LAMRINI Mohamed-Amin, MONIN Louis

ING2 MI2 – Mathématiques Appliquées

Introduction aux séries temporelles

Projet

LAIB Naamane 2023-2024

Contents

Introduction

Le mini-projet a pour objectif de mettre en application les différentes méthodes vues en cours sur un jeu de données réelles que nous aurons choisi. Pour mener à bien ce projet, ce dernier est composé de trois étapes qui sont les caractéristiques de la série. Puis la désaisonnalisation avec deux méthodes qui sont la moyenne mobile et la régression sur le temps et avec des variables indicatrices. Enfin, nous aurons une prévision de la série par lissage exponentiel.

Partie 1 : Caractéristiques de la série

Description de la série

Présentation du dataset

A l'origine le fichier représente la température quotidienne régionale (depuis le 1er janvier 2016 au 31 janvier 2024). Ce jeu de données présente les températures minimales, maximales et moyennes quotidiennes (en degré celsius), par région administrative française, du 1er janvier 2016 à aujourd'hui.

Il est basé sur les mesures officielles du réseau de stations météorologiques françaises. La mise à jour de ce jeu de données est mensuelle.

Lien du site internet : https://www.data.gouv.fr/fr/datasets/temperature-quotidienne-regionale-depuis-janvier-2016/

URL du dataset : https://www.data.gouv.fr/fr/datasets/r/d707a3b3-c0eb-404b-b3ef-2d4e0ddb96f4

Description du fichier :

-ID : id[text]

-Date : date[date] Date de l'observation

-Code INSEE région : code_insee_region[int] Code INSEE région administrative

-Région : region[text] Région administrative

-TMin (°C) : tmin[double] Température minimale quotidienne

-TMax (°C) : tmax[double] Température maximale quotidienne

-TMoy (°C) : tmoy[double] Température moyenne quotidienne

A l'aide d'un programme python nous avons supprimer toutes les données des régions qui n'était pas celles de l'ile-de-France.

Puis nous avons supprimer les colonnes contenant l'ID, le code_insee_region, la région, la température minimale et maximale. Nous avons conservé uniquement la date et la température moyenne.

Enfin nous avons fait une moyenne par mois des températures. Le code se présente sous la forme d'un fichier .py nommé « Code_Transformer_Donnees » donné en annexe. [1]

Finalement, notre dataset correspond à une moyenne par mois des relevés de la température de l'île-de-France du 1er janvier 2016 au 31 janvier 2024.

```
donnees <- read.xlsx("Temperature_moyenne_mensuelle_ile_de_france_v2.xlsx", sheet = 1)
View(donnees)
```

	Mois-Annee	tmoy
1	Janvier-2016	5.534839
2	Février-2016	6.138276
3	Mars-2016	6.791290
4	Avril-2016	9.883000
5	Mai-2016	14.412258
6	Juin-2016	17.576667
7	Juillet-2016	20.325484
8	Août-2016	20.919677
9	Septembre-2016	18.782000
10	Octobre-2016	11.459355
11	Novembre-2016	7.657333
12	Décembre-2016	4.457742
13	Janvier-2017	1.950000
14	Février-2017	7.284286
15	Mars-2017	10.492903
16	Avril-2017	10.490667

Il est composé de 2 colonnes les mois entre 2016 et 2024 et la température moyenne (tmoy). Avec la librairie knitr, on peut présenter le dataset sous une autre forme.

```
#Générer le tableau des valeurs avec knitr

library(knitr)

knitr::kable(donnees, booktabs = TRUE, col.names = c("Mois-Annee", "tmoy"))
```

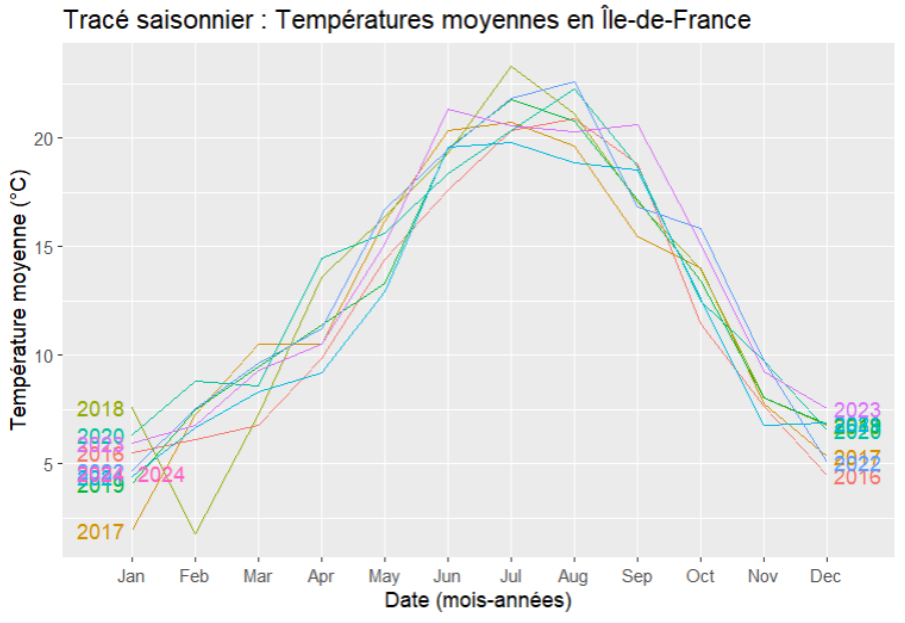
Mois-Annee	tmoy
Janvier-2016	5.534839
Février-2016	6.138276
Mars-2016	6.791290
Avril-2016	9.883000
Mai-2016	14.412258
Juin-2016	17.576667
Juillet-2016	20.325484
Août-2016	20.919677
Septembre-2016	18.782000
Octobre-2016	11.459355
Novembre-2016	7.657333
Décembre-2016	4.457742
Janvier-2017	1.950000
Février-2017	7.284286
Mars-2017	10.492903
Avril-2017	10.490667
Mai-2017	16.156774
Juin-2017	20.366333
Juillet-2017	20.752903
Août-2017	19.644839
Septembre-2017	15.462000
Octobre-2017	14.045161
Novembre-2017	7.749333
Décembre-2017	5.342903
Janvier-2018	7.578387
Février-2018	1.783929
Mars-2018	7.249032

Pour continuer l'étude de notre dataset, nous utilisions la commande ts pour transformer les données des températures moyennes en une série temporelle en supposant une fréquence annuelle de 12 mois et en indiquant que la série commence en 2016 et se termine en 2024.

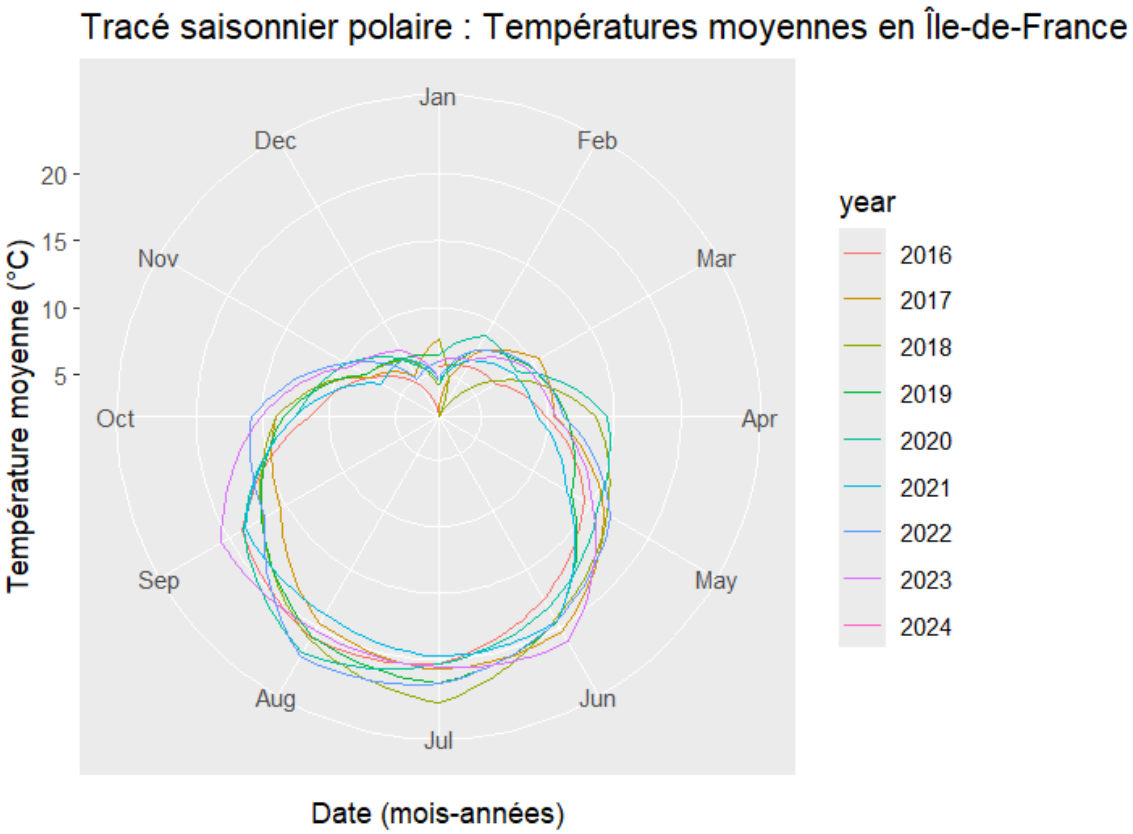
```
# Créer une série temporelle

donnees_ts <- ts(donnees$tmoy, frequency=12, start=c(2016,1), end =c(2024,1))
donnees_ts
```

