans, débutant le 1er janvier 2012 à minuit et se terminant le 31 août 2023 à 23h30, totalisant ainsi 204 528 valeurs. Afin de traiter ces données, il était nécessaire de convertir la colonne "Date - Heure" en objet "datetime", en prenant en compte le fuseau horaire UTC.

Ensuite, nous avons ajusté la colonne "Consommation brute d'électricité (MW) - RTE" afin d'obtenir la somme totale de la consommation brute pour chaque journée, en regroupant les données initiales par date.

Et pour finir, nous avons uniquement inclu les données allant du 1er janvier 2012 au 31 décembre 2021.

```
copy = data.copy()
copy['Date - Heure'] = pd.to_datetime(copy['Date - Heure'], utc=True)
```

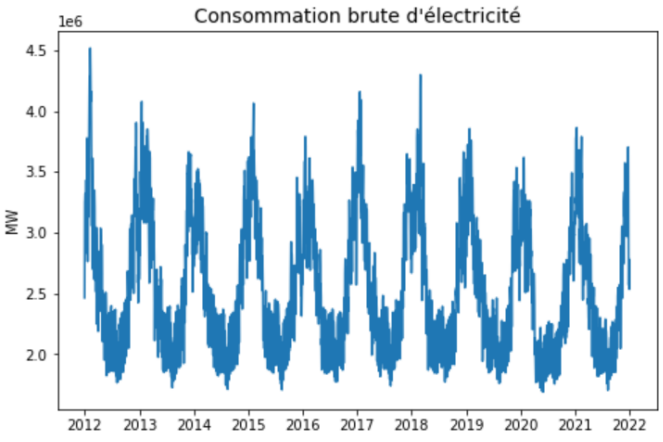
```
data_fi = copy.groupby(copy['Date - Heure'].dt.date)['Consommation brute électricité (MW) - RTE'].sum().reset_index()
```

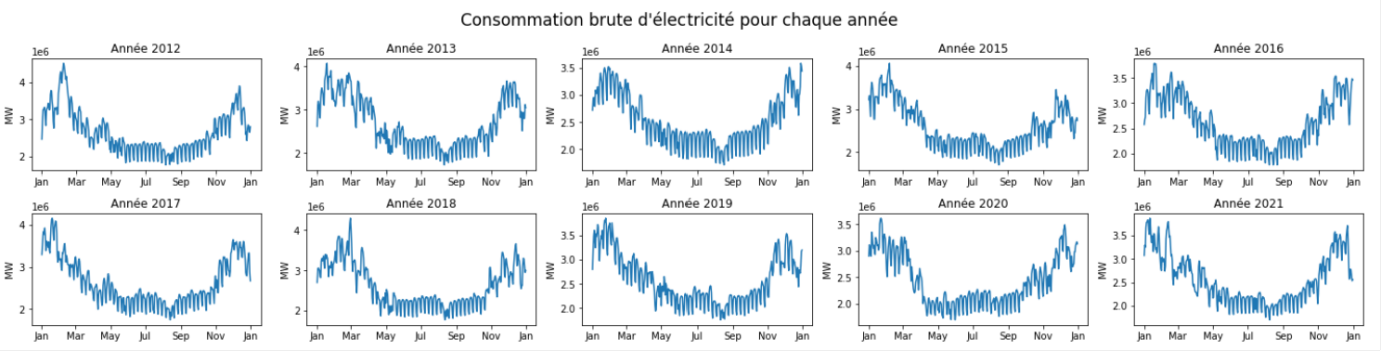
```
data_fi['Date - Heure'] = pd.to_datetime(data_fi['Date - Heure'])
date_debut = '2012-01-01'
date_fin = '2021-12-31'
data_final = data_fi.loc[(data_fi['Date - Heure'] >= date_debut) & (data_fi['Date - Heure'] <= date_fin)]
```

Ainsi, nous nous retrouvons avec une base de données contenant 3653 valeurs, réparties sur 10 ans :

	Date	Consommation brute d'électricité (MW)
1	2012-01-01	2463746
2	2012-01-02	2931616
3	2012-01-03	3253129
4	2012-01-04	3269333
5	2012-01-05	3318744
...	...	...
3649	2021-12-27	2781846
3650	2021-12-28	2761976
3651	2021-12-29	2657124
3652	2021-12-30	2534659
3653	2021-12-31	2559279

[3653 rows x 2 columns]





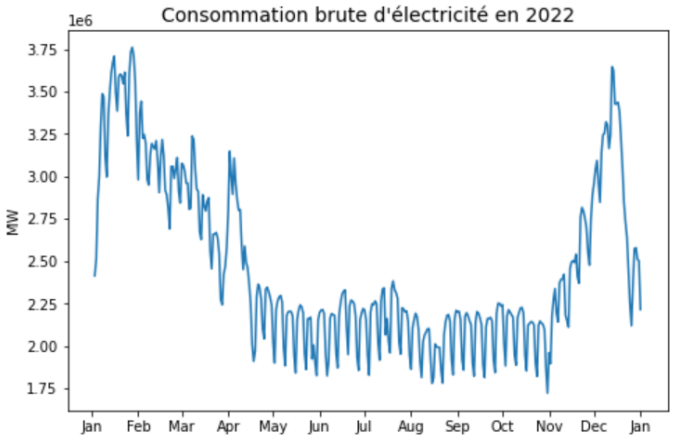
1.2 Données que nous souhaitons prédire :

Notons que notre objectif central demeure la prédiction de la consommation d'électricité pour l'année 2022, en nous appuyant sur les données accumulées au cours de la décennie précédente.

Afin d'évaluer la précision de nos prévisions, nous utiliserons les données officielles de la consommation d'électricité en 2022, qui comportent 365 valeurs, présentées ci-dessous :

	Date	Consommation brute d'électricité (MW)
3654	2022-01-01	2415921
3655	2022-01-02	2516642
3656	2022-01-03	2866781
3657	2022-01-04	2998008
3658	2022-01-05	3300587
...	...	...
4014	2022-12-27	2576548
4015	2022-12-28	2581792
4016	2022-12-29	2513400
4017	2022-12-30	2504314
4018	2022-12-31	2216417

[365 rows x 2 columns]



2 Analyse de la série temporelle

Dans cette partie, nous étudirons analytiquement notre série temporelle. Nous nous placerons dans le cas où notre série temporelle est univariée. Nous pouvons donc décomposer notre série, notée D_t, sous la forme d'une somme de trois termes :

$$D_t = f(t) + S_t + X_t, \qquad t \geq 1$$

où $f(t)$, la tendance, est une fonction de la forme $a + b.t$, S_t est la saisonnalité de période T (que nous chercherons à déterminer) et X_t est la stationnarité.
Notre objectif est donc de déterminer chacun de ces termes ce qui entrainera la prédiction de la consommation d'électricité de l'année 2022.

