Code

Projet Série Temporelle

SCHULTZ Martin - Master SISE - Université Lyon 2
Janvier 2023

Utilisation des librairies nécessaires

Hide

```
library("readxl")
library("forecast")
library("fpp2")
library("ggplot2")
library("dplyr")
library("xlsx")
```

Importation des données

Hide

```
data = read_excel("Elec-train.xlsx")
```

La fréquence est de 96 car les données ont été mesuré toutes les 15min.
1jour = 96 * 15min

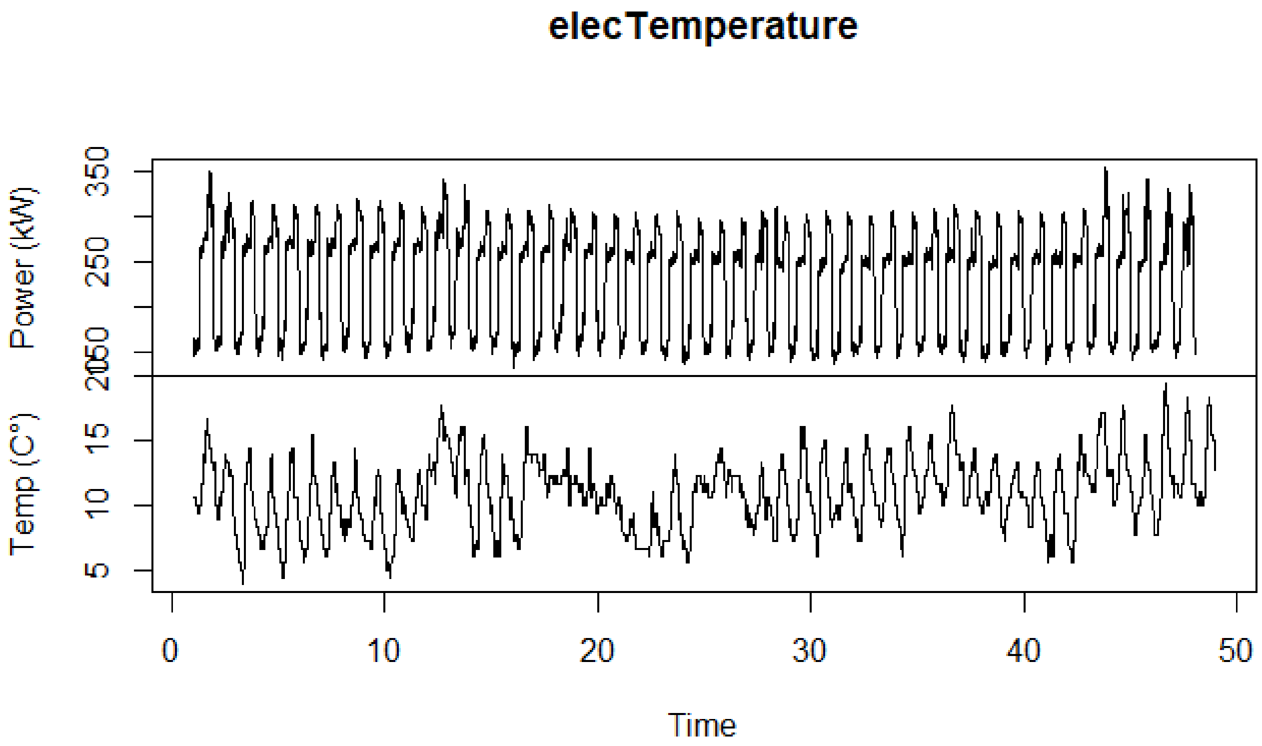
Hide

```
data=mutate(data, Timestamp = as.POSIXct(Timestamp,format = "%m/%d/%Y %H:%M"))
elec<- ts(data[,2], start = c(1,6), end = c(47,96),frequency = 96)
elecTemperature=ts(data[,2:3], start = c(1,6), end = c(48,96),frequency = 96)
```

Analyse des donénes

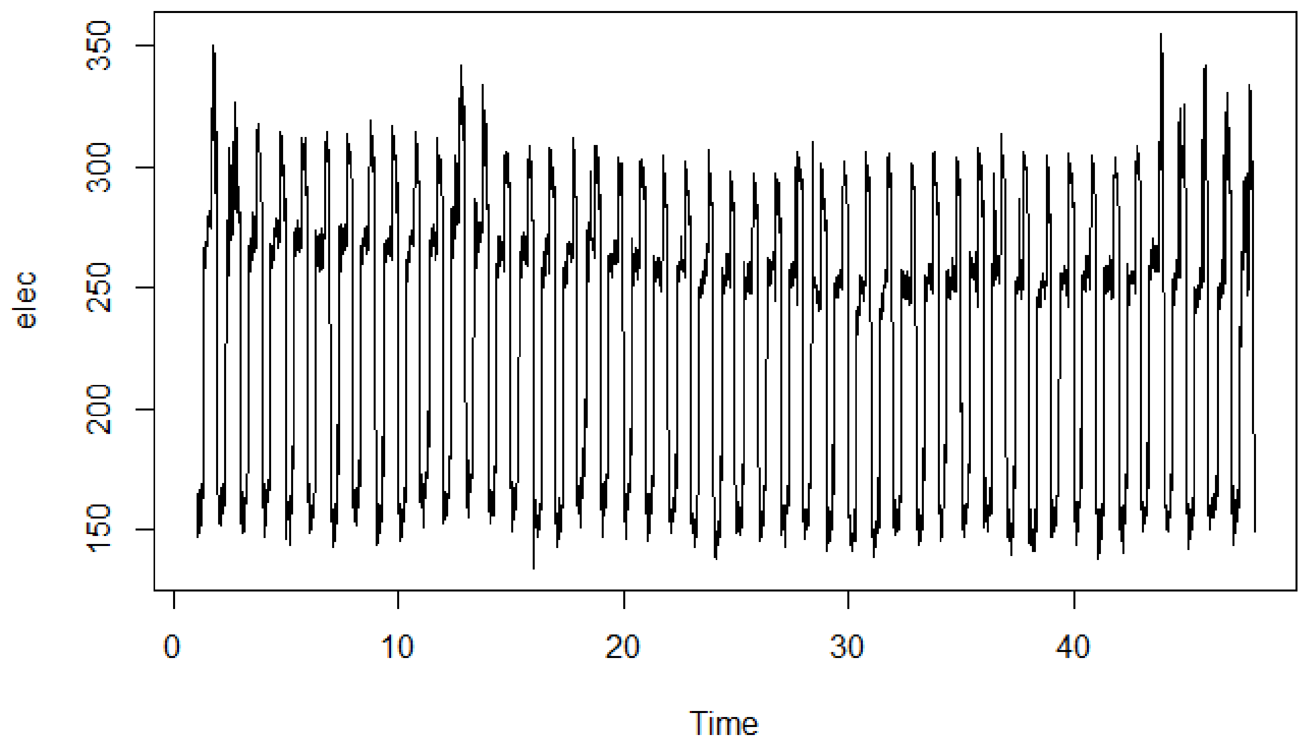
Hide

```
plot(elecTemperature)
```



Hide

```
plot(elec)
```



Il semble avoir une saisonnalité dans la consommation électrique, Mais il ne semble pas qu'il existe une tendance, car la consommation électrique ne semble ni augmenter ni diminiuer.

Division des données en échantillons d’apprentissage et de test.

L’échantillon de test représente le dernier jour de consommation.

Hide

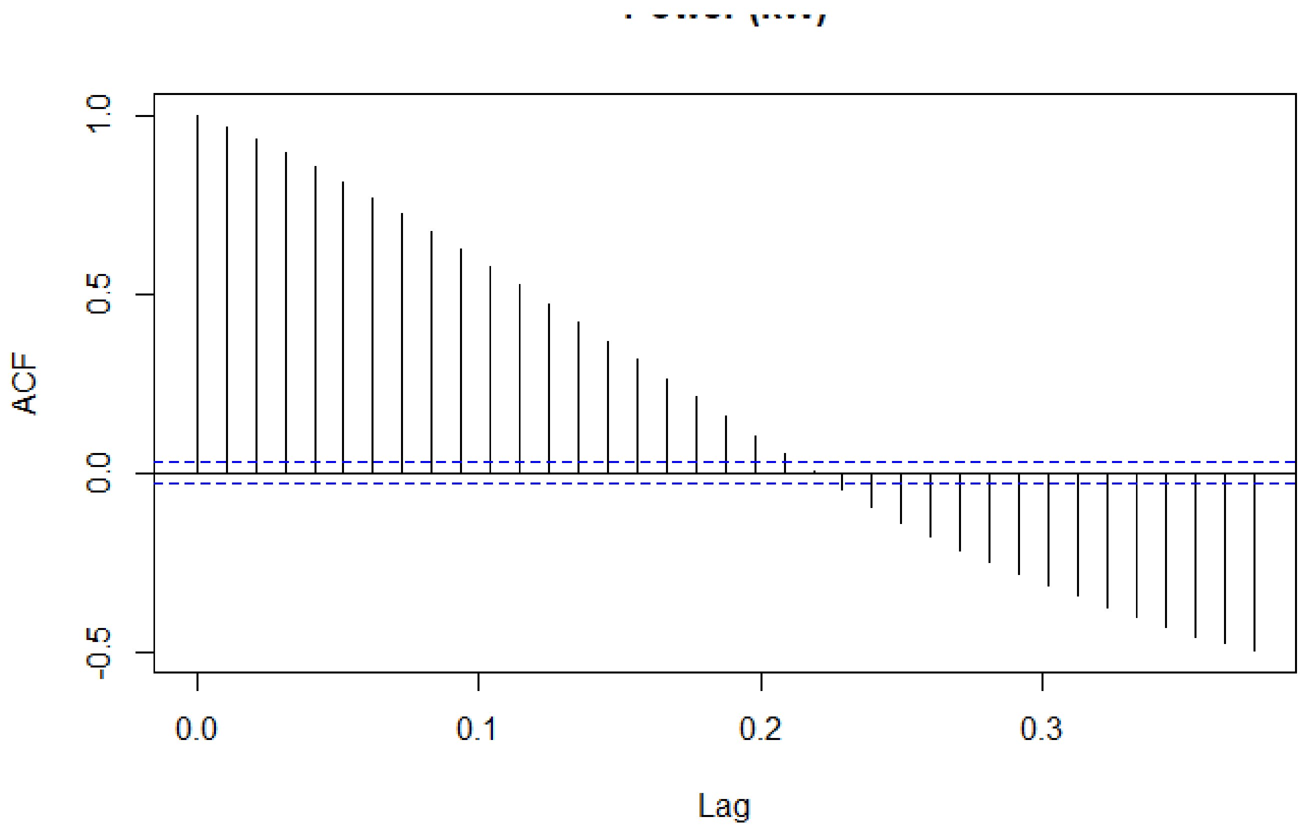
```
elec_train = window(elec, start=c(1,6), end=c(46,96))
elec_test = window(elec, start=c(47,1), end=c(47,96))
elec_prev = window(elec, start=c(1,6))

elecTemperature_train = window(elecTemperature, start=c(1,6), end=c(46,96))
elecTemperature_test = window(elecTemperature, start=c(47,1), end=c(47,96))
elecTemperature_prev = window(elecTemperature, start=c(1,6))
```