Pour mieux analyser le contenu de chaque périodicité, nous avons effectué un tracé saisonnier des températures moyennes en Ile-de-France. On constate une forte hausse des températures à la période de Juillet-Août pour chaque année, ce qui en adéquation avec le fait que c'est l'été et que les températures augmentent.

```
# Utiliser ggseasonplot pour visualiser les données
# Charger les packages nécessaires

library(forecast)
library(ggplot2)

ggseasonplot(donnees_ts, year.labels=TRUE, year.labels.left=TRUE) +
  ylab("Température moyenne (°C)") +
  xlab("Date (mois-années)") +
  ggtitle("Tracé saisonnier : Températures moyennes en Île-de-France")
```

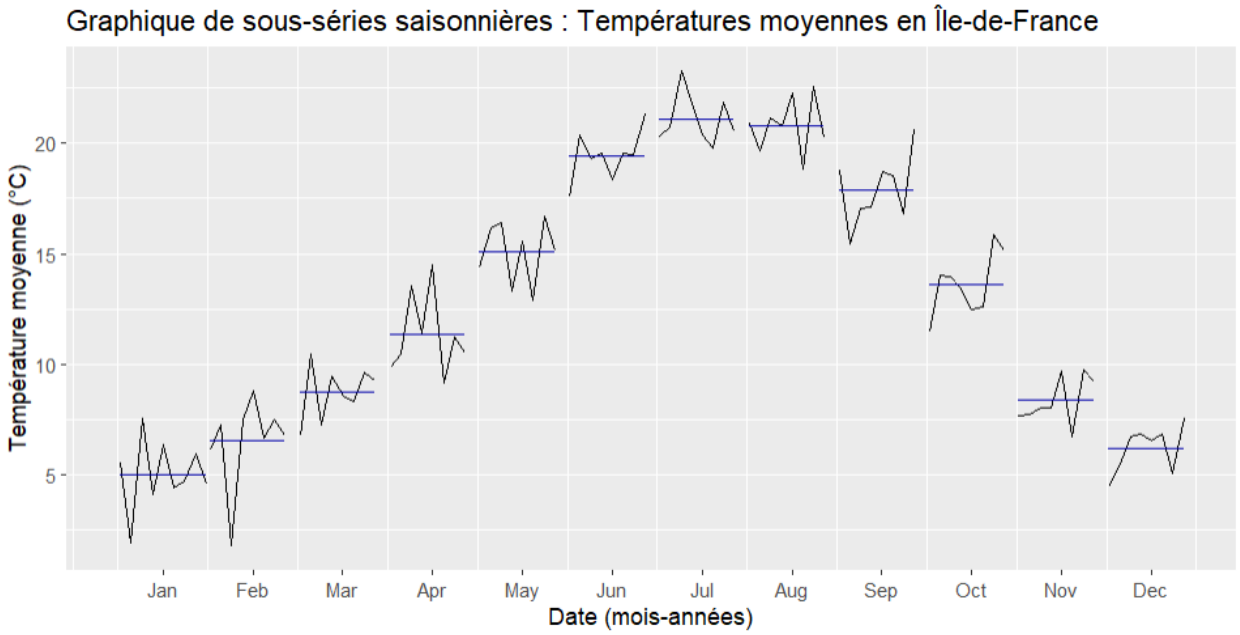


```r
# Utiliser ggseasonplot en mode polaire pour visualiser les données
ggseasonplot(donnees_ts, polar=TRUE) +
  ylab("Température moyenne (°C)") +
  xlab("Date (mois-années)") +
  ggtitle("Tracé saisonnier polaire : Températures moyennes en Île-de-France")
```



Pour compléter notre étude sur notre dataset, nous avons réalisé un graphique saisonnier alternatif. Les lignes horizontales indiquent les moyennes pour chaque mois. Cette forme de graphique permet de voir clairement le modèle saisonnier sous-jacent et montre également les changements de saisonnalité au fil du temps. De ce graphique, on peut observer que les températures augmentent et diminuent dans un motif répétitif qui correspond aux saisons. Les températures culminent en été (Juillet-Août) et son au plus bas en hiver (Janvier-Février). On peut aussi discerner quelques fluctuations importantes d'une année à l'autre, comme témoignent les pointes et les creux variables à certains mois. Par exemple, il y a des pointes qui dépassent les tendances saisonnières habituelles, qui pourraient indiquer des vagues de chaleur exceptionnelles.

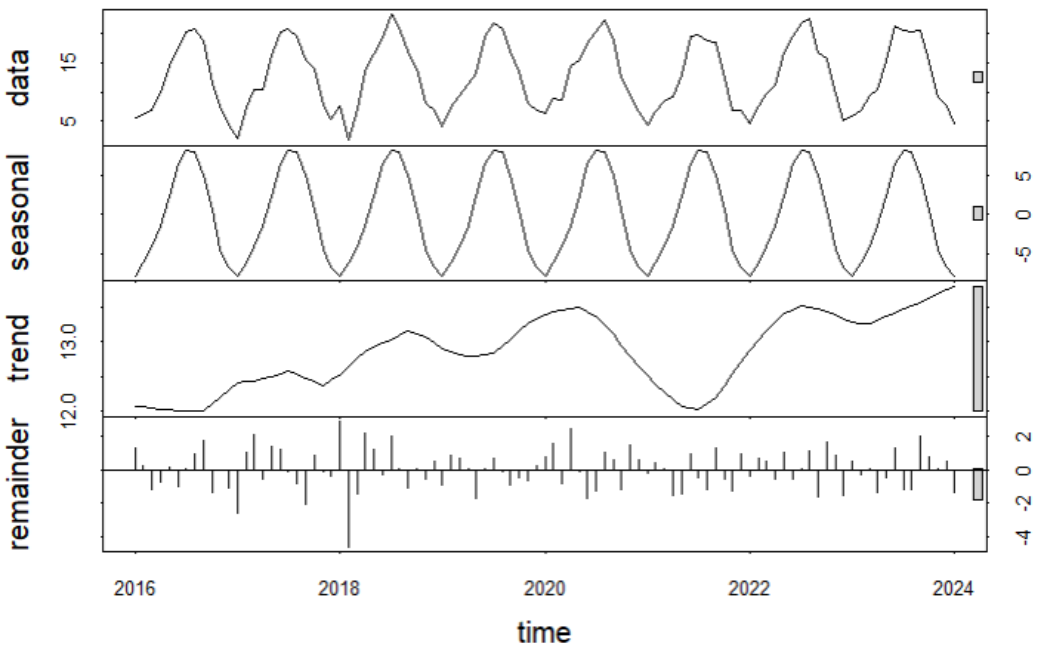
```r
ggsubseriesplot(donnees_ts) +
  ylab("Température moyenne (°C)") +
  xlab("Date (mois-années)") +
  ggtitle("Graphique de sous-séries saisonnières : Températures moyennes en Île-de-France")
```



Analyse de la tendance et de la saisonnalité

Pour réaliser une étude de la tendance et de la saisonnalité, nous avons réaliser une étude sur R avec la méthode stl.

```r
#Analyse de la tendance et de la saisonnalité
# Décomposition STL

decomp <- stl(donnees_ts, s.window="periodic")

# Visualiser la décomposition

plot(decomp)
```



Le premier graphique montre la décomposition STL de notre série temporelle en trois composantes : saisonnalité, tendance et résidus. Au niveau des données saisonnières (deuxième graphique du haut), il semble y avoir un motif saisonnier clair et cohérent qui se répète chaque année, ce qui est attendu pour des données de températures (plus chaud en été, plus froid en hiver).