2.1 Tendance

Nous nous intéressons à la tendance. Pour se faire, nous utilisons la méthode de la régression linaire.

```
data_final_train = data_final

dp = DeterministicProcess(
    index=data_final_train["Consommation brute d'électricité (MW)"].index,
    constant=True,
    order=1,
    drop=True,
)

X = dp.in_sample()

y = data_final_train["Consommation brute d'électricité (MW)"]
model = LinearRegression(fit_intercept=False)
model.fit(X, y)

y_pred = pd.Series(model.predict(X), index=X.index)
```

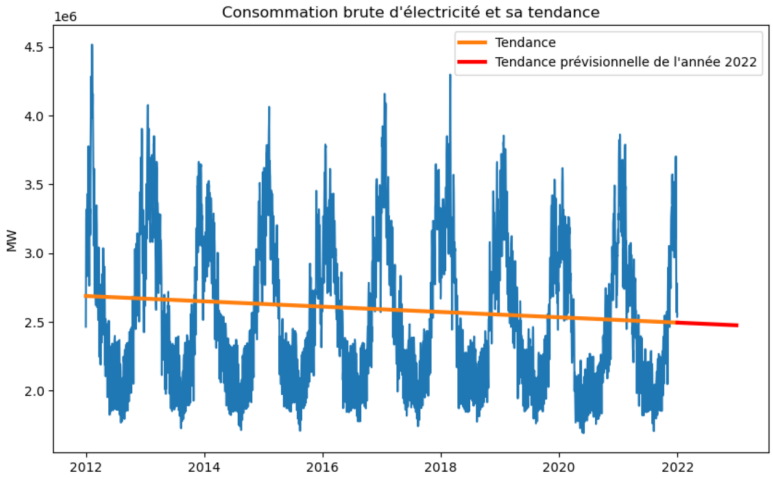
```
print('f(t) =', model.coef_[0], model.coef_[1],'*t')
```

```
f(t) = 2687894.7001258405 -53.03496096432459 *t
```

Nous obtenons ainsi $a = 2687894.7001258405$ et $b = -53.03496096432459$. Notre tendance s'écrit donc sous la forme :

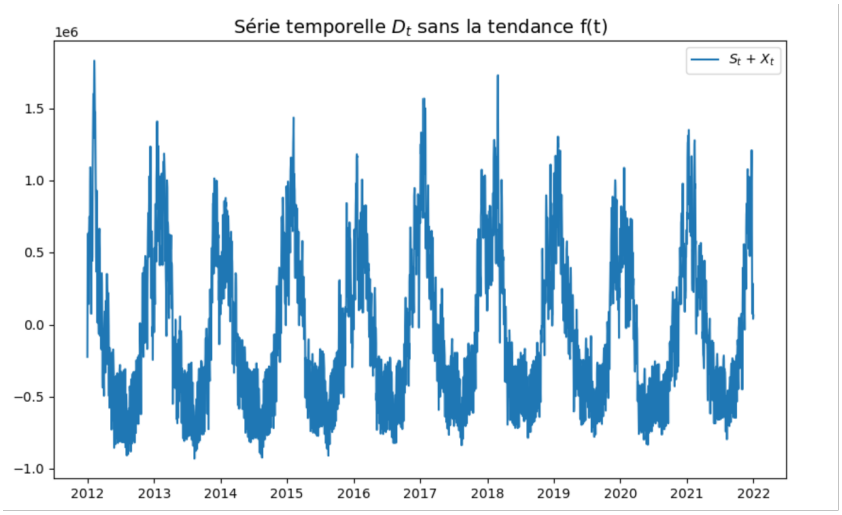
$$f(t) = 2687894.7001258405 - 53.03496096432459.t, \qquad t \geq 1$$

Ainsi nous pouvons représenter graphiquement la tendance $f(t)$ pour nos données allant de janvier 2012 à décembre 2021, ainsi que la tendance prévisionnelle pour l'année 2022 :



2.2 Saisonnalité

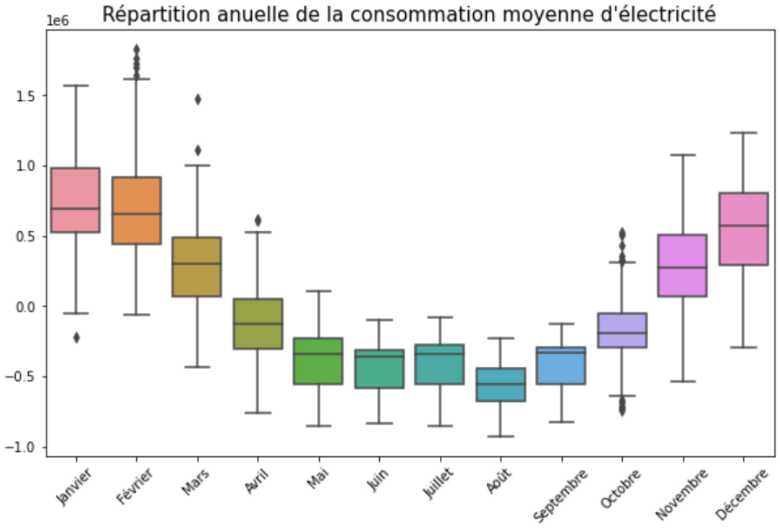
Après avoir déterminé la tendance $f(t)$ de notre série temporelle, nous nous intéressons à la saisonnalité, S_t. Pour cela, nous commençons par soustraire la tendance de notre série temporelle.



Notons cette série temporelle D_t sans la tendance $f(t)$: $D'_t = S_t + X_t$.

Puisque d'après le cours, la décomposition n'est pas unique, la période de S_t ne l'est pas non plus. Nous allons chercher à estimer une période pertinente.