On commence la série à partir de 6 car la première valeur à été prise à 1h15. (6ième quarts-heure de la journée)

Sans Utilisation de la variable Temperature

Hide

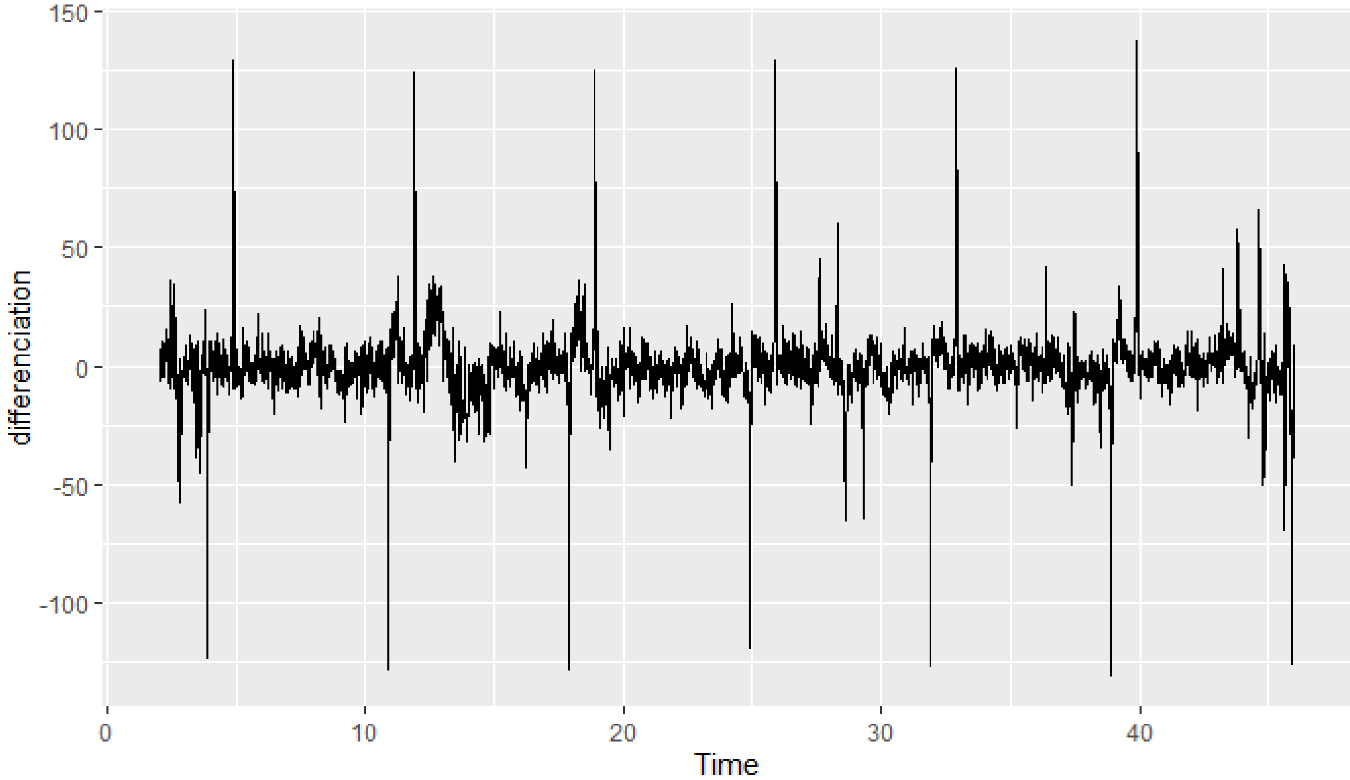
```
tmp=acf(elec_train,type="cor",plot = FALSE)
plot(tmp)
```



Grâce à notre ACF, je suis sûr que la serie a une Saisonnalité, car la fonction décroit exponentiellement

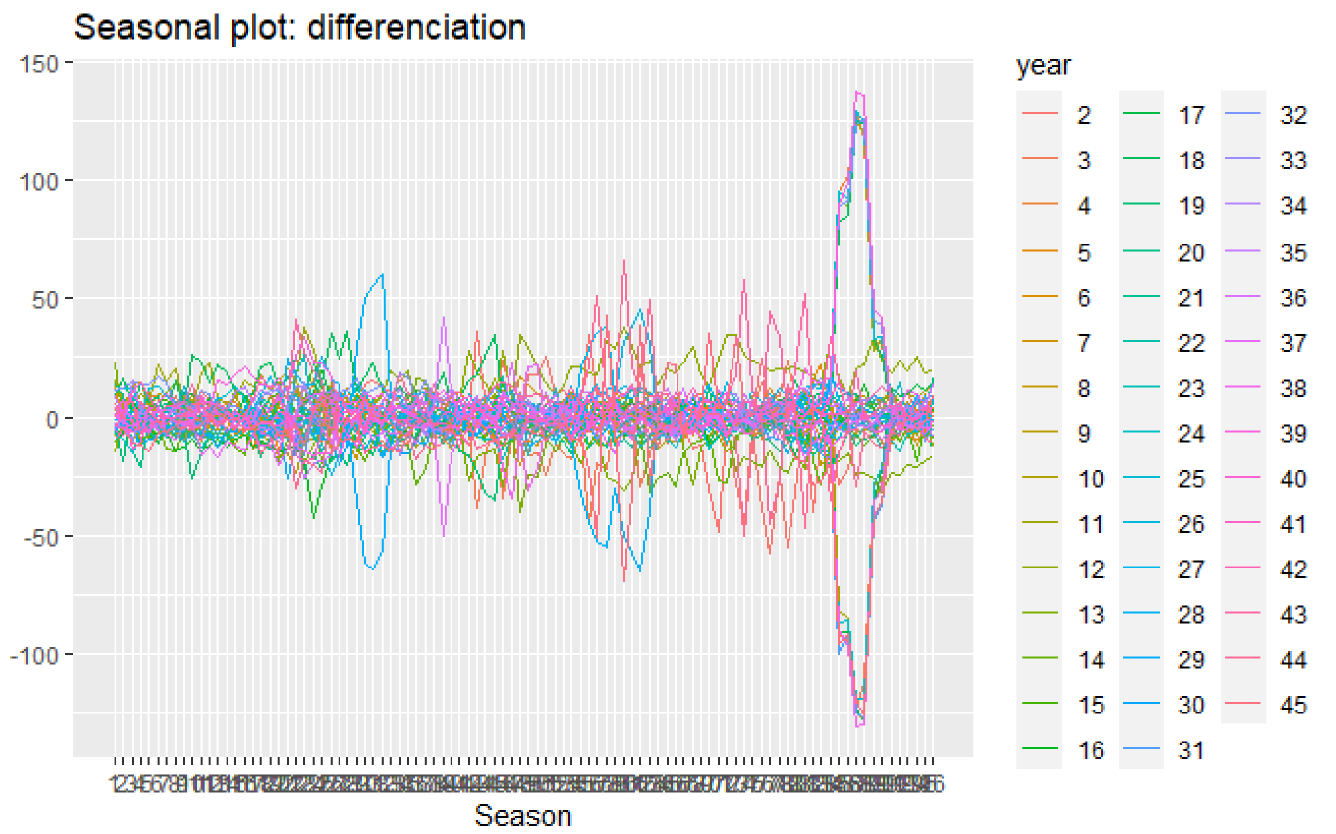
Hide

```
differenciation=diff(elec_train,lag=96)
autoplot(differenciation)
```