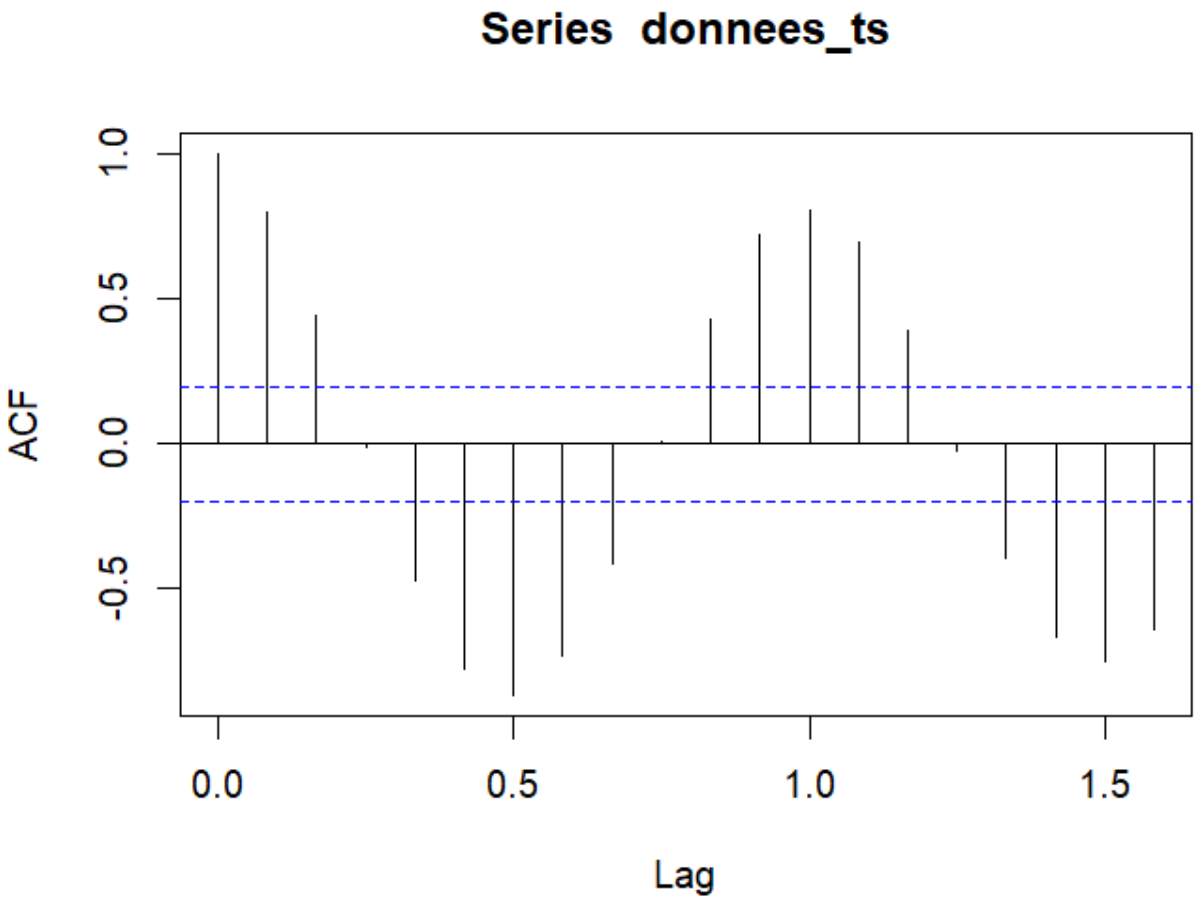
La tendance (troisième graphique du haut) est relativement lisse sans fluctuations extrêmes, suggérant des changements graduels dans les températures moyennes sur plusieurs années.

Enfin, le résidus (graphique du bas) semblent être relativement faibles et aléatoires, ce qui indique que la décomposition a capturé la plupart des comportement systématiques de la série temporelle et ce qui reste est principalement du bruit.

Analyse de la corrélation

Lors de l'étude de la corrélation de notre série nous avons obtenu ce résultat.

```
#Analyse de la corrélation

#Autocorrélation

acf(donnees_ts)
```



Dans notre cas la série $(X_t)1\leq t\leq n$ est une série périodique pure $X_t = a\cos(2\pi t/p)$, où p est la période, car pour tout h fixé :

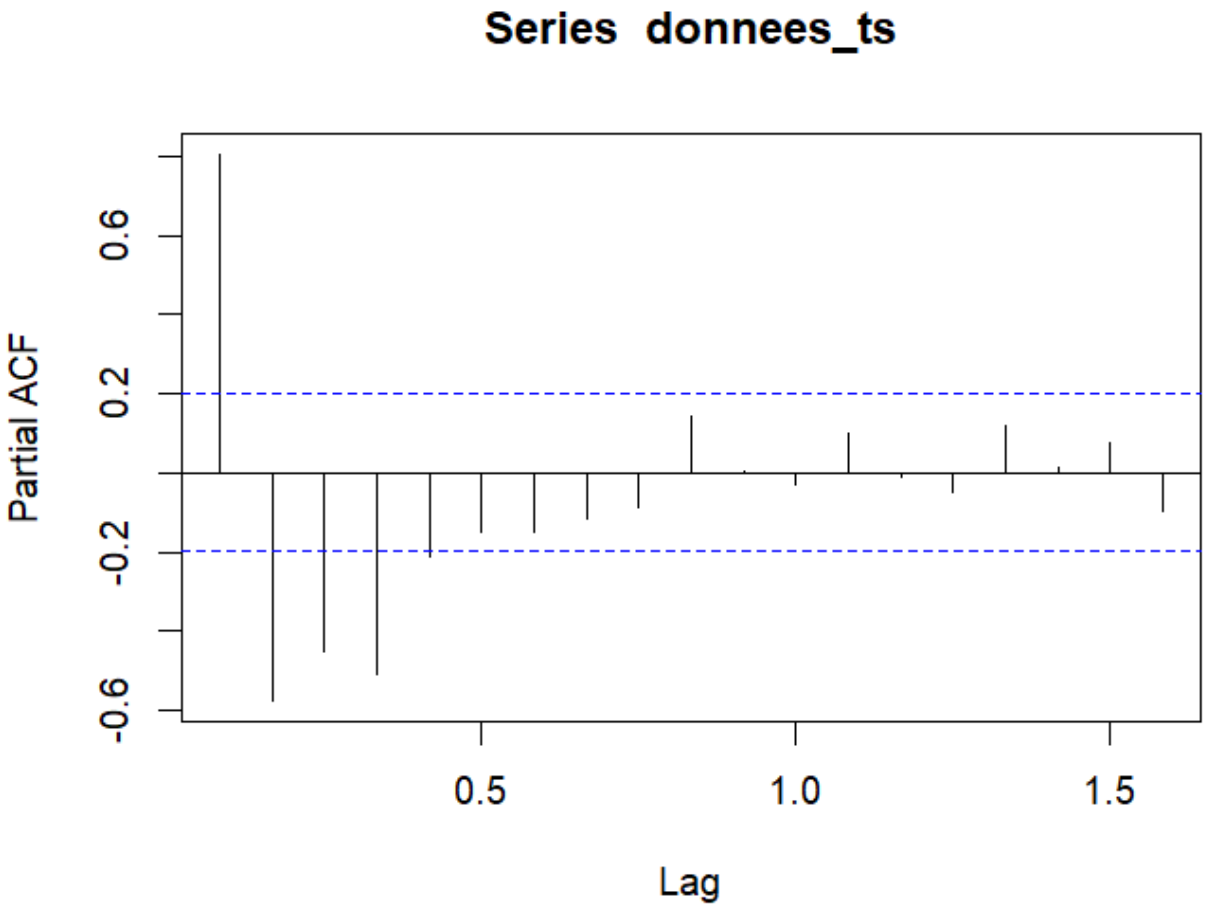
$\rho_n(h) \rightarrow \cos(2h\pi/p)$ quand $n \rightarrow +\infty$

Finalement on retrouve la présence d'une tendance et d'une saisonnalité en examinant les auto-corrélations.

Analyse de l'auto-corrélation partielle

L'auto-corrélation partielle empirique d'ordre h est une fonction de h, noté r(h), qui quantifie la corrélation entre deux données espacées de h par pas de temps, en enlevant l'effet des données intermédiaires.

```
# Autocorrélation partielle

pacf(donnees_ts)
```



Analyse de la stationnarité

Le test de Dickey-Fuller augmenté (ADF) est utilisé pour déterminer si une série temporelle est stationnaire ou non, c'est-à-dire si la série présente une tendance au fil du temps.

```
> adf.test(donnees_ts)

        Augmented Dickey-Fuller Test

data:  donnees_ts
Dickey-Fuller = -10.548, Lag order = 4, p-value = 0.01
alternative hypothesis: stationary
```

Valeur de Dickey-Fuller : -10.548. Cette statistique est négative, ce qui est un bon indicateur. Plus cette valeur est négative, plus la preuve contre l'hypothèse nulle (la présence d'une racine unitaire, indiquant une non-stationnarité) est forte.

Ordre de retard (Lag order) : 4. Cela indique que le test a utilisé 4 retards dans la construction de l'équation de test ADF pour la série temporelle, ce qui correspond au nombre de périodes prises en compte pour calculer la corrélation dans la série.