Nous nous intéressons tout d'abord à une échelle annuelle, car cette période semble être la plus pertinente compte tenu des variations de température entre l'été et l'hiver, qui ont un impact considérable sur la consommation d'électricité. En calculant la moyenne, les quartiles et la médiane de la consommation d'électricité pour chaque année de nos données, nous pouvons représenter le boxplot ci-dessous :



D'après le graphique, nous observons une saisonnalité pertinente : nous prenons donc une période $T = 365$ jours. Nous noterons cette saisonnalité S'_t car nous ne savons toujours pas si cette saisonnalité sera suffisante ou non. Nous avons donc au t-ième jour de l'année :

$$\lim_{n\to+\infty} \quad \frac{1}{n}\sum_{k=1}^{n} S'_{t+kT} + X_{t+kT} = D_t - f(t)$$

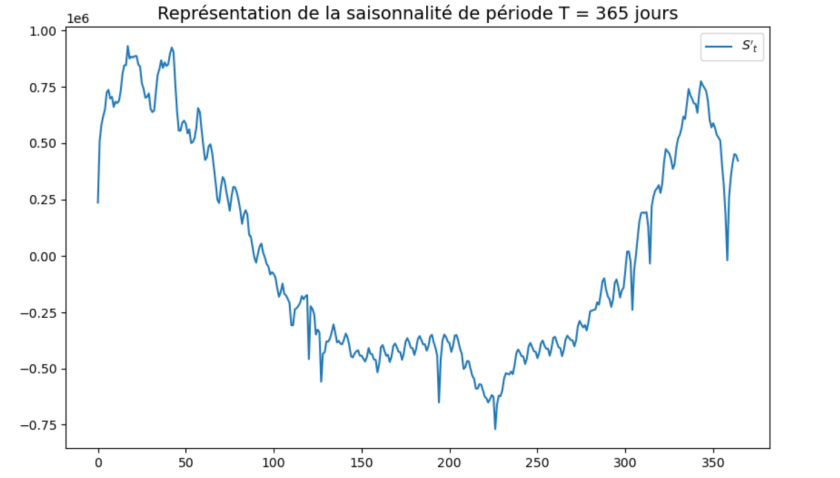
Or puisque par hypothèse $\forall k \in [\![0, n]\!]$, $S'_{t+kT} = S'_t$, ainsi nous avons :

$$\lim_{n\to+\infty} \quad S'_t + \frac{X_t + X_{t+T} + ... + X_{t+9T}}{n} = D_t - f(t)$$

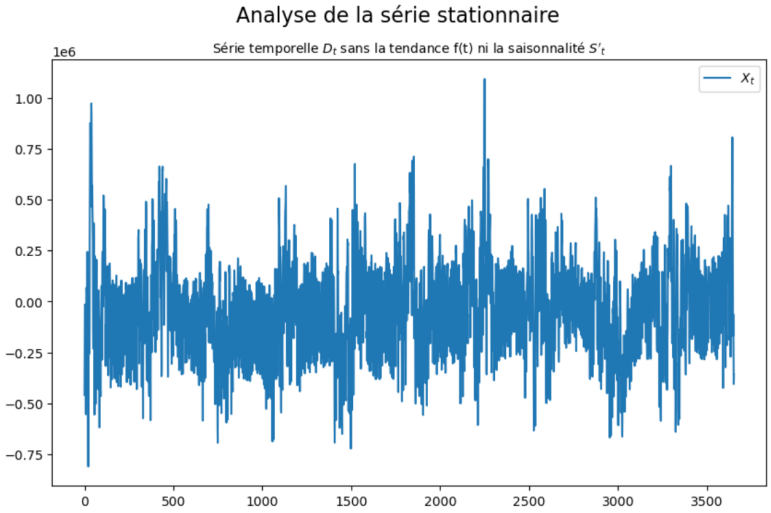
Et puisque $\frac{X_t+X_{t+T}+...+X_{t+9T}}{n} \underset{n\to+\infty}{\overset{\text{p.s.}}{\to}} E[X_t] = 0$, il en résulte que :

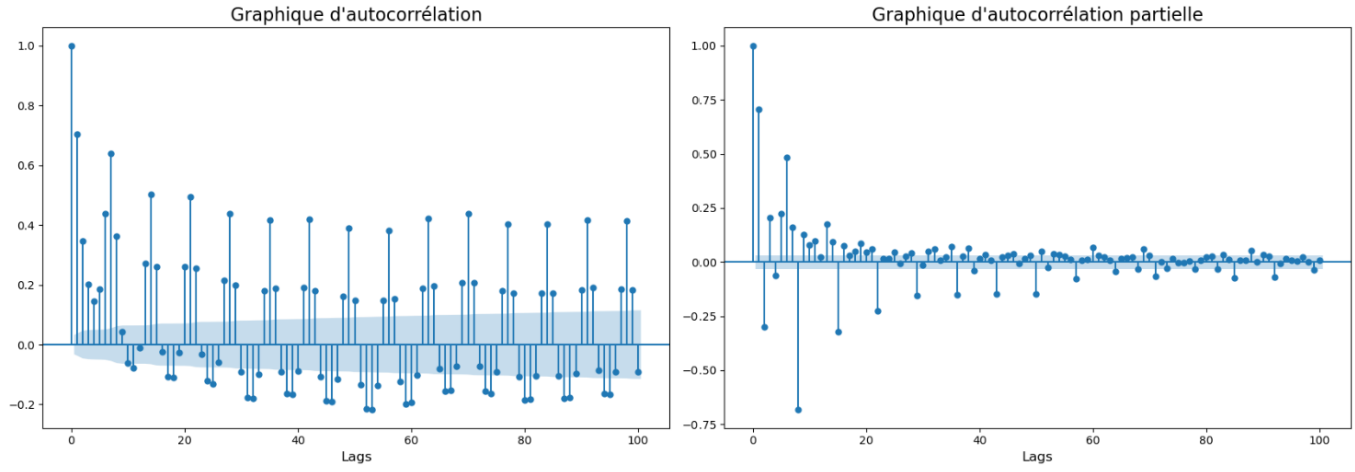
$$S'_t \underset{n\to+\infty}{\overset{\text{p.s.}}{\to}} D_t - f(t)$$