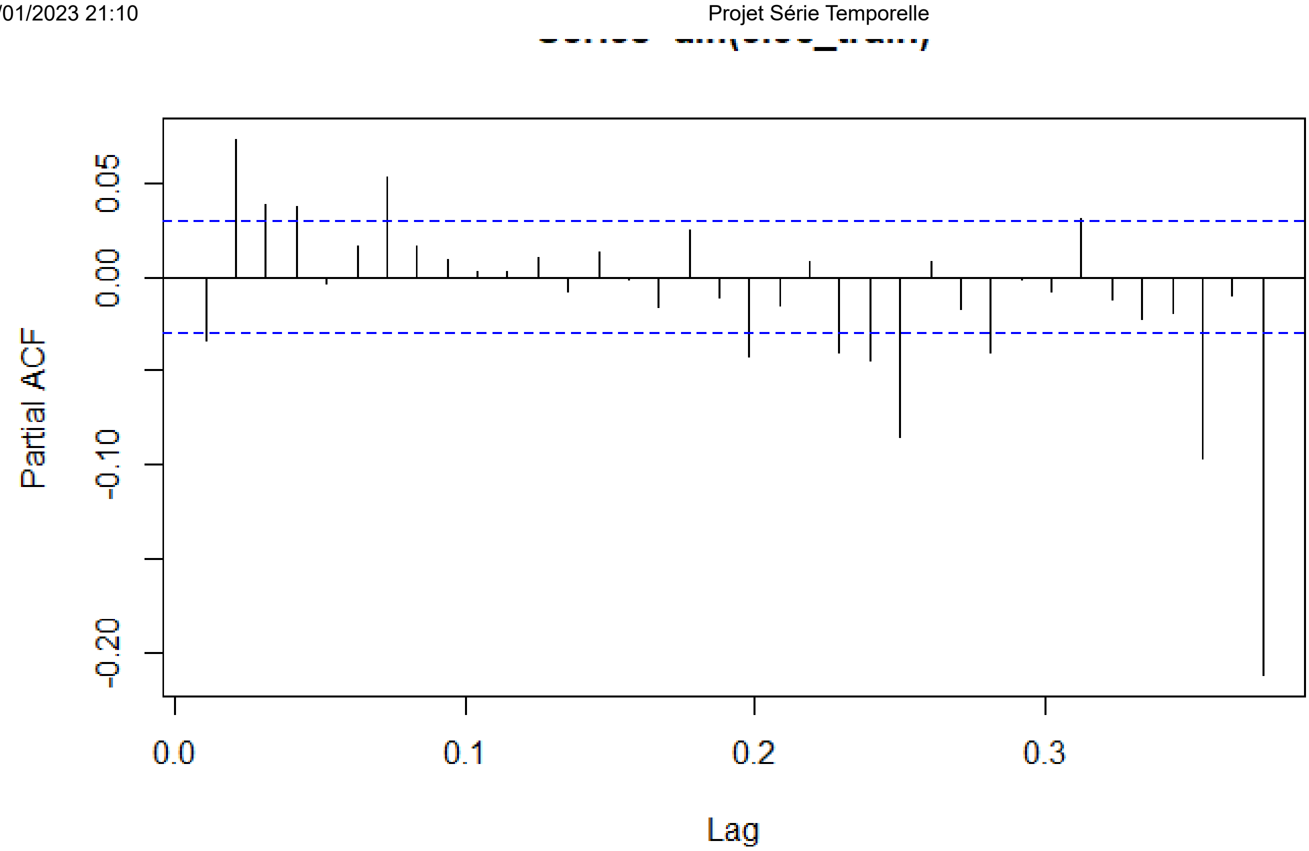
Hide

```
ggseasonplot(differenciation)
```



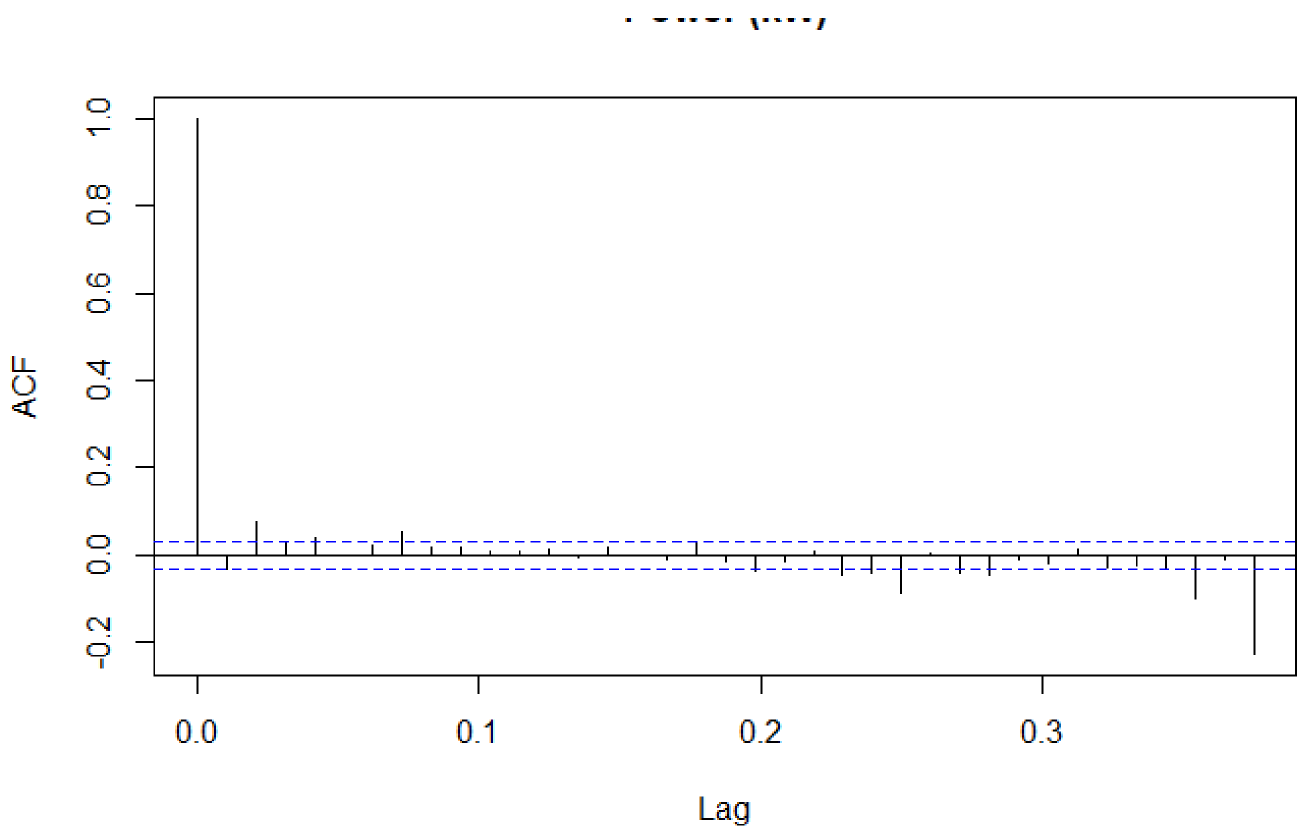
Hide

```
diffpacf=pacf(diff(elec_train),type="cor",plot = FALSE)
plot(diffpacf)
```



Hide

```
diffacf=acf(diff(elec_train),type="cor",plot = FALSE)
plot(diffacf)
```



SARIMA

On va essayer d'utiliser les modèles auto-régressifs :

En consultant le PACF, on voit un pic sur le lag 2, On essaye le modèle suivant :

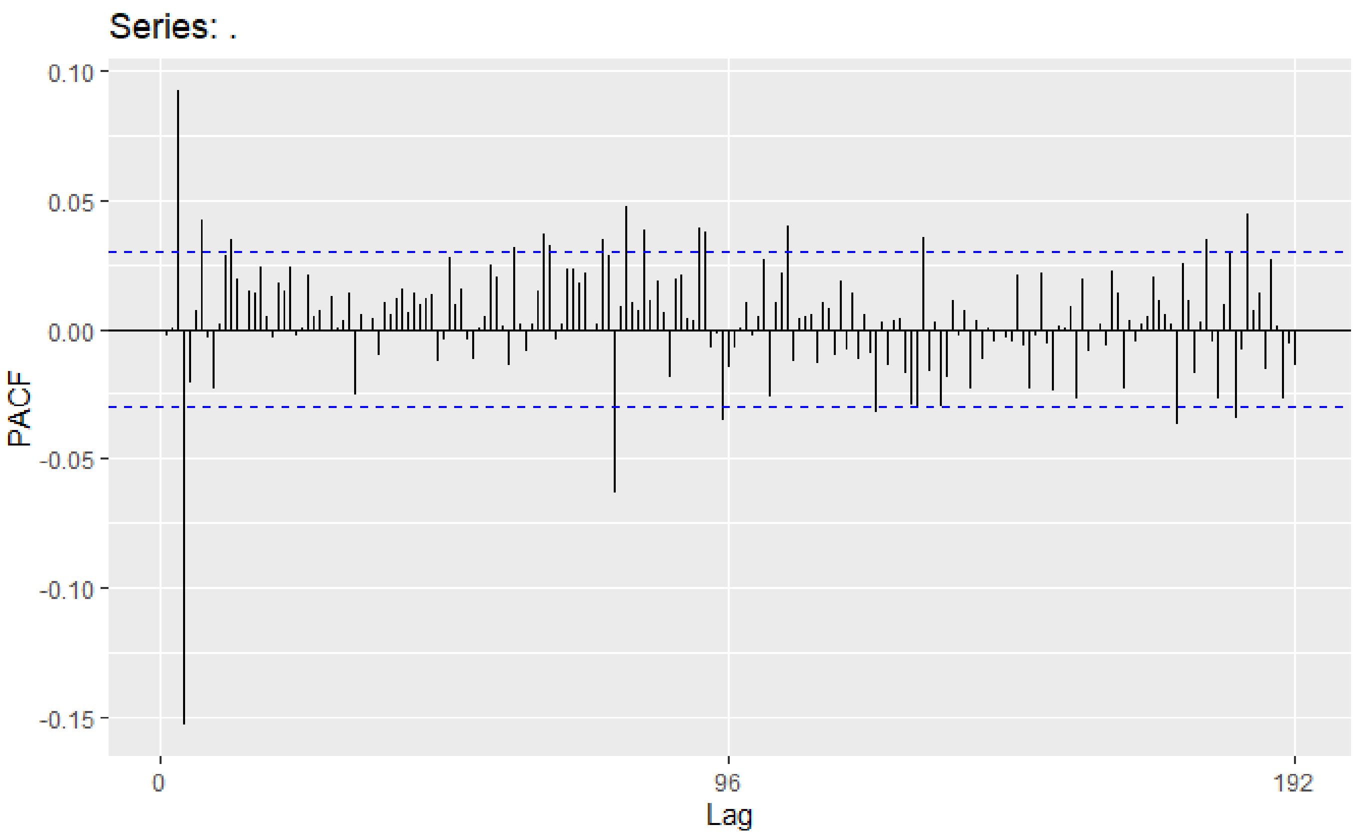
Hide

```
prev= forecast(fitAuto)
cat("RMSE", sqrt(mean(elec_test-prev$mean)^2))
```

```
RMSE 9.129775
```