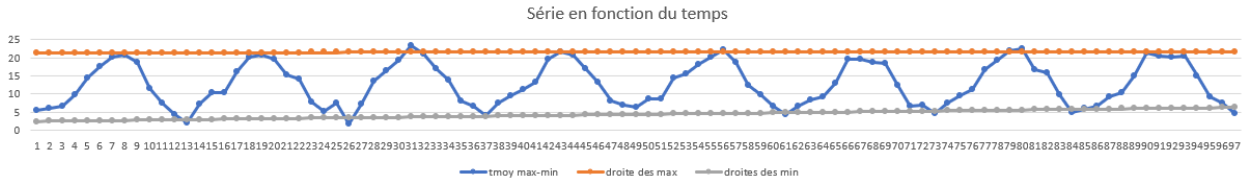
Valeur-p: 0.01. En statistiques, la valeur-p est utilisée pour déterminer la signification statistique du résultat du test. Une valeur-p inférieure à un seuil (généralement 0.05) indique que vous pouvez rejeter l'hypothèse nulle. Dans ce cas, une valeur-p de 0.01 suggère que vous pouvez rejeter l'hypothèse de non-stationnarité avec une confiance de 99 %.

Hypothèse alternative : stationnaire. Le test ADF a une hypothèse alternative selon laquelle la série est stationnaire. Le message d'avertissement indique que la valeur-p réelle est encore plus petite que la valeur imprimée, renforçant l'évidence contre l'hypothèse nulle.

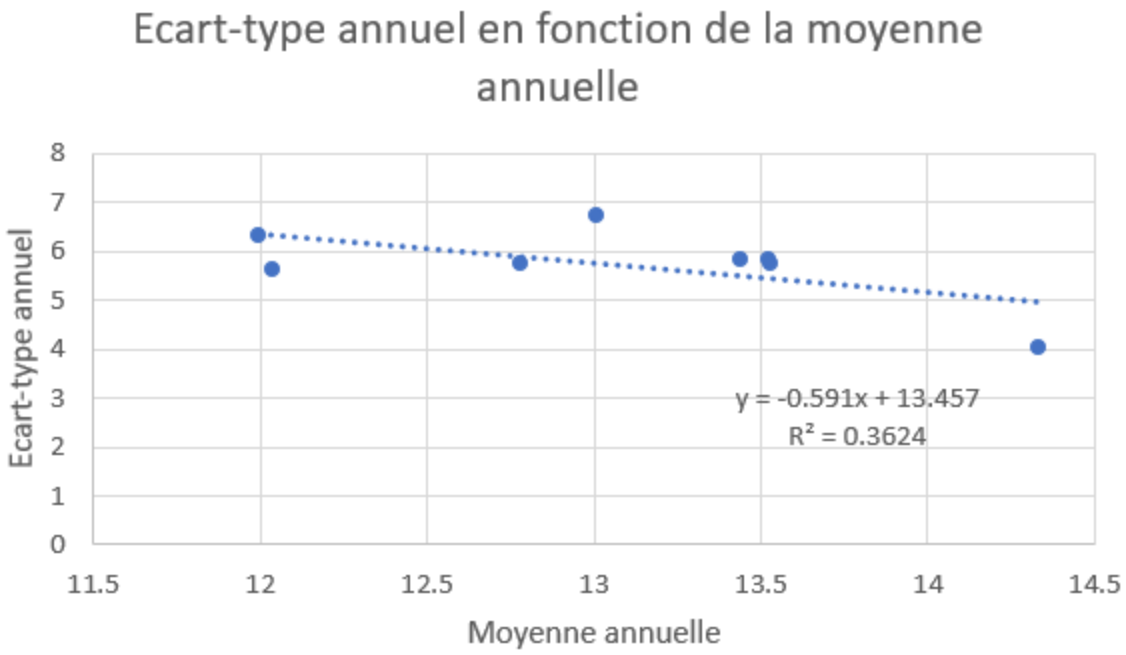
En conclusion, le résultat du test ADF suggère fortement que la série temporelle est stationnaire. Cela signifie que la série temporelle n'a pas de tendance ou de modèle autorégressif intégré, ce qui est une propriété souhaitable lors de l'utilisation de modèles de prévision tels que ARIMA. Cela signifie également que la série temporelle est appropriée pour une analyse plus approfondie et le développement de modèles prédictifs sans nécessiter de différenciation pour rendre la série stationnaire.

Type de modèle (additif/multiplicatif)

Pour choisir notre modèle, nous avons choisi de d'abord procédé par la méthode de l'amplitude qui consiste à représenter les données en fonction de t, puis de tracer les droites des max et des min et de voir si elles sont parallèles dans ce cas on a un modèle additif, sinon on a un modèle multiplicatif.



Nous avons donc obtenu ce résultat qui conforte l'idée que notre modèle est un modèle additif car la droite des max (en orange) et la droite des min (en gris) ne sont pas parallèles.



On constat une forte corrélation entre l'écart-type et la moyenne, avec un coefficient de corrélation linéaire égal à 0.3624. Ce qui renforce l'idée que notre modèle est multiplicatif.

Pour analyser convenablement le modèle et conforter l'idée que notre modèle est bien multiplicatif, nous avons classé les trimestres selon la méthode de Buys-Ballot

		Classement trimestre		
2016	T3	T2	T4	T1
2017	T3	T2	T1	T4
2018	T3	T2	T4	T1
2019	T3	T2	T4	T1
2020	T3	T2	T4	T1
2021	T3	T2	T4	T1
2022	T3	T2	T4	T1
2023	T3	T2	T4	T1

Ce tableau indique la persistance du même classement quelque soit l'année avec le troisième trimestre en première place et le deuxième trimestre en deuxième place et presque tous les premiers trimestres en dernière place. Il existe donc une stationnarité rigide à cette série, qui nous conforte dans le choix d'un modèle additif

Partie 2 : Désaisonnalistion (tableur/R)

1ère méthode : Moyenne mobile

Calculs des coefficients saisonniers par Moyenne mobile

Pour calculer les moyennes mobiles, nous devons calculer les coefficients saisonniers. Notre première application fût sur R, en utilisant la fonction decompose() :

```
#Effectuer la décomposition saisonnière

decomp <- decompose(donnees_ts)
decomp
```

```
 [1] -7.8420075 -6.2373793 -3.8519769 -1.3403172  2.2518118  6.7492985  8.2403162  8.1210957  4.7178015  0.6027868
[11] -4.5619475 -6.8494819
```

Pour comparer les résultats obtenus sur R, nous avons effectuer un travail pour obtenir la valeur des coefficients saisonniers. Tout d'abord, nous avons calculer les moyennes mobiles pour 12 mois soit à partir de t=6 jusqu'à t=72-6. Ensuite, on soustrait aux moyennes mobiles trouvées (Dt), puis on calcule les coefficients provisoires St en calculant la moyenne du résultat précédent pour chaque mois. Afin de center nos coefficients saisonniers, nous soustrayons aux coefficients provisoires leur moyenne.

En

	A	B	C	D	E	F	G	H
1						Lissage Moyenne Mobile		
2	Mois-Annee	t	tmoy (Xt)	M12(X)	Dt=Xt-M12	St = ct (provisoire)	ct*(modifié	Xt_CSV(Additif)
3	Janvier-2016	1	5.53483871				-7.897202	13.43204085
4	Février-2016	2	6.13827586				-6.360918	12.49919419
5	Mars-2016	3	6.79129032				-3.920104	10.71139479
6	Avril-2016	4	9.883				-0.925349	10.8083488
7	Mai-2016	5	14.4122581				2.121601	12.29065697
8	Juin-2016	6	17.5766667				6.65589	10.92077651
9	Juillet-2016	7	20.3254839	11.8455	8.480025399	8.554671944	*8.53206*	11.79342366
10	Août-2016	8	20.9196774	11.7438	9.175836816	8.205511647	*8.1829*	12.73677751
11	Septembre-2016	9	18.782	11.9458	6.836175115	4.665681457	*4.64307*	14.13893028
12	Octobre-2016	10	11.4593548	12.1254	-0.666023361	0.310653324	*0.288042*	11.17131325
13	Novembre-2016	11	7.65733333	12.2234	-4.566052483	-4.507845777	*-4.530458*	12.18779085
14	Décembre-2016	12	4.45774194	12.4123	-7.954568164	-6.766919702	*-6.789531*	11.24727337
15	Janvier-2017	13	1.95	12.5464	-10.59635535	-7.874590401	*-7.897202*	9.847202135
16	Février-2017	14	7.28428571	12.511	-5.226760497	-6.338306593	*-6.360918*	13.64520404
17	Mars-2017	15	10.4929032	12.3196	-1.826691372	-3.89749273	*-3.920104*	14.41300769
18	Avril-2017	16	10.4906667	12.289	-1.798336534	-0.90273707	*-0.925349*	11.41601547
19	Mai-2017	17	16.1567742	12.4006	3.756195725	2.144212825	*2.121601*	14.0351731
20	Juin-2017	18	20.3663333	12.4413	7.925039811	6.678501893	*6.65589*	13.71044317
21	Juillet-2017	19	20.7529032	12.7127	8.040211854		8.53206	12.22084302
22	Août-2017	20	19.6448387	12.718	6.926812756		8.1829	11.4619388
23	Septembre-2017	21	15.462	12.3537	3.108316884		4.64307	10.81893028
24	Octobre-2017	22	14.0451613	12.3473	1.697847798		0.288042	13.7571197
25	Novembre-2017	23	7.74933333	12.4867	-4.737390105		-4.530458	12.27979085
26	Décembre-2017	24	5.34290323	12.4539	-7.111007937		-6.789531	12.13243466
27	Janvier-2018	25	7.5783871	12.5162	-4.937792435		-7.897202	15.47558923
28	Février-2018	26	1.78392857	12.6836	-10.89965687		-6.360918	8.144846899
29	Mars-2018	27	7.24903226	12.8115	-5.562454621		-3.920104	11.16913672
30	Avril-2018	28	13.5816667	12.8737	0.707950397		-0.925349	14.50701547
31	Mai-2018	29	16.4116129	12.8811	3.530486687		2.121601	14.29001181
32	Juin-2018	30	19.324	12.9514	6.372600934		6.65589	12.66810984
33	Juillet-2018	31	23.2896774	12.866	10.4236547		8.53206	14.75761721
34	Août-2018	32	21.1258065	12.9599	8.165947805		8.1829	12.94290654
35	Septembre-2018	33	17.0506667	13.2913	3.759324245		4.64307	12.40759694
36	Octobre-2018	34	13.95	13.2942	0.6558117		0.288042	13.66195841
37	Novembre-2018	35	8.02233333	13.0745	-5.0521202		-4.530458	12.55279085