En outre, puisque la convergence est rapide d'après la loi forte des grands nombres, nous pouvons donc estimer notre saisonnabilité S'_t en faisant la moyenne empirique de chaque jour de l'année sur les 10 années. Notons que dans notre code, nous pouvons enlever le 29 février car l'année 2022 à prédire n'est pas bissextile. Ainsi, la représentation graphique de notre saisonnalité est :



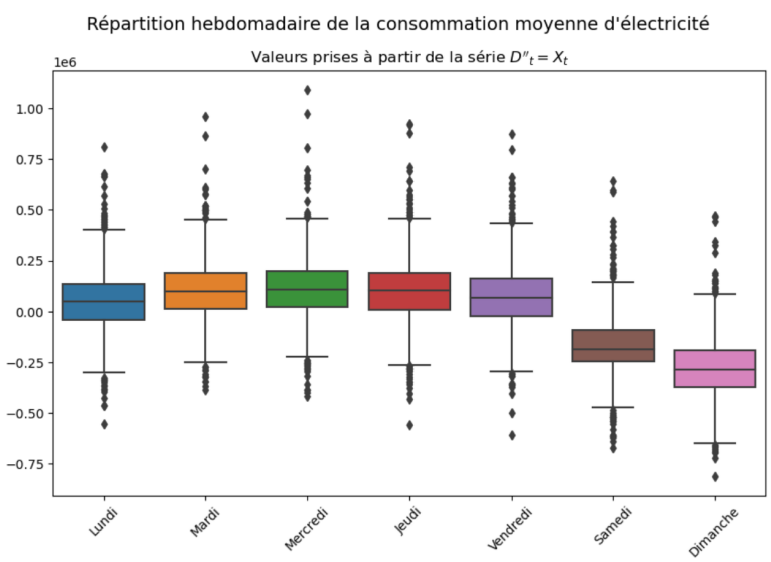
En supposant que nous avons approché notre saisonnalité S_t de la meilleure manière possible (c'est à dire $S'_t = S_t$), nous pouvons désormais la soustraire à notre D'_t. Cela nous permet de représenter notre série stationnaire, notre graphique d'autocorrélation (ACF) et d'autocorrélation partielle (PACF) :



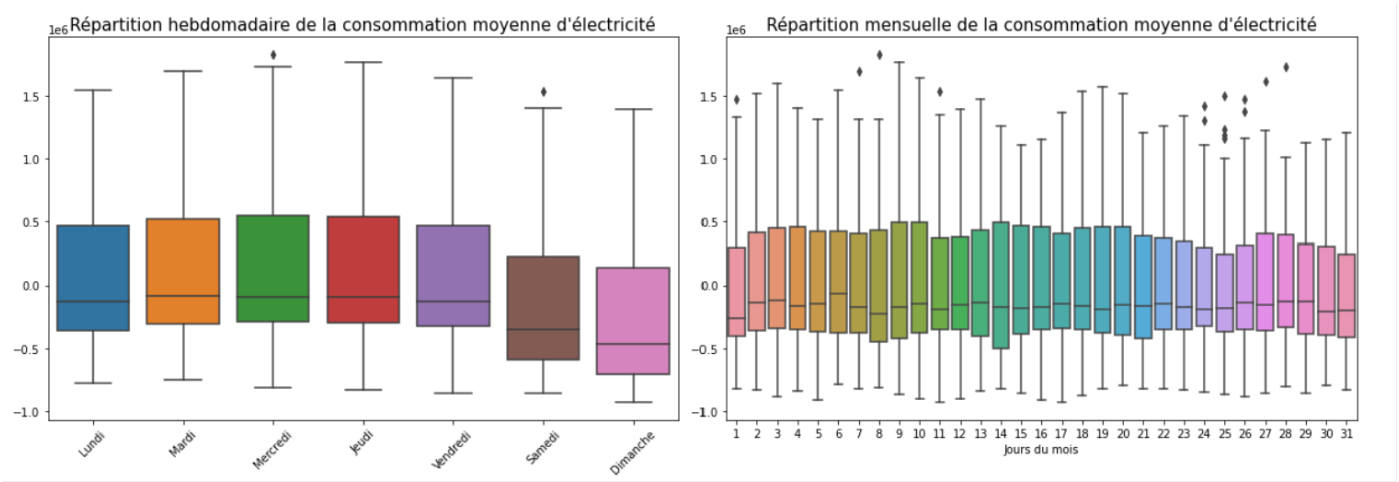


Notons cette série temporelle D_t sans la tendance $f(t)$ ni la saisonnalité S'_t : $D''_t = X_t$.
Sur le graphique d'autocorrelation, nous observons une saisonnalité de période $T = 7$.

En effet, en effectuant un boxplot de la série temporelle précèdente $D''_t = X_t$, nous pouvons remarquer que le week-end, la consommation d'électricité baisse comparé aux autres jours de la semaine. Sans doute dû à la réduction des activités industrielles et commerciales des français.



Effectivement, en reprenant la série temporelle $D'_t = S_t + X_t$, réintéressons-nous au calcul de la moyenne, des quartiles et de la médiane, mais cette fois à l'échelle hebdomadaire et mensuelle.
Nous observons bien une période de $T = 7$ pour les 7 jours de la semaine. Cependant à l'échelle mensuelle, nous pouvons remarquer qu'il n'y a pas de période pertinente.
Nous illustrons cela en réprésentant les boxplots de ces deux échelles :



Nous allons donc enlenver la saisonnalité sur les 7 jours de la semaine afin d'avoir une série totalement stationnaire, notons cette saisonnalité hebdomadaire S''_t.

Pour cela, il était nécessaire de déterminer la répartition des jours entre janvier 2012 et décembre 2021. Pour ensuite faire la moyenne.