Hide

```
fit200 %>% residuals() %>% ggPacf()
```



On voit un pic sur le lag 3, on va essayer d'améliorer les performances de notre modèle :

Hide

```
fit300=Arima(elec_train, order=c(3,0,0), seasonal=c(0,1,1))#96
summary(fit300)
```

```
Series: elec_train
ARIMA(3,0,0)(0,1,1)[96]

Coefficients:
         ar1     ar2     ar3     sma1
      0.7070  0.0758  0.0107  -0.8665
s.e.  0.0154  0.0188  0.0154   0.0085

sigma^2 = 63.43:  log likelihood = -14805.9
AIC=29621.8   AICc=29621.81   BIC=29653.54

Training set error measures:
                     ME     RMSE      MAE        MPE     MAPE      MASE          ACF1
Training set -0.3655502 7.871433 4.692265 -0.2530029 2.144352 0.5750232 -0.0006245872
```

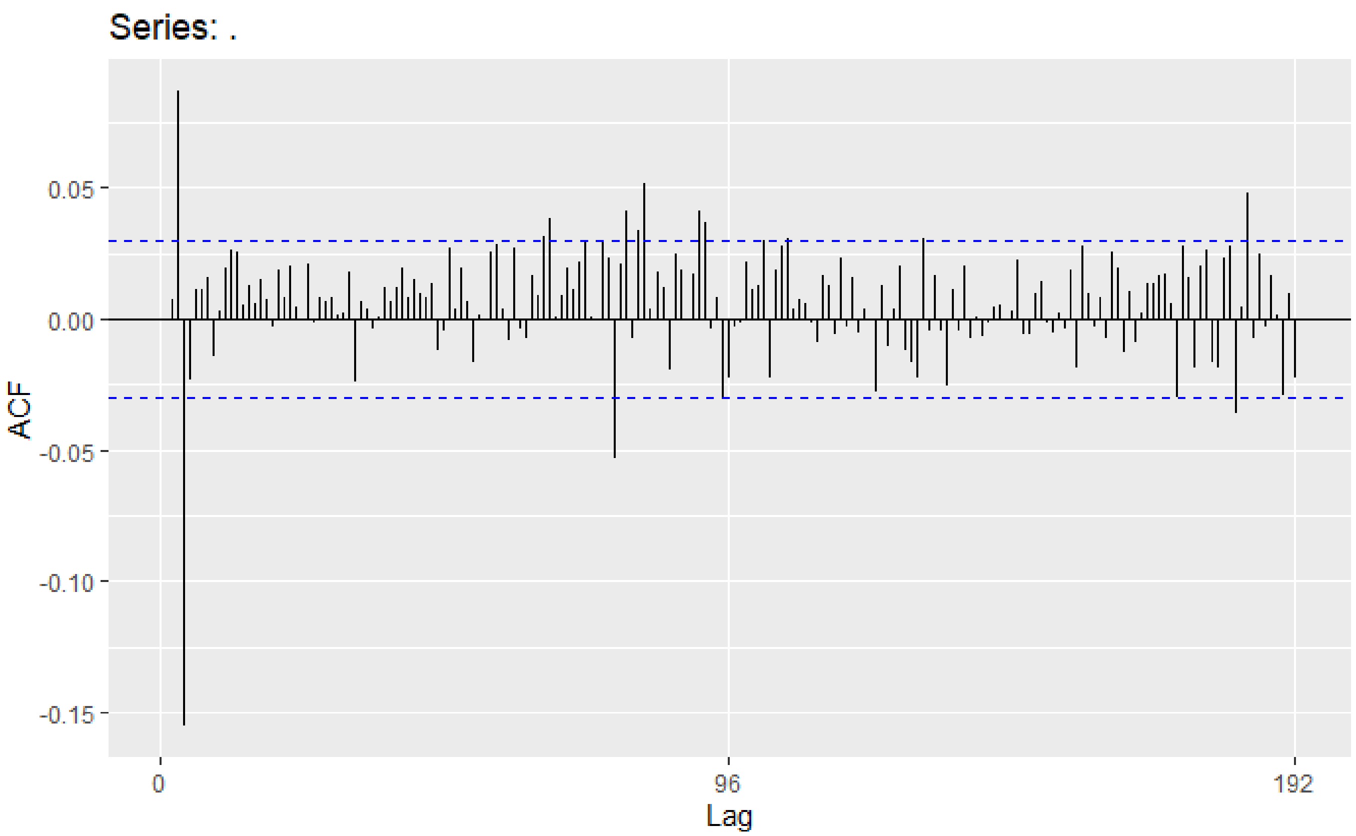
Hide

```
prevfit300= forecast(fit300)
cat("RMSE", sqrt(mean(elec_test-prevfit300$mean)^2))
```

```
RMSE 4.005757
```

Hide

```
fit300 %>% residuals() %>% ggAcf()
```



On voit un pic sur le ACF, sur le lag 3 :

Hide

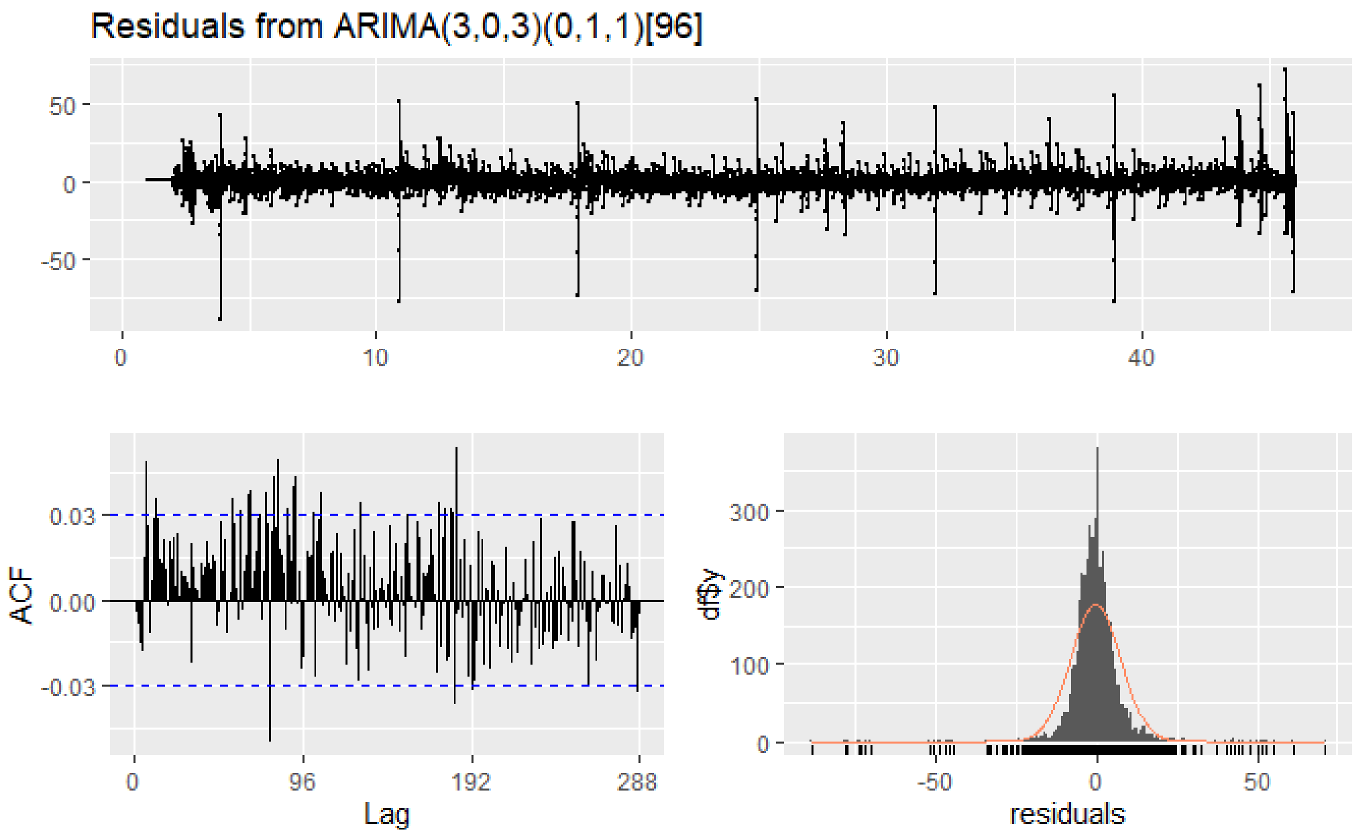
```
fit303=Arima(elec_train, order=c(3,0,3), seasonal=c(0,1,1))#96
prevfit303 = forecast(fit303)
cat("RMSE", sqrt(mean(elec_test-prevfit303$mean)^2))
```

```
RMSE 3.982057
```