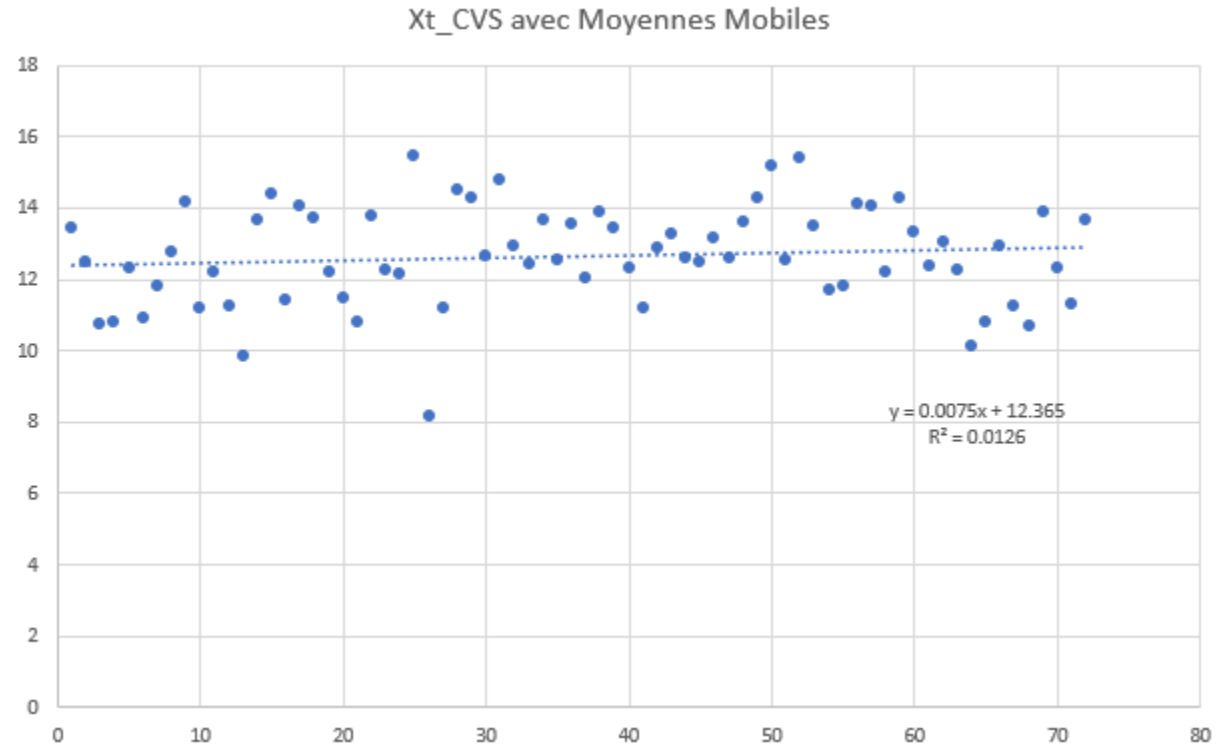
En comparant nos deux résultats sur R et sur Excel, on observe que les résultats sont presque similaires ce qui implique une bonne réalisation de la méthode sur R et Excel.

Série corrigée des variations saisonnières avec Moyennes mobiles

Par la suite de cette étude, on va calculer la série corrigée des variations saisonnières X_t^{cvs} grâce à la formule suivante :

$$S^*_T = \{c^*_j, j \in T_j\}, X_t^{cvs} = X_t - S^*_t$$

Par construction graphique sur Excel, on obtient le résultat suivant :



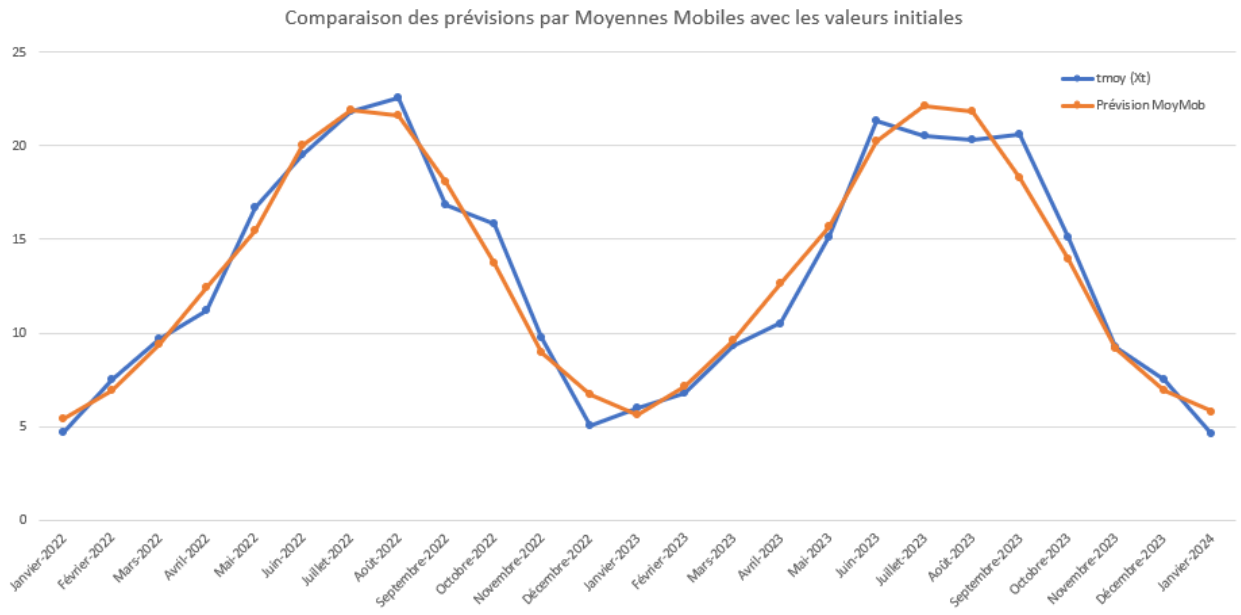
Puis, avec le logiciel R, on obtient :

```
#Régression linéaire sur les données lissées par moyenne mobile

Xt_CVS <- read.xlsx("Tableur_MONIN_Louis_LAMRINI_Mohamed_ING2MI2_Serie_Temporelle_Projet.xlsx", sheet = "Xt_CVS MoyMob")
View(Xt_CVS)

reg_moymob <- lm(`Xt_CSV(Additif)` ~ t, data = Xt_CVS)
summary(reg_moymob)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 12.364912   0.331760  37.271   <2e-16 ***
t            0.007465   0.007899   0.945    0.348
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.393 on 70 degrees of freedom
Multiple R-squared:  0.0126,    Adjusted R-squared:  -0.001507
F-statistic: 0.8932 on 1 and 70 DF,  p-value: 0.3479
```

On obtient exactement les mêmes résultats avec l'intercept égale à 11.99, le coefficient directeur de la droite de régression linéaire est égal à 0.01 et un R^2 égale à 0.003, ce qui signifie qu'il y a une faible corrélation entre la série CVS et l'indice de temps t.

Prévision par moyenne mobile



Nous constatons que les prédictions de notre modèle sur les données de validations sont très proches de la réalité car l'écart entre le courbe tmoy (Xt) et la courbe des prévisions MoyMob est faible : la méthode des moyennes mobiles est relativement efficace.

2ème méthode : Régression sur le temps et avec des variables indicatrices

Calculs des coefficients saisonniers par Moyenne mobile

Pour la suite de cette étude, nous allons effectuer la même chose mais avec la méthode des moyenne mobiles.