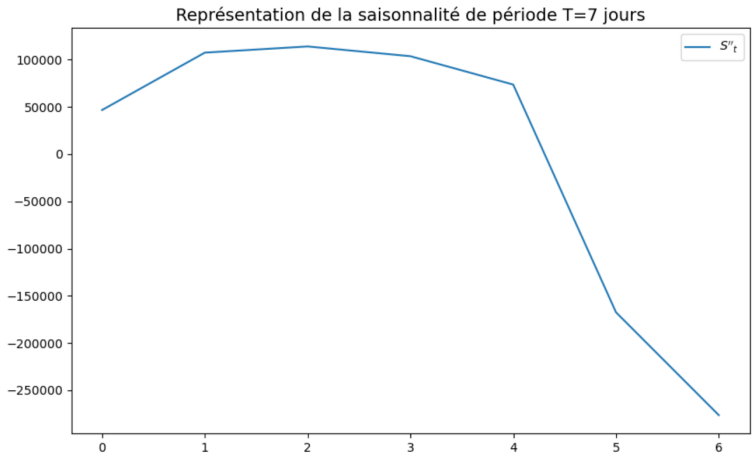
```
for i in range(7):
    num = new_data_stationnaire2.loc[new_data_stationnaire2['Day_of_Week'] == i].shape[0]
    jour = ['Lundi', 'Mardi', 'Mercredi', 'Jeudi', 'Vendredi', 'Samedi', 'Dimanche'][i]
    print("Il y a", num, jour, "de janvier 2012 à décembre 2021.")
```

```
Il y a 522 Lundi de janvier 2012 à décembre 2021.
Il y a 522 Mardi de janvier 2012 à décembre 2021.
Il y a 522 Mercredi de janvier 2012 à décembre 2021.
Il y a 522 Jeudi de janvier 2012 à décembre 2021.
Il y a 522 Vendredi de janvier 2012 à décembre 2021.
Il y a 521 Samedi de janvier 2012 à décembre 2021.
Il y a 522 Dimanche de janvier 2012 à décembre 2021.
```

```
daily_sum = new_data_stationnaire2.groupby('Day_of_Week')["Consommation brute d'électricité (MW)"].sum().reset_index()
```

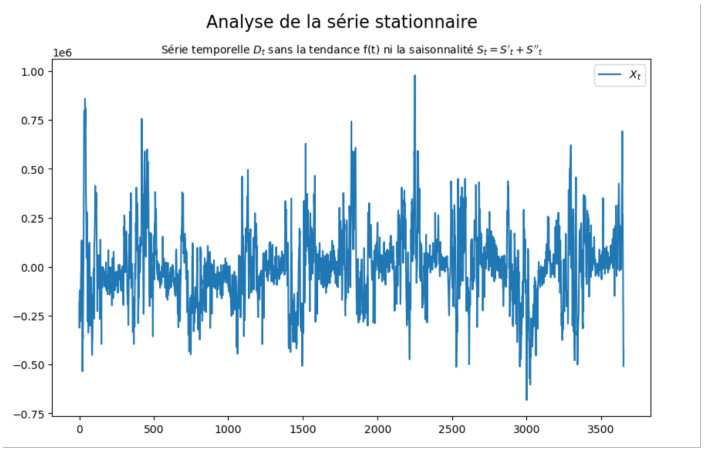
```
moyennes = np.zeros(7)
num = 522
for i in range(7):
    if i != 5:
        moyennes[i] = daily_sum["Consommation brute d'électricité (MW)"][i] / num
    else:
        moyennes[i] = daily_sum["Consommation brute d'électricité (MW)"][i] / 521
```

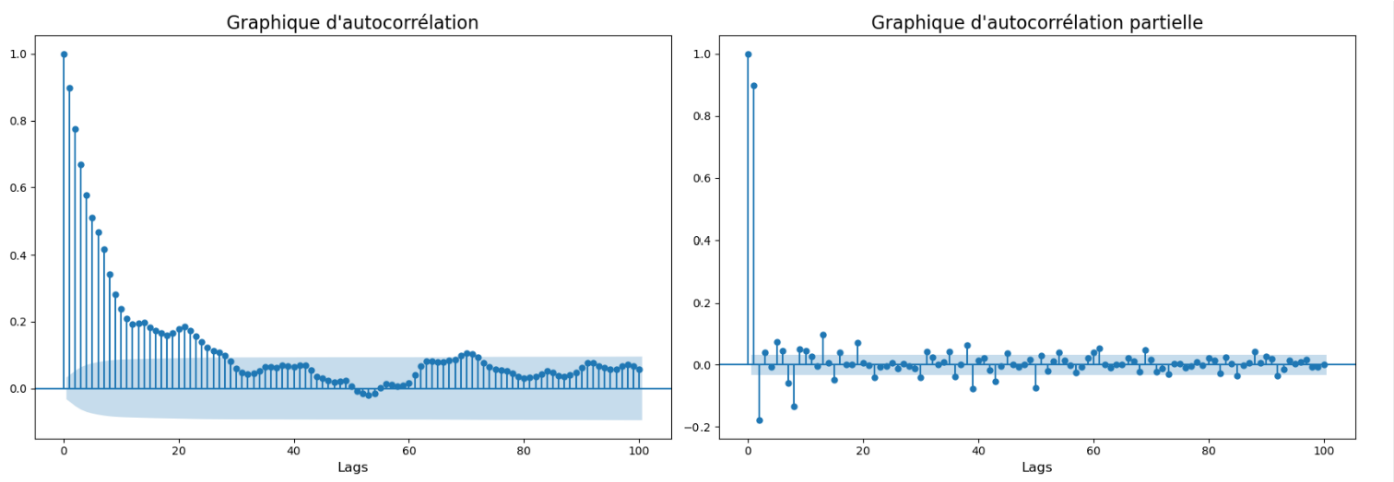
Ainsi, la représentation graphique de notre saisonnalité S_t'' est :



Dès lors, en additionnant S_t' et S_t'', nous pouvons supposer que nous avons approché S_t de la meilleure manière possible (c'est à dire $S_t = S_t' + S_t''$)

Afin de vérifier cela, nous pouvons soustraire notre saisonnalité S_t à D_t'. Cela nous permet d'afficher notre série stationnaire, notre graphique d'autocorrélation (ACF) et d'autocorrélation partielle (PACF) :





Visuellement, nous ne discernons plus de saisonnalité apparente. À présent, nous pouvons nous intéresser à la stationnarité de notre série.