Hide

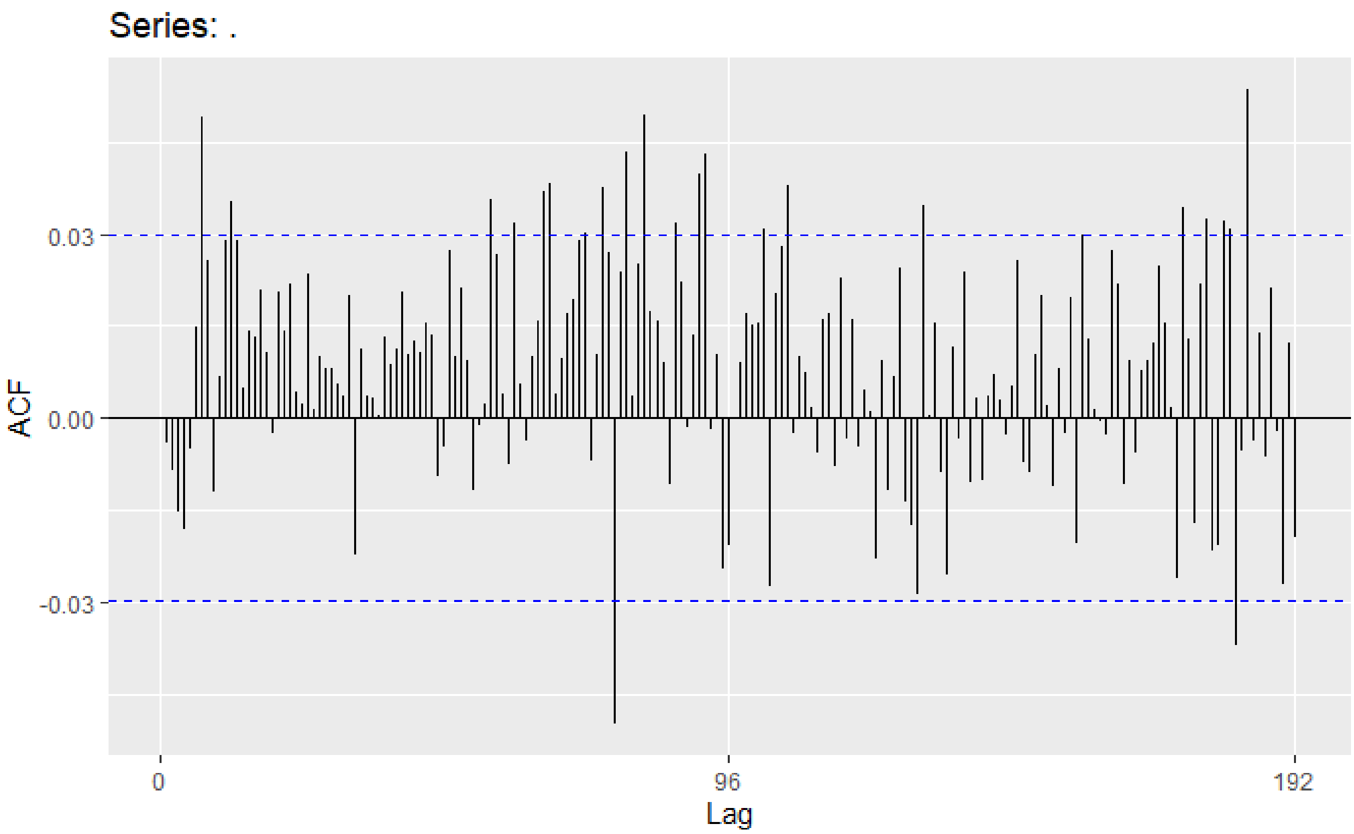
```
checkresiduals(fit303)
```

```


	Ljung-Box test

data:  Residuals from ARIMA(3,0,3)(0,1,1)[96]
Q* = 318.64, df = 185, p-value = 3.682e-09

Model df: 7.   Total lags used: 192
```



Hide

```
fit303 %>% residuals() %>% ggAcf()
```



On voit un pic sur le lag 7, on va encore améliorer notre modèle :

Hide

```
fit307=Arima(elec_train, order=c(3,0,7), seasonal=c(0,1,1))#96
summary(fit307)
```

```
Series: elec_train
ARIMA(3,0,7)(0,1,1)[96]

Coefficients:
         ar1     ar2     ar3     ma1     ma2      ma3      ma4      ma5      ma6      ma7
sma1
      0.0680  0.1428  0.6068  0.6467  0.4033  -0.1743  -0.2658  -0.2454  -0.2176  -0.0490  -
0.8628
s.e.  0.7076  0.5576  0.6702  0.7075  0.6475   0.1823   0.0481   0.1560   0.1898   0.0353
0.0087

sigma^2 = 61.19:  log likelihood = -14725.52
AIC=29475.03   AICc=29475.11   BIC=29551.2

Training set error measures:
                     ME     RMSE      MAE       MPE     MAPE      MASE          ACF1
Training set -0.2898195 7.724954 4.695657 -0.218786 2.144737 0.5754388 -0.001255298
```

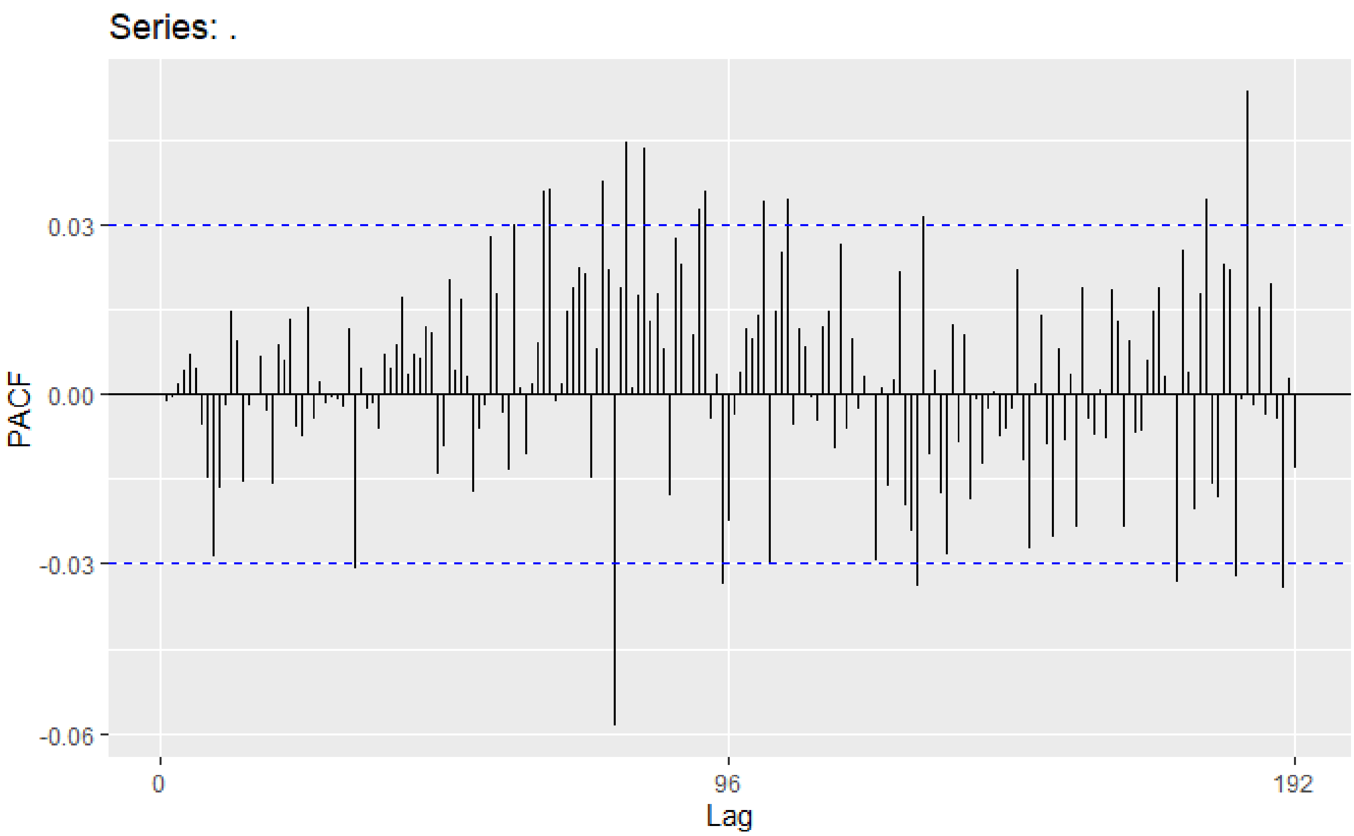
Hide

```
prevfit307 = forecast(fit307)
cat("RMSE", sqrt(mean(elec_test-prevfit307$mean)^2))
```

```
RMSE 5.081641
```