	A	B	C	D	E	F	G	H	I	J	K	L
1					Lissage Régression linéaire							
2	Mois-Annee	t	tmoy (Xt)	St = Ct (provisoire)	Ct* (modifié)	M(t)=B(0)+B(1)*t	Xt_CVS (Additif)					
3	Janvier-2016	1	5.53483871	4.997903226	-7.639479852	12.00930296	4.369823111					
4	Février-2016	2	6.138275862	6.36058087	-6.276802208	12.02699536	5.750193153			Moyenne(t)	36.5	
5	Mars-2016	3	6.791290323	8.487311828	-4.15007125	12.04468776	7.894616508			Moyenne(Xt)	12.63738308	
6	Avril-2016	4	9.883	11.50419349	-1.133189592	12.06238016	10.92919056			Cov(t,Xt)	7.74927015	
7	Mai-2016	5	14.41225806	14.79973118	2.162348105	12.08007255	14.24242066			Var(t)	438	
8	Juin-2016	6	17.57666667	19.11627778	6.4788947	12.09776495	18.57665965			Moyenne des St	12.63738308	
9	Juillet-2016	7	20.32548387	21.04801075	8.410627675	12.11545735	20.52608502			Moyenne des Ct*	1.62833E-15	
10	Août-2016	8	20.91967742	20.60145161	7.964068535	12.13314975	20.09721828					
11	Septembre-2016	9	18.782	17.60844444	4.971061366	12.15084214	17.12190351					
12	Octobre-2016	10	11.45935484	12.99548387	0.358100793	12.16853454	12.52663533			alpha=	11.99161057	
13	Novembre-2016	11	7.657333333	7.993777778	-4.6436053	12.18622694	7.542621639			bêta=	0.017692398	
14	Décembre-2016	12	4.457741935	6.135430108	-6.501952971	12.20391934	5.701966366					
15	Janvier-2017	13	1.95		-7.639479852	12.22161173	4.582131882					
16	Février-2017	14	7.284285714		-6.276802208	12.23930413	5.962501924					
17	Mars-2017	15	10.49290323		-4.15007125	12.25699653	8.10692528					
18	Avril-2017	16	10.49066667		-1.133189592	12.27468893	11.14149934					
19	Mai-2017	17	16.15677419		2.162348105	12.29238132	14.45472943					
20	Juin-2017	18	20.36633333		6.4788947	12.31007372	18.78896842					
21	Juillet-2017	19	20.75290323		8.410627675	12.32776612	20.73839379					
22	Août-2017	20	19.64483871		7.964068535	12.34545852	20.30952705					
23	Septembre-2017	21	15.462		4.971061366	12.36315092	17.33421228					
24	Octobre-2017	22	14.04516129		0.358100793	12.38084331	12.73894411					
25	Novembre-2017	23	7.749333333		-4.6436053	12.39853571	7.75493041					
26	Décembre-2017	24	5.342903226		-6.501952971	12.41622811	5.914275137					
27	Janvier-2018	25	7.578387097		-7.639479852	12.43392051	4.794440653					
28	Février-2018	26	1.783928571		-6.276802208	12.4516129	6.174810695					
29	Mars-2018	27	7.249032258		-4.15007125	12.4693053	8.319234051					
30	Avril-2018	28	13.58166667		-1.133189592	12.4869977	11.35380811					
31	Mai-2018	29	16.4116129		2.162348105	12.5046901	14.6670382					
32	Juin-2018	30	19.324		6.4788947	12.52238249	19.00127719					
33	Juillet-2018	31	23.28967742		8.410627675	12.54007489	20.95070257					

Pour calculer les c_t provisoires, on prend la moyenne de chaque mois de l'année en question en utilisant les températures enregistrées en 2016 et 2021. Si la moyenne des c_t n'est pas égale à zéro, on doit soustraire la moyenne de chaque c_t individuel pour obtenir les valeurs modifiées dans la colonnes c_t^*.

Une fois les valeurs modifiées obtenues, on peut effectuer une régression pour trouver les valeurs α et de β en fonction du temps en utilisant les ventes enregistrées sur la période spécifiée.

$$S_T^* = \{c_j^*, j \in T_j\}, \quad c_j^* = S_j - \bar{S_j}$$

On effectue la même démarche sur R et on obtient les résultats suivants.

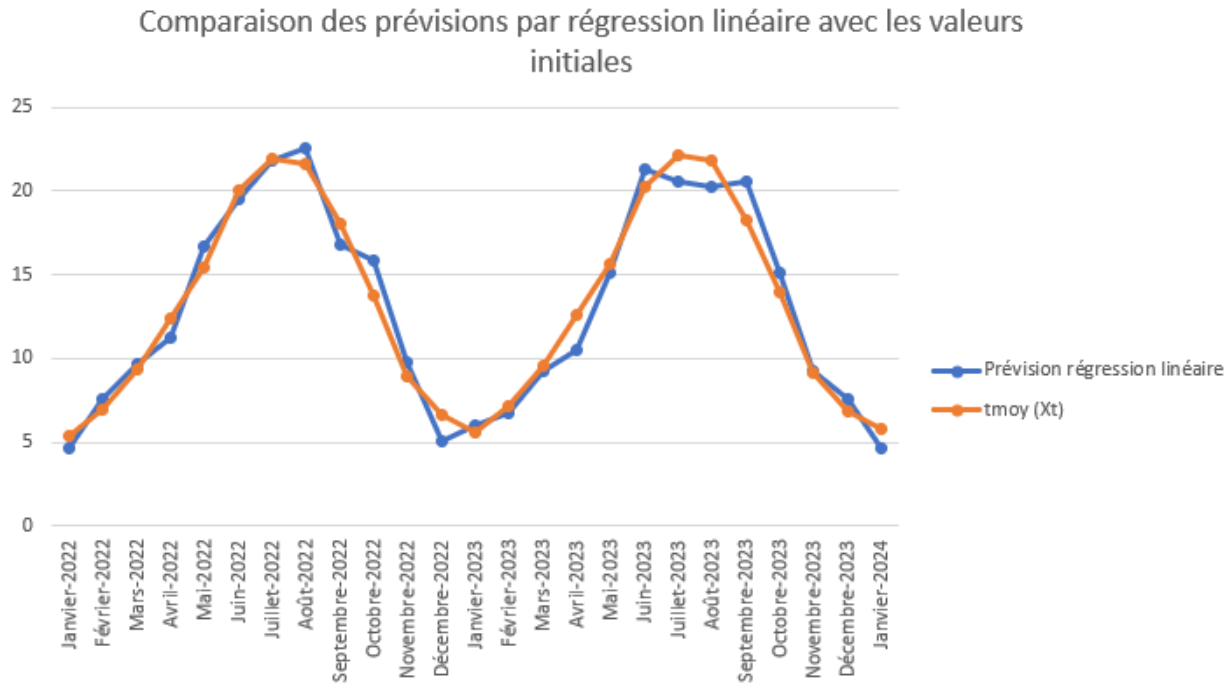
```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 11.99161    1.40608   8.528 1.94e-12 ***
t            0.01769    0.03348   0.529    0.599
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.903 on 70 degrees of freedom
Multiple R-squared:  0.003974,  Adjusted R-squared:  -0.01025
F-statistic: 0.2793 on 1 and 70 DF,  p-value: 0.5988
```

On remarque tout de suite que les résultats obtenus sont les mêmes sur R et Excel.

Prévisions par régression linéaire

Pour calculer les c_t provisoires, on prend la moyenne de chaque mois de l'année en question en utilisant les ventes enregistrées entre 2022 et 2024. Si la moyenne des c_t n'est pas égale à zéro, on doit soustraire la moyenne de chaque c_t individuel pour obtenir les valeurs modifiées dans la colonne $c_t{}^*$. Une fois les valeurs modifiées obtenues, on peut effectuer une régression pour trouver les valeurs de α et de β en fonction des ventes en utilisant les ventes enregistrées sur la période spécifiée. Les prévisions sont trouvées grâce à la formule :

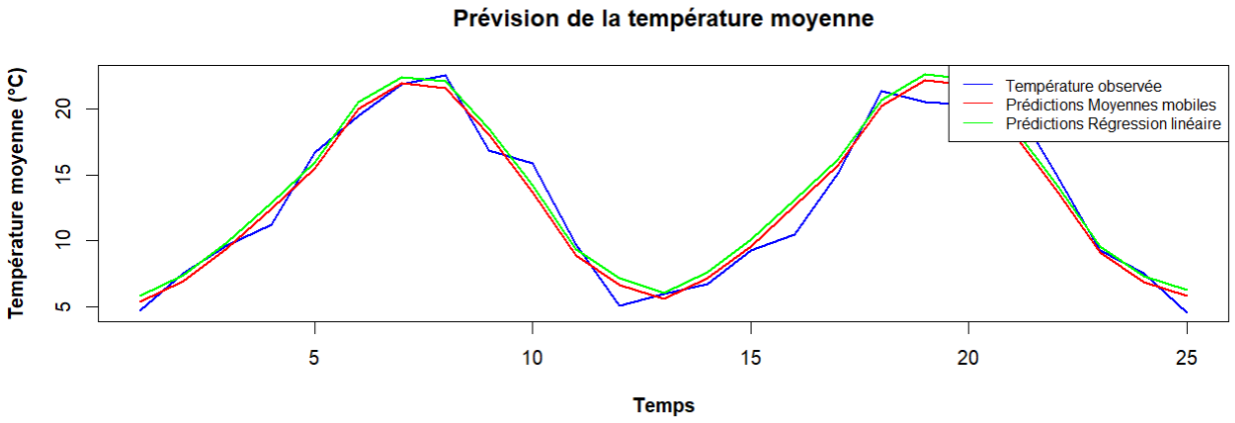
$$X_t{}^{cvs} = \alpha + \beta t + c_t{}^*$$



Nous pouvons ainsi constater que la méthode de régression linéaire fournit une prédiction précise de la réalité.