2.3 Stationnarité

Avant toute chose, assurons-nous que la série X_t est bien stationnaire.
Pour cela, nous effecturons un test ADF (Augmented Dickey-Fuller), ce test est utilisé pour évaluer si une série temporelle est stationnaire ou non. La p-value est particulièrement importante : si elle est inférieure à un seuil défini (souvent 0.05), on peut conclure que la série temporelle est stationnaire.

```
result = adfuller(data_stationnaire3)

print('Statistique ADF :', result[0])
print('p-value :', result[1])
print('Nombre de lags utilisés :', result[2])
print("Nombre d'observations :", result[3])
print('Valeurs critiques :', result[4])
```

```
Statistique ADF : -9.065033858098372
p-value : 4.456401858434083e-15
Nombre de lags utilisés : 21
Nombre d'observations : 3631
Valeurs critiques : {'1%': -3.4321522387754775, '5%': -2.862336328589075, '10%': -2.567193897993964}
```

Notre p-value est extrêmement basse (p-value $= 4,45.10^{-15}$), cela suggère une forte évidence contre l'hypothèse nulle selon laquelle la série temporelle n'est pas stationnaire. En d'autres termes, nous pouvons rejeter l'idée que notre série temporelle est non stationnaire.

De plus, en examinant plus précisément le graphique de l'ACF obtenu précédemment, nous pouvons observer que les pics décroissent géométriquement. En observant également le graphique de la PACF, nous pouvons remarquer la présence de 2 pics significatifs aux lags 1 et 2. Cela nous amène à penser que le modèle le plus approprié pour prédire notre série temporelle serait un modèle autorégressif d'ordre 2 (AR(2)).

Plus concrètement, une série stationnaire X_t qui satisfait un modèle AR(2) est une équation de la forme :

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + Z_t \qquad t \in \mathbb{Z}$$

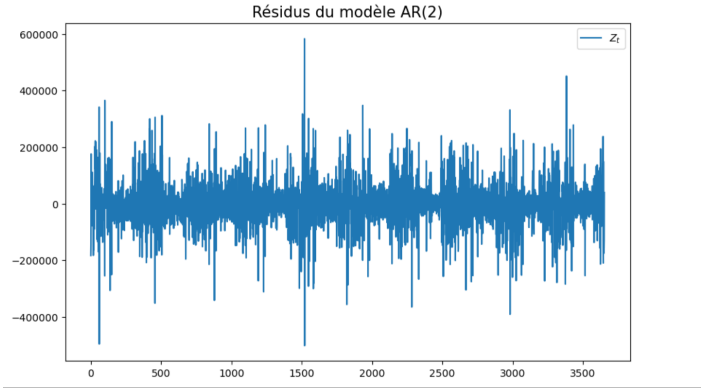
où ϕ_1 et ϕ_2 sont les coefficients autorégressifs qui mesurent l'impact des valeurs passées sur la valeur actuelle ; et $Z_t \in BB(\sigma^2)$ représentant le terme d'erreur, c'est à dire les résidus à l'instant t. Ces résidus capturent les informations non expliquées par le modèle autorégressif lui-même, c'est à dire les variations aléatoires.

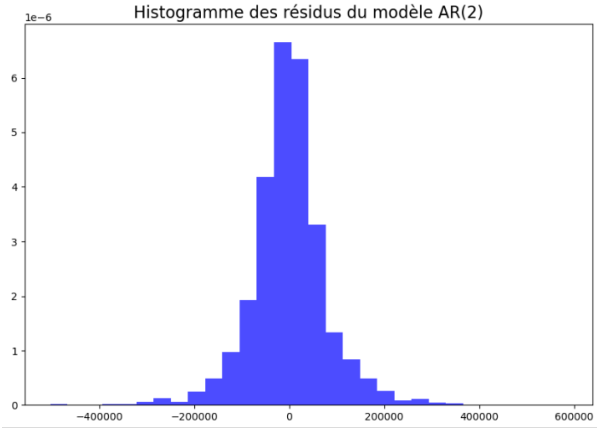
Ainsi, en ajustant un modèle ARIMA spécifique à nos données stationnaires, nous obtenons, par maximum de vraisemblance, les coefficients ϕ_1 et ϕ_2 de notre modèle AR(2) :

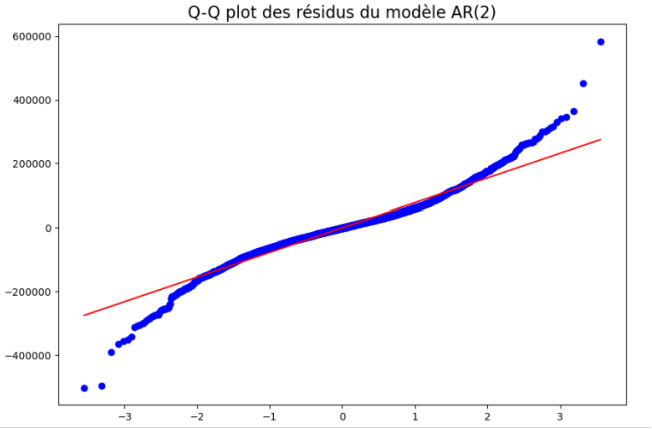
```
p, q = 2, 0
model = sm.tsa.ARIMA(data_stationnaire3, order=(p, 0, q))
result = model.fit()
print('phi_1 =', result.arparams[0])
print('phi_2 =', result.arparams[1])
```

```
phi_1 = 1.0594341337217001
phi_2 = -0.17680893234600245
```

L'analyse des résidus est également une étape importante dans la modélisation de notre séries temporelle. Des résidus bien comportés (par exemple, des résidus indépendants et identiquement distribués) sont indicatifs d'un bon ajustement du modèle aux données, tandis que des motifs dans les résidus pourraient indiquer des aspects non capturés par le modèle. Dès lors, étudions plus précisèment ces résidus :







Le graphique des résidus ne nous montre pas de tendance significative au fil du temps, ce qui est positif car cela suggère que le modèle n'est pas corrélé avec le temps.
L'histogramme des résidus ressemble à une distribution normale, de plus elle est symétrique. Donc les résidus suivent approximativement une distribution normale, cela suggère que le modèle capture bien la variabilité dans les données.
Le graphique quantile-quantile (Q-Q plot) des résidus compare les quantiles théoriques d'une distribution normale avec les quantiles observés des résidus. Nous remarquons que les points suivent approximativement la ligne de référence, cela indique une bonne adéquation à la distribution normale.