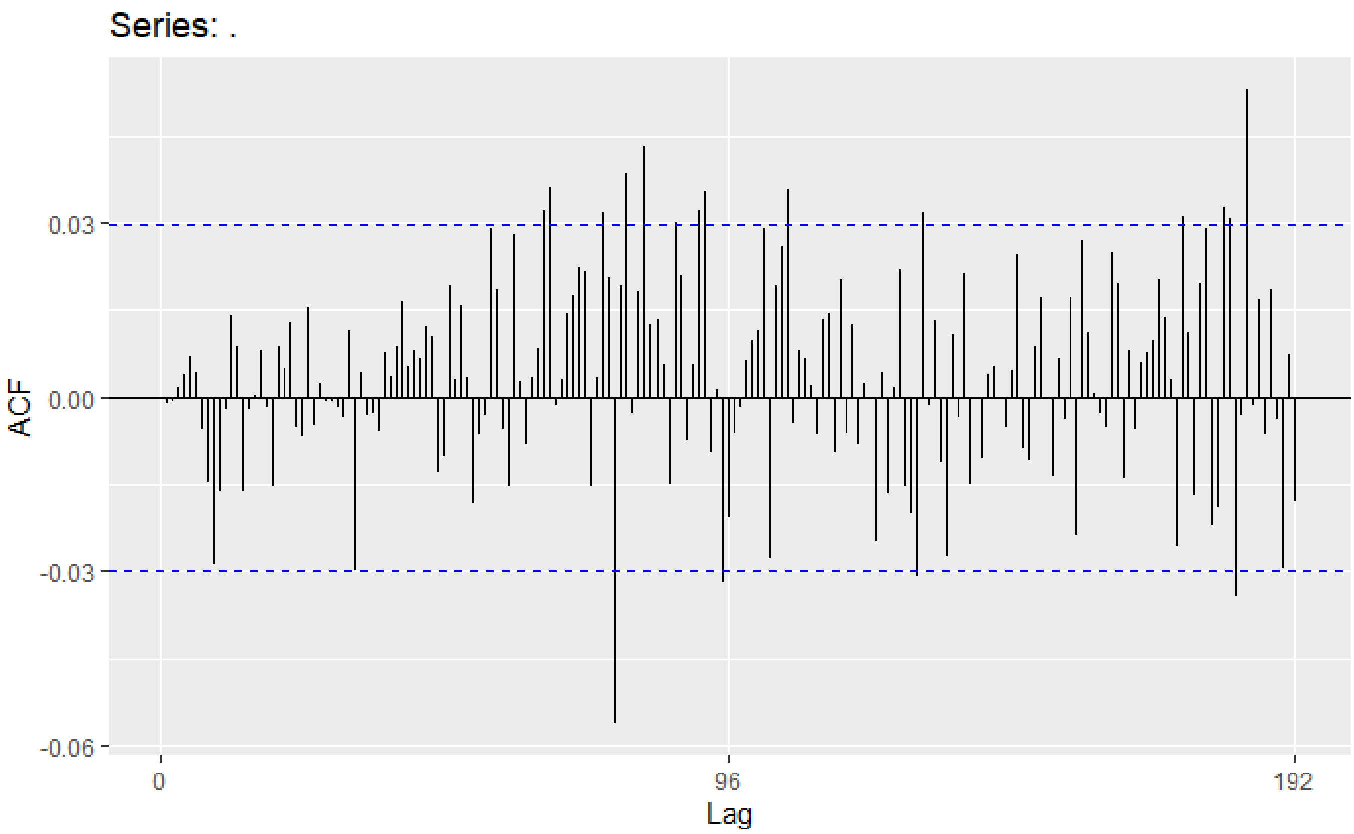
Hide

```
fit307 %>% residuals() %>% ggPacf()
```



Hide

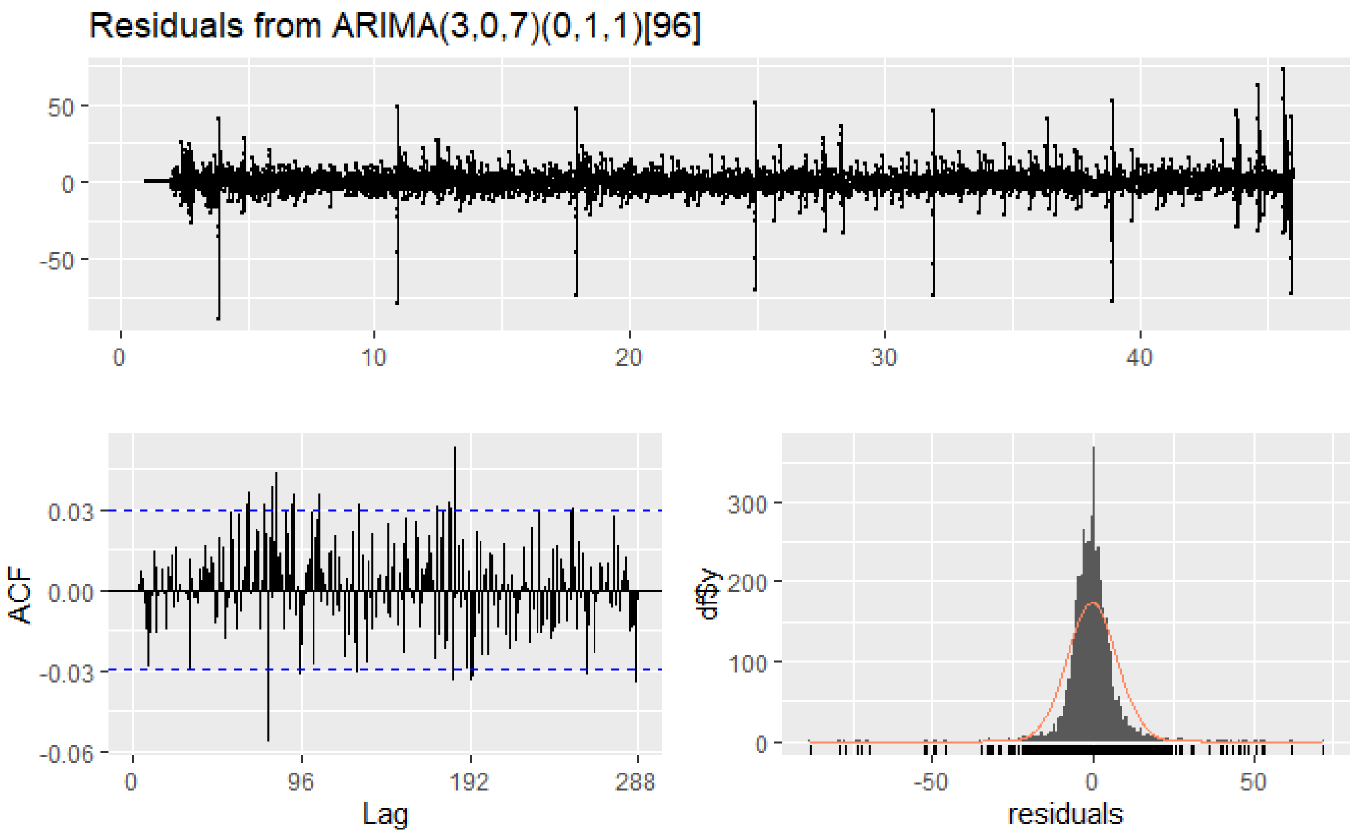
```
fit307 %>% residuals() %>% ggAcf()
```



Hide

```
checkresiduals(fit307)
```

```
	Ljung-Box test

data:  Residuals from ARIMA(3,0,7)(0,1,1)[96]
Q* = 248.16, df = 181, p-value = 0.0006805

Model df: 11.   Total lags used: 192
```



Il reste des corrections pour capturer toutes les informations elles nécessitent une trop grande puissance de calcul pour que l'on puisse continuer.
Cependant avec une p-value 0.0006805, les bruits blanc ne sont pas pris en compte.

SARIMA Auto

On va essayer de voir ce qu'on peut obtenir avec un SARIMA automatique :

Hide

```
fitAuto=auto.arima(elec_train,lambda = "auto")
summary(fitAuto)
```

```
Series: elec_train
ARIMA(5,0,1)(0,1,0)[96]
Box Cox transformation: lambda= 0.443743

Coefficients:
         ar1      ar2     ar3      ar4     ar5      ma1
      0.9335  -0.0531  0.0256  -0.2543  0.1586  -0.1900
s.e.  0.0996   0.0747  0.0223   0.0208  0.0195   0.1007

sigma^2 = 0.2459:  log likelihood = -3024.7
AIC=6063.4   AICc=6063.43   BIC=6107.83

Training set error measures:
                     ME     RMSE      MAE        MPE     MAPE      MASE         ACF1
Training set -0.0871035 10.08052 5.942022 -0.1357493 2.701743 0.7281771 0.0007368769
```

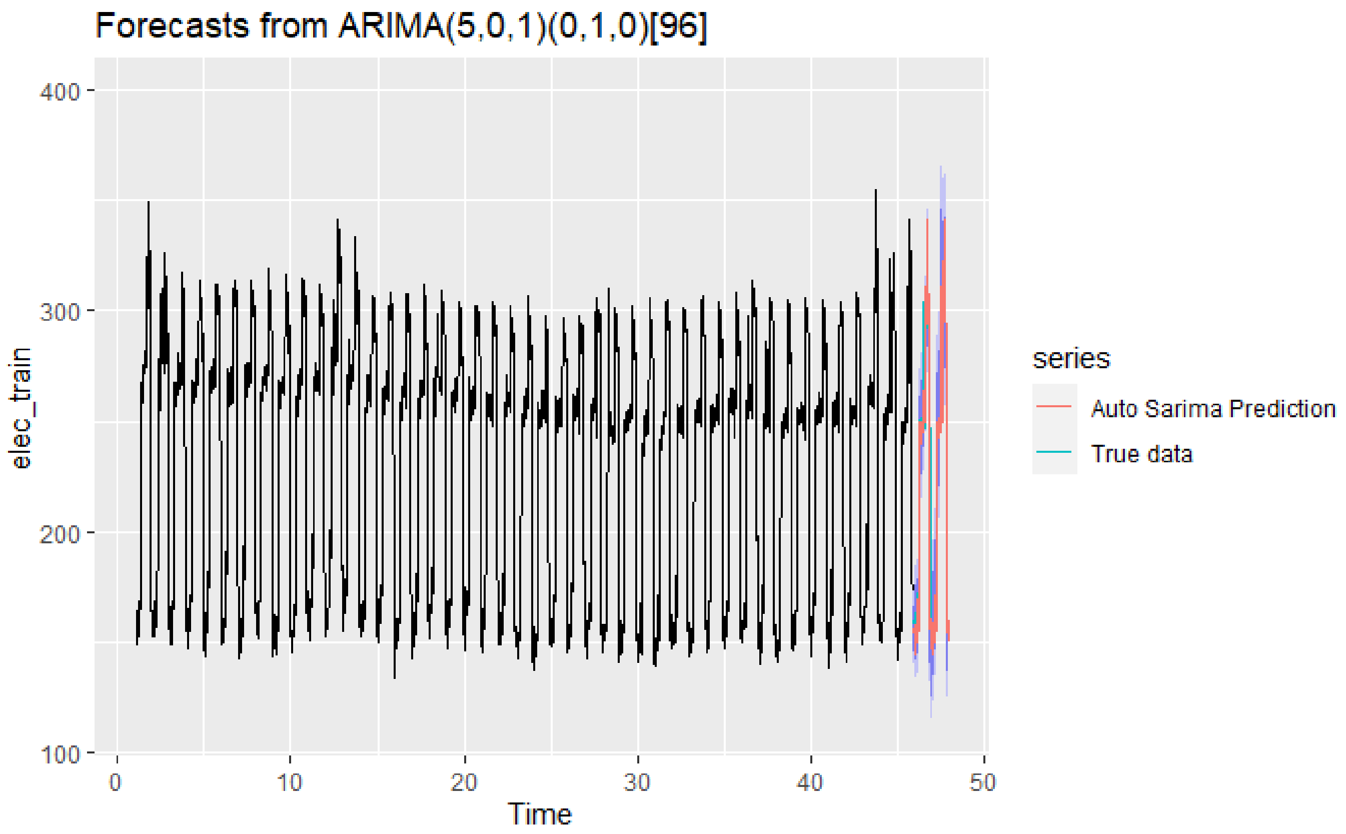
Hide

```
prev= forecast(fitAuto)
cat("RMSE", sqrt(mean(elec_test-prev$mean)^2))
```

```
RMSE 9.129775
```