Comparaison des deux méthodes

```
par(mfcol=c(1,1))
plot(test$tmoy, type = "l",lwd=2.0,col="blue",xlab="Temps",font.lab= 2, ylab="Température moyenne (°C)",main="Prévision de la température moyenne")
lines(prevision_moymob$Prevision_MoyMob + 0.46, type = "l", col = "green", lwd = 2.0)
lines(prevision_moymob$Prevision_MoyMob, type = "l",col="red",lwd=2.0)
legend("topright", legend=c("Température observée", "Prédictions Moyennes mobiles", "Prédictions Régression linéaire"), col=c("blue", "red", "green"), lty=1, cex=0.8)
```



Lorsque l'on trace les courbes obtenues, on constate que les courbes des deux prédictions sont presque superposées, donc identiques, ce qui rend difficile leur comparaison. La courbe de régression linéaire est plus adaptée pour étudier les tendances à court et très court terme, tandis que la moyenne mobile convient mieux pour les moyens et longs termes. A vu d'œil, on observer une très légère victoire pour la régression linéaire, mais qui n'est clairement pas importante au vu des résultats.

Partie 3 : Prévision par lisaage exponentiel

1ère méthode : Lissage exponentiel double

Lissage

On a opté pour le lissage exponentiel double en raison de la présence d'une tendance dans notre ensemble de données. Pour déterminer les coefficients saisonniers, nous avons procédé de la même manière que précédemment en utilisant la moyenne des ventes pour chaque période mensuelle.

De plus, on a choisi un α grand ($\alpha > 0.7$) pour donner moins d'importance aux observations les plus anciennes.

Le premier lissage peut être calculé par :

$$L_1(t) = X_t\alpha + (1 - \alpha) + L_1(t - 1)$$

Le deuxième lissage peut être calculé par :

$$L_2(t) = L_1(t)\alpha + (1 - \alpha) + L_2(t - 1)$$

Ensuite, on calcule a et b avec les formules suivantes :

$$a = (\alpha / (1 - \alpha)) * (L_1(t) - L_2(t))$$

$$b = 2L_1(t) - L_2(t)$$

Avec l'utilisation d'Excel, on obtient les résultats suivants :

D	E	F	G	H	I	J	K	L	M	N
		Lissage exponentiel double								
St=Ct (provisoire	Ct* (modifié	1er lissage (L1(t))	2ème lissage (L2(t	a = â2	b = â1	Prévision Lissage exp double				
4.997903226	-7.639479852	5.53483871	5.53483871	0	5.5348	5.53483871				
6.36058087	-6.276802208	6.077932147	6.023622803	0.4888	6.1322	6.621025584			Moyenne des St	12.6374
8.487311828	-4.15007125	6.719954505	6.650321335	0.6267	6.7896	7.416286207			alpha (donné)	0.9
11.50419349	-1.133189592	9.566695451	9.275058039	2.6247	9.8583	12.48306957				
14.79973118	2.162348105	13.9277018	13.46243743	4.1874	14.393	18.58034557			Moyenne(t)	36.5
19.11627778	6.4788947	17.21177018	16.8368369	3.3744	17.587	20.96110293			Moyenne(Xt)	12.6585
21.04801075	8.410627675	20.0141125	19.69638494	2.8595	20.332	23.1913881			Cov(t,Xt)	1.33298
20.60145161	7.964068535	20.82912093	20.71584733	1.0195	20.942	21.96185691			Var(t)	438
17.60844444	4.971061366	18.98671209	19.15962562	-1.5562	18.814	17.25757686				
12.99548387	0.358100793	12.21209056	12.90684407	-6.2528	11.517	5.264555512			alpha=	12.5474
7.993777778	-4.6436053	8.112809056	8.592212558	-4.3146	7.6334	3.318774043			bêta=	0.00304
6.135430108	-6.501952971	4.823248648	5.200145039	-3.3921	4.4464	1.054284737				
	-7.639479852	2.237324865	2.533606882	-2.6665	1.941	-0.725495309				
	-6.276802208	6.779589629	6.354991355	3.8214	7.2042	11.02557238				
	-4.15007125	10.12157187	9.744913815	3.3899	10.498	13.88815238				
	-1.133189592	10.45375719	10.38287285	0.638	10.525	11.16260056				
	2.162348105	15.58647249	15.06611253	4.6832	16.107	20.79007214				
	6.4788947	19.88834725	19.40612378	4.34	20.371	24.71058197				
	8.410627675	20.66644763	20.54041524	1.1343	20.792	21.92677148				
	7.964068535	19.7469996	19.82634117	-0.7141	19.668	18.95358396				
	4.971061366	15.89049996	16.28408408	-3.5423	15.497	11.95465875				
	0.358100793	14.22969516	14.43513405	-1.849	14.024	12.17530623				
	-4.6436053	8.397369516	9.001145969	-5.434	7.7936	2.359604982				

Puis, avec l'utilisation de R, on obtient :

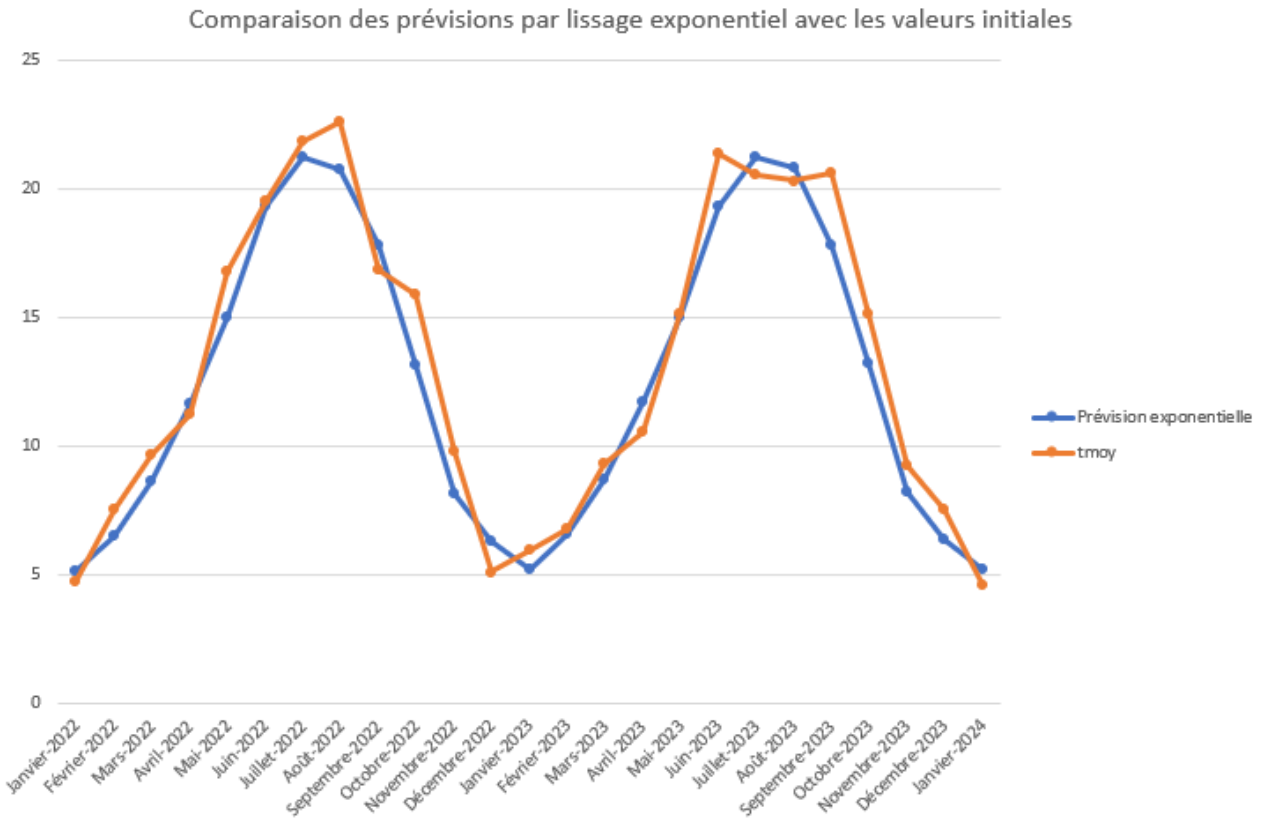
```
> fitLED <- ets(ts_train,model="AAN")
> summary(fitLED)
ETS(A,Ad,N)

Call:
 ets(y = ts_train, model = "AAN")

  Smoothing parameters:
    alpha = 0.8343
    beta  = 0.8343
    phi   = 0.8057

  Initial states:
    l = 3.2042
    b = 3.76

  sigma:  3.3312

     AIC     AICc      BIC
396.8394 398.4243 409.4055

Training set error measures:
                     ME     RMSE      MAE       MPE     MAPE     MASE        ACF1
Training set -0.09533787 3.189368 2.498579 -2.024306 27.97603 1.393125 0.01367607
```