Ainsi, l'objectif était d'identifier toute structure restante dans les résidus. Etant donné que les résidus ne montrent pas de schéma évident, cela suggère que le modèle a bien capturé les motifs dans les données. De plus :

```
print('sigma^2 =', result.params[2])
```

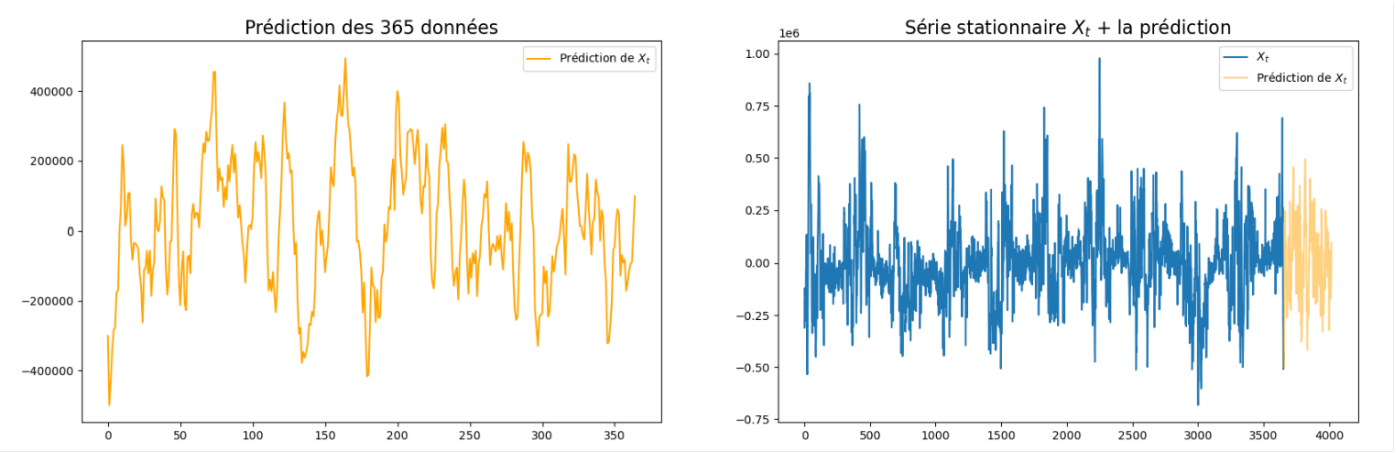
```
sigma^2 = 6269308157.138818
```

Nous pouvons donc affirmer que notre série stationnaire est une équation de la forme :

$$X_t = 1.0594341337217001.X_{t-1} - 0.17680893234600245.X_{t-2} + Z_t \qquad t \in \mathbb{Z}$$

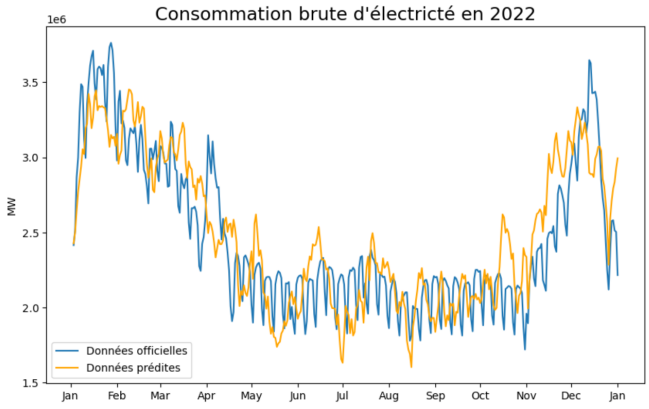
avec $Z_t \in BB(\sigma^2 \approx 6,2.10^9)$, les résidus bien comportés signifiant que le modèle est en mesure de fournir des prédictions fiables.

En conséquence, nous pouvons afficher la prédiction de notre série stationnaire pour l'année 2022 :

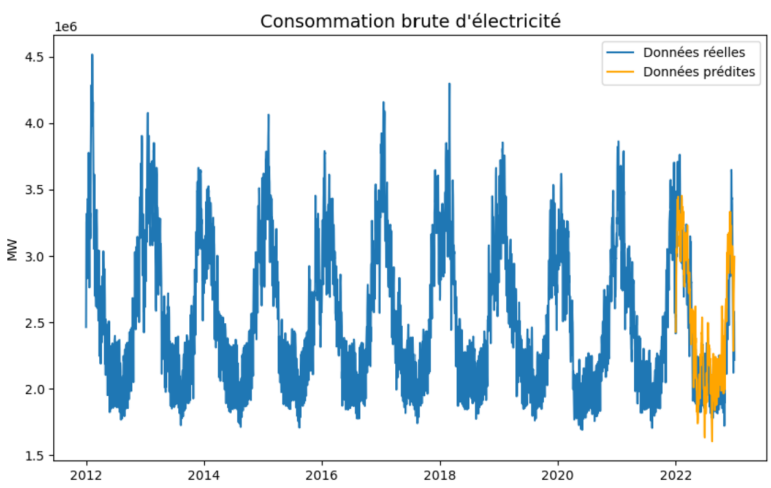


3 Analyse des résultats

Grâce à l'étude approfondie de notre série temporelle, nous sommes désormais en mesure de prédire la consommation brute d'électricité pour l'année 2022. En effet, en combinant la prédiction de notre tendance $f(t)$, notre saisonnalité, ainsi que la prédiction de notre stationnarité X_t, nous pouvons afficher l'ensemble de notre prédiction afin de la confronter aux données officielles de la consommation d'électricité en 2022.



Afin d'obtenir une vision plus globale de nos prédictions, il est également intéressant de les confronter avec celles des dix années précédentes :



Il est donc pertinent de calculer l'erreur quadratique moyenne (MSE), une mesure utilisée pour évaluer la performance d'un modèle de régression. Elle quantifie la moyenne des carrés des écarts entre les valeurs prédites par le modèle et les valeurs réelles. La formule mathématique de la MSE est la suivante :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

où :
- n est le nombre total d'observations dans l'ensemble de données
- y_i est la valeur réelle pour l'observation i
- $\hat{y}_i$ est la valeur prédite par le modèle pour l'observation i

Avant de calculer notre MSE, commençons d'abord par normaliser nos données. En effet, étant donné que nos valeurs sont très élevées, il serait plus judicieux de les ramener à une échelle comparable :