Hide

```
autoplot(prev) + autolayer(elec_test, series="True data")+ autolayer(prev$mean, series="Auto
Sarima Prediction")
```



Les résulats de ce modèle sont inférieur à ce que l'on a trouvé auparavant

NEURAL NETWORK

On essaye de faire une prédiction avec un réseau de neurones

Hide

```
NNfit=nnetar(elec_train)
summary(NNfit)
```

```
          Length Class        Mode
x         4315   ts           numeric
m            1   -none-       numeric
p            1   -none-       numeric
P            1   -none-       numeric
scalex       2   -none-       list
size         1   -none-       numeric
subset    4315   -none-       numeric
model       20   nnetarmodels list
nnetargs     0   -none-       list
fitted    4315   ts           numeric
residuals 4315   ts           numeric
lags        12   -none-       numeric
series       1   -none-       character
method       1   -none-       character
call         2   -none-       call
```

Hide

```
prevNNfit= forecast(NNfit)
cat("RMSE", sqrt(mean(elec_test-prevNNfit$mean)^2))
```

```
RMSE 8.360124
```

Le résultat n’est pas assez performant, comparé aux RMSE du SARIMA

Holt-Winters

On essaye de faire une prédiction avec un Holt-Winters.

Hide

```
fitHW= HoltWinters(elec_train)
summary(fitHW)
```

```
             Length Class  Mode
fitted       16876  mts    numeric
x             4315  ts     numeric
alpha            1  -none- numeric
beta             1  -none- numeric
gamma            1  -none- numeric
coefficients    98  -none- numeric
seasonal         1  -none- character
SSE              1  -none- numeric
call             2  -none- call
```

Hide

```
prevfitHW= forecast(fitHW)
cat("RMSE", sqrt(mean(elec_test-prevfitHW$mean)^2))
```

```
RMSE 9.046928
```

De même pour cette méthode, le résultat n’est pas assez performant

Choix Modèle

Même si le modèle SARIMA(3,0,3)(0,1,1)[96] n’est pas le modèle qui a capté le plus d’informations.
C’est le modèle avec le RMSE le plus bas : 3.982057
Son bruit balnc n’est pas pris en compte avec une p-value = 3.682e-09

Export des prédictions

Pour la prévision sans l’utilisation de la température extérieure

Hide

```
fit303F=Arima(elec_prev, order=c(3,0,3), seasonal=c(0,1,1))#96
prevFinal = forecast(fit303F, h=96)
```

Hide

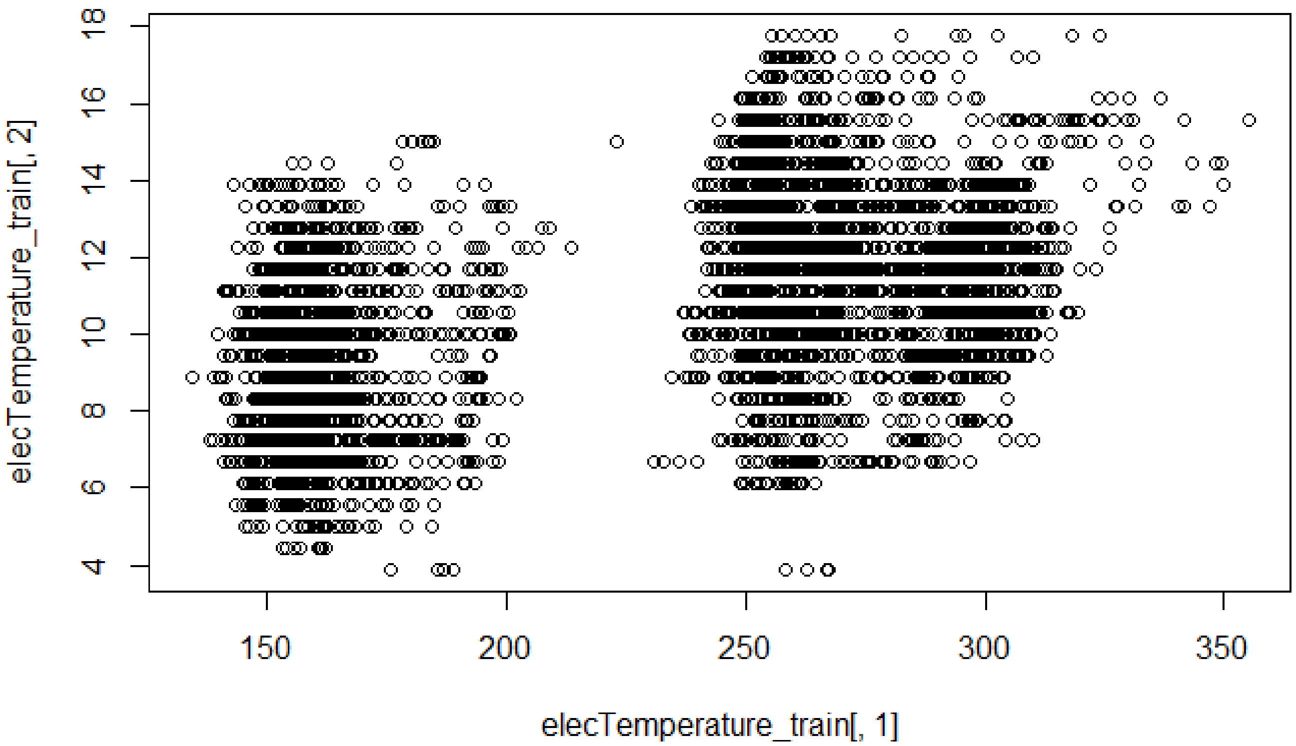
```
write.xlsx(as.data.frame(prevFinal$mean), "exportPower.xlsx")
```

Utilisation de la variable Temperature

On va d’abord essayer de voir si il y a correlation entre les variables Temperature et Power

Hide

```
plot(elecTemperature_train[,1], y=elecTemperature_train[,2])
```



Hide

```
cor(elecTemperature_train)
```

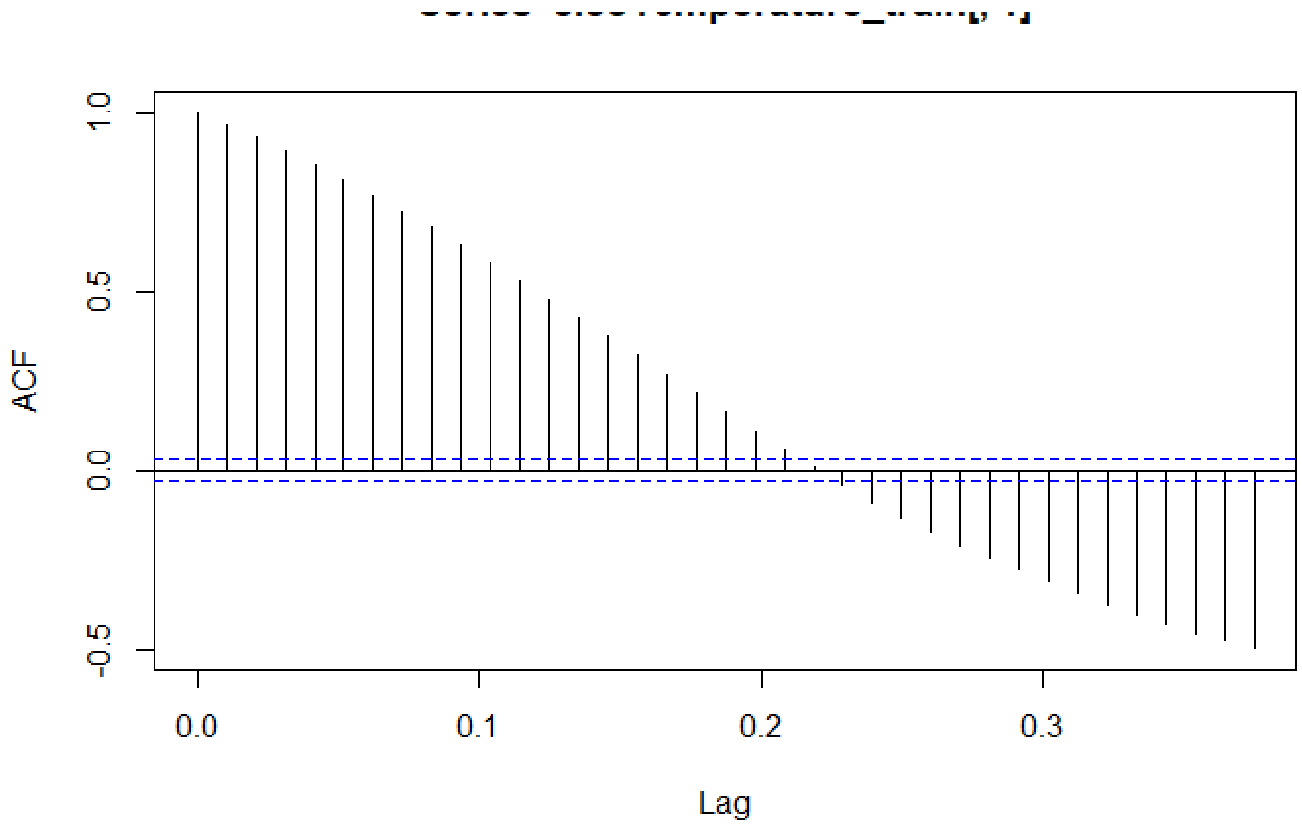
```
            Power (kW) Temp (C°)
Power (kW)  1.0000000 0.4722473
Temp (C°)   0.4722473 1.0000000
```

Nous obtenons une valeur de 0.472
Les variables ne semblent pas vraiment corrélées.