Prévisions

Les prévisions sont calculées avec la formule suivante :

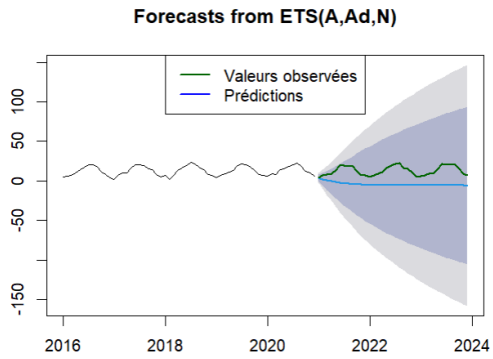
$$X_t{}^{exp} = a_{t-1} - b_{t-1}$$

Ainsi, on obtient les prédictions suivantes avec Excel :



En poursuivant cette étude, on obtient le résultat suivant sur R :

```
fitLED <- ets(ts_train,model="AAN")
summary(fitLED)
predLED <- forecast(fitLED,h=36)
plot(predLED)
points(ts_test,type='l',col='darkgreen',lwd=2)
legend('top',c("Valeurs observées","Prédictions"),col=c('darkgreen',"blue"),lty=rep(1,2),lwd = rep(2,2))
```



On remarque que le modèle étant plus général, l'erreur moyenne est nécessairement inférieure mais cette méthode ne semble pas préférable car le paramètre β estimé est très petit, l'AIC, l'AICc et le BIC sont plus grands et l'erreur moyenne relative est à peu près du même ordre.

2ème méthode : Lissage exponentiel avec Holt-Winters

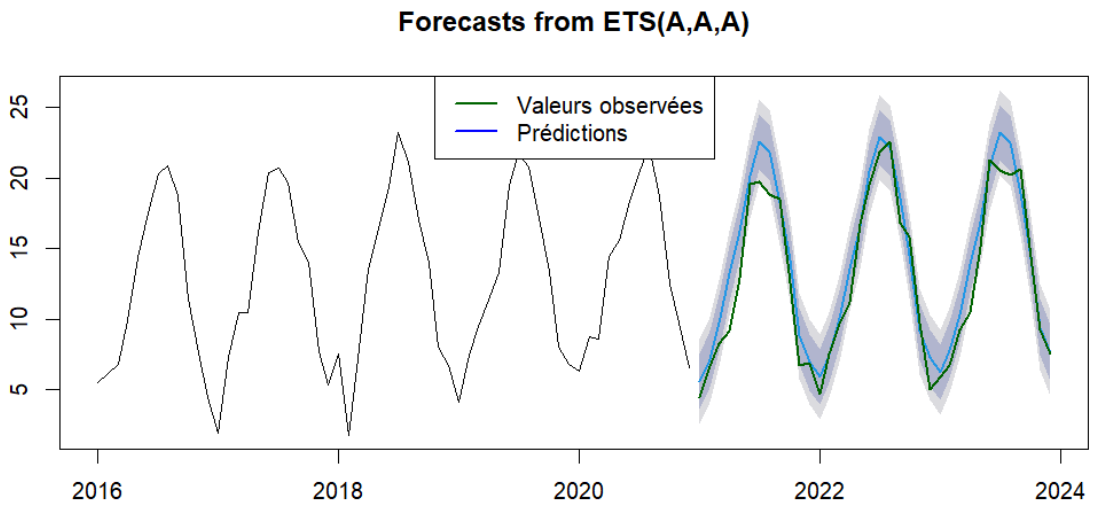
Lissage

Contrairement à la méthode précédente, le paramètre β estimé est élevé, l'AIC, l'AICc et le BIC sont assez faible et de même pour l'erreur moyenne.

```
> fitHW <- ets(ts_train,model="AAA")
> summary(fitHW)
ETS(A,A,A)

Call:
 ets(y = ts_train, model = "AAA")

  Smoothing parameters:
    alpha = 3e-04
    beta  = 1e-04
    gamma = 1e-04

  Initial states:
    l = 11.9532
    b = 0.0271
    s = -6.9356 -5.044 0.4254 4.5068 8.0751 8.847
           6.3936 2.5355 -0.4558 -3.9335 -6.4414 -7.9733

  sigma:  1.5366

     AIC     AICc      BIC
312.6014 327.1728 348.2052

Training set error measures:
                     ME     RMSE      MAE       MPE     MAPE      MASE
Training set -0.01754173 1.31588 1.052914 -5.119992 15.07378 0.5870705
                    ACF1
Training set -0.07019371
```

Prévision

```
  #Méthode 2 : Lissage Holt-Winters

fitHW <- ets(ts_train,model="AAA")
summary(fitHW)
predHW <- forecast(fitHW,h=36)
plot(predHW)
points(ts_test,type='l',col='darkgreen',lwd=2)
legend('top',c("Valeurs observées","Prédictions"),col=c("darkgreen","blue"),lty=rep(1,2),lwd = rep(2,2))
```



Comparaison des deux méthodes

Après avoir comparé notre lissage exponentiel double et celui de Holt-Winters, nous avons remarqué qu'ils présentent quelques différences notables, bien que tous deux soient fiables. Si nous devions choisir une prévision entre les deux, nous opterions pour notre lissage exponentiel double, car il s'approche plus fidèlement du modèle réel sur plusieurs parties de la courbe. Néanmoins, il faut garder à l'esprit que les deux prévisions sont très similaires, mais notre lissage exponentiel double présente une légère avance en termes de précision.

Annexe

```python
import pandas as pd

# Charger le fichier CSV
df = pd.read_csv('temperature-quotidienne-regionale.csv')

# Filtrer les lignes qui contiennent le mot "Île-de-France" dans la première colonne
df_idf = df[df.iloc[:, 0].str.contains('Île-de-France', case=False, na=False)]

# Trier les lignes par ordre de dates
df_idf_sorted = df_idf.sort_values(by='Nom_de_la_colonne_date')

# Enregistrer le résultat dans un nouveau fichier CSV
df_idf_sorted.to_csv('idf_seulement.csv', index=False)

# Charger le fichier CSV en spécifiant le délimiteur comme ';'
df = pd.read_csv('idf_seulement.csv', delimiter=';')

# Renommer les colonnes
df.columns = ['ID', 'Date', 'code_insee_region', 'region', 'tmin', 'tmax', 'tmoy']

# Supprimer les colonnes spécifiées
colonnes_a_supprimer = ['code_insee_region', 'region', 'tmin', 'tmax']
df = df.drop(columns=colonnes_a_supprimer)

# Enregistrer le fichier modifié
df.to_csv('Temperature_moyenne_ile_de_france.csv', index=False)

#Enfin nous avons fait une moyenne par mois des températures.

# Charger les données
df = pd.read_csv('Temperature_moyenne_ile_de_france.csv')

# Convertir les dates en datetime pour extraire facilement l'année et le mois
df['Date'] = pd.to_datetime(df['Date'], format='%d/%m/%y')
```

```python
# Ajouter des colonnes pour l'année et le mois
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
# Dictionnaire pour le nom des mois en français
months_fr = {
  1: 'Janvier', 2: 'Février', 3: 'Mars', 4: 'Avril', 5: 'Mai', 6: 'Juin',
  7: 'Juillet', 8: 'Août', 9: 'Septembre', 10: 'Octobre', 11: 'Novembre', 12: 'Décembre'
}

# Calculer la moyenne des températures par mois et par année
monthly_avg = df.groupby(['Year', 'Month'])['tmoy'].mean().reset_index()

# Convertir le numéro du mois en son nom en français et ajouter l'année
monthly_avg['Mois-Annee'] = monthly_avg.apply(lambda x: f"{months_fr[x['Month']]}-{x['Year']}", axis=1)

# Sélectionner les colonnes d'intérêt pour le fichier final
result = monthly_avg[['Mois-Annee', 'tmoy']]

# Sauvegarder le résultat dans un nouveau fichier Excel
result.to_excel('Temperature_moyenne_ile_de_france_v2.xlsx', index=False)
```

[1]