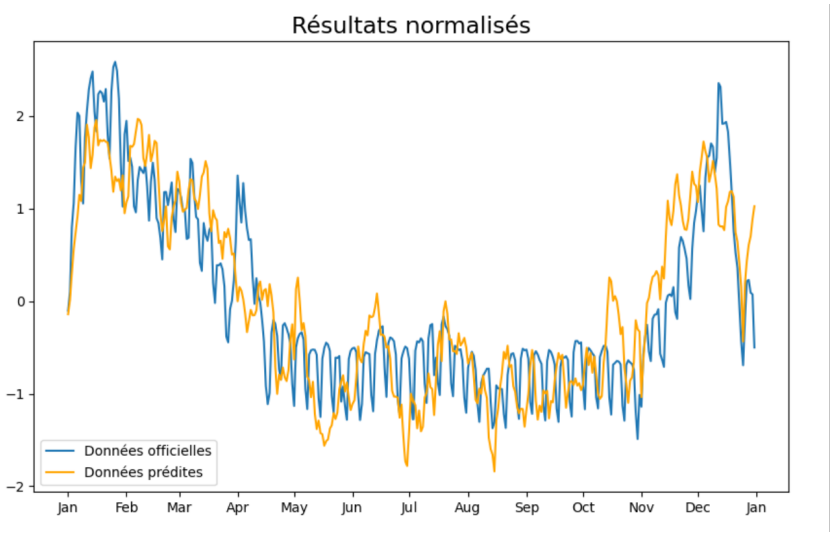
```
prediction_norm = prediction.copy()
data_final_test_norm = data_final_test["Consommation brute d'électricité (MW)"].copy()
```

```
prediction_norm = prediction_norm.values
data_final_test_norm = data_final_test_norm.values
```

```
prediction_norm = prediction_norm.reshape(-1, 1)
data_final_test_norm = data_final_test_norm.reshape(-1, 1)
```

```
sc1 = StandardScaler()
sc2 = StandardScaler()
prediction_norm = sc1.fit_transform(prediction_norm)
data_final_test_norm = sc1.fit_transform(data_final_test_norm)
```

Ceci nous permet donc d'obtenir ce graphique :



Dès lors, nous pouvons désormais calculer notre MSE grâce à la fonction python "mean_squared_error" :

```
mse = mean_squared_error(data_final_test_norm, prediction_norm)
print("MSE =", mse)
```

```
MSE = 0.34282029259533614
```

Une MSE d'environ 0.34 pour des données normalisées variant entre -2 et 2.5, suggère que notre modèle présente une performance relativement bonne en terme de prédiction.
De plus, en tenant compte de l'aspect visuel de nos prédictions, nous pouvons penser que celles-ci sont relativement correctes, ce qui renforce la confiance dans la performance globale de notre modèle !

Conclusion

La réalisation de ce projet dédié à la prédiction de la consommation d'électricité en France pour l'année 2022 a été une immersion captivante dans le monde « des séries temporelles ». Notre exploration approfondie de la série temporelle provenant de la base de données du Réseau de transport d'électricité (RTE), couvrant une période de 10 ans, nous a permis de dégager des tendances significatives, des motifs saisonniers et d'évaluer la stationnarité des données.

Guidés par une méthodologie rigoureuse, nous avons opté pour un modèle autorégressif AR(2) pour générer nos prédictions. L'évaluation de la performance de notre modèle, exprimée par un Mean Squared Error (MSE) suggère une performance relativement bonne de notre modèle en terme de précision des prédictions. De même, l'aspect visuel de nos prédictions renforce notre confiance dans la robustesse du modèle. Les graphiques illustrant les données réelles par rapport à nos prédictions démontrent une cohérence apparente, renforçant ainsi la validité de notre approche de modélisation.

En conclusion, notre projet a réussi à développer un modèle de prédiction, combinant l'analyse approfondie des données avec de la programmation Python. Les résultats obtenus offrent des perspectives intéressantes pour anticiper la consommation d'électricité en France pour l'année 2022. Ce travail représente une contribution significative à la compréhension des modèles de séries temporelles dans le domaine de l'énergie, démontrant le potentiel de l'approche holistique que nous avons adopté.

En poursuivant cette voie, il serait pertinent d'explorer des techniques plus avancées de modélisation (comme l'utilisation de modèles différents ou de variables explicatives) et d'élargir la portée temporelle pour une meilleure généralisation. Ce projet ouvre ainsi la porte à des développements futurs dans le domaine de la prévision de la consommation d'électricité, contribuant ainsi à relever les défis pratiques du secteur de l'énergie.

Bibliographie

- Lien complet du jeu de données :

https ://odre.opendatasoft.com/explore/dataset/consommation-quotidienne-brute/information/ ?flg=fr-fr&sort=-date_heure&dataChart=eyJxdWVyaWVzIjpbeyJjaGFydHMiOlt7InR5cGUiOiJsaW5lIiwiZnVuYyI6IlNVTSIsInlBeGlzIjoiY29uc29tbWF0aW9uX2JydXRlX2VsZWN0cmljaXRlX3J0ZSIsInNjaWVudGlmaWNEaXNwbGF5Ijp0cnVlLCJjb2xvciI6IiM4ZGEwY2IifV0sInhBeGlzIjoiZGF0ZV9oZXVyZSIsIm1heHBvaW50cyI6bnVsbCwidGltZXNjYWxlIjoiZGF5Iiwic29ydCI6IiIsImNvbmZpZyI6eyJkYXRhc2V0IjoiY29uc29tbWF0aW9uLXF1b3RpZGllbm5lLWJydXRlIiwib3B0aW9ucyI6eyJmbGciOiJmci1mciIsInNvcnQiOiItZGF0ZV9oZXVyZSJ9fX1dLCJzaW5nbGVBeGlzIjp0cnVlLCJkaXNwbGF5TGVnZW5kIjp0cnVlLCJhbGlnbk1vbnRoIjp0cnVlfQ%3D%3D

- Time Series Analysis (Frédéric Guilloux) :
https://moodle-sciences-23.sorbonne-universite.fr/pluginfile.php/175543/mod$_r esource/content/2/TSA2023.pdf$

- Time Series Analysis (Olivier Wintenberger) :
https://wintenberger.fr/cours/TSA/TSA2022.pdf

- Méthodes et formules pour le test ADF :
https://support.minitab.com/fr-fr/minitab/21/help-and-how-to/statistical-modeling/time-series/how-to/augmented-dickey-fuller-test/methods-and-formulas/methods-and-formulas/

- Librairie Python pour le modèle ARIMA :
https://www.statsmodels.org/dev/generated/statsmodels.tsa.arima.model.ARIMAResults.html