Prévisions

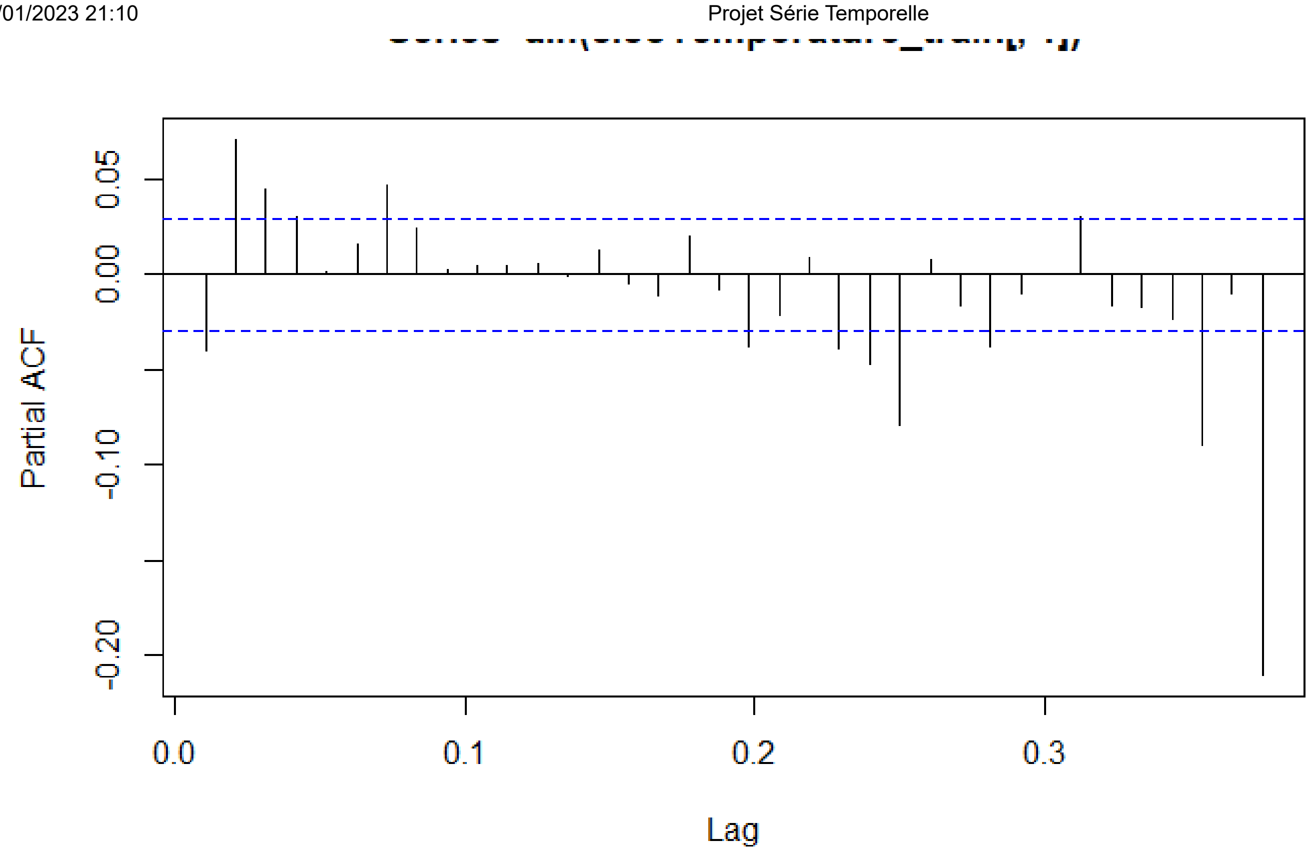
Hide

```
tmp=acf(elecTemperature_train[,1],type="cor",plot = FALSE)
plot(tmp)
```



Hide

```
diffpacf=pacf(diff(elecTemperature_train[,1]),type="cor",plot = FALSE)
plot(diffpacf)
```



On remarque à nouveau une saisonnalité et avec le PACF on a un pic du lag à 2

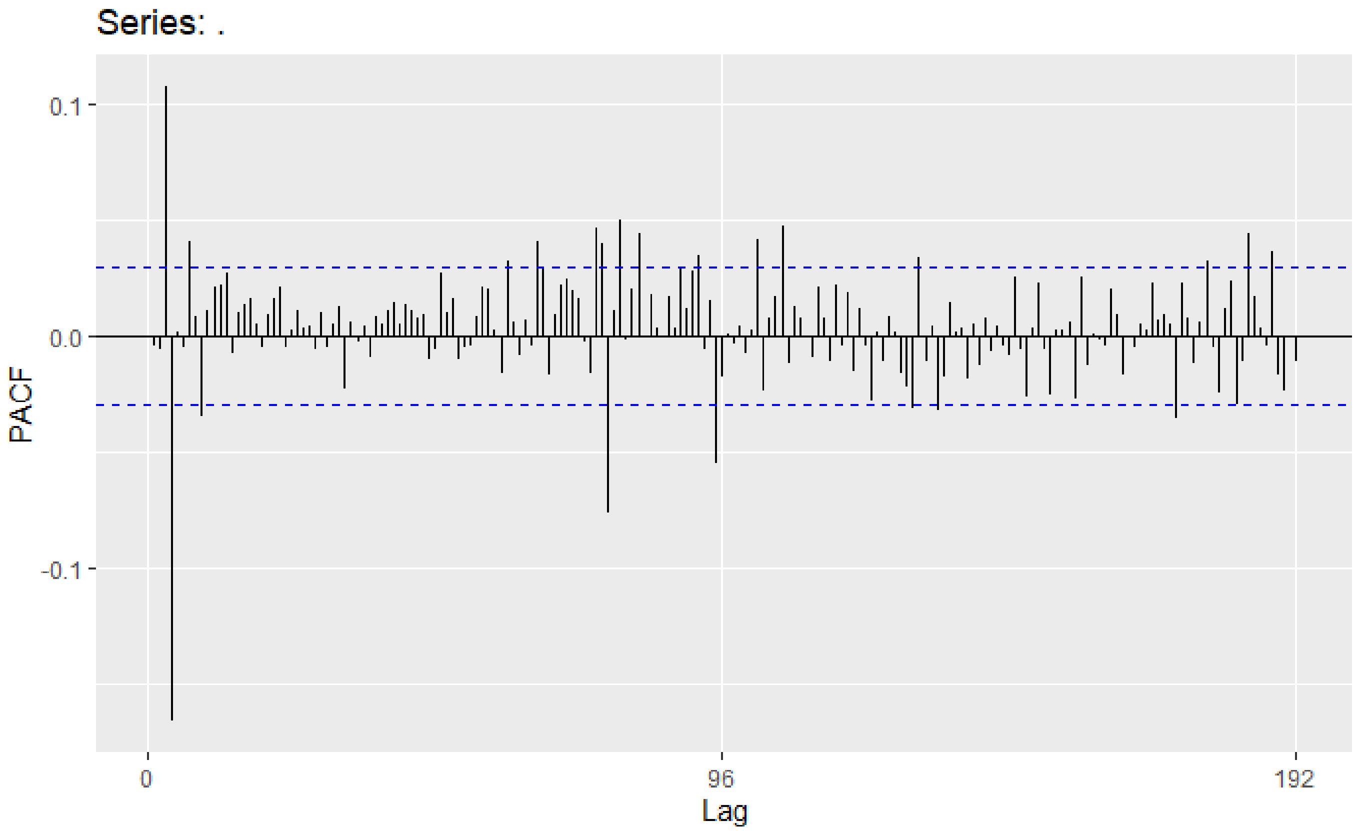
Hide

```
cat("RMSE", sqrt(mean(elecTemperature_test[,1]-prevfit200Temp$mean)^2))
```

```
RMSE 3.261178
```

Hide

```
fit200Temp %>% residuals() %>% ggPacf()
```



Grâce au PACF, on voit un pic au lag 3, on va améliorer le modèle

Hide

```
fit400Temp=Arima(elecTemperature_train[,1], order=c(4,0,0), seasonal=c(0,1,1), xreg = elecTem
perature_train[,2])#96
summary(fit400Temp)
```

```
Series: elecTemperature_train[, 1]
Regression with ARIMA(4,0,0)(0,1,1)[96] errors

Coefficients:
         ar1     ar2     ar3      ar4     sma1    xreg
      0.6792  0.0959  0.1244  -0.1493  -0.8631  0.7070
s.e.  0.0151  0.0182  0.0182   0.0152   0.0084  0.2005

sigma^2 = 64.55:  log likelihood = -15176.92
AIC=30367.84   AICc=30367.87   BIC=30412.43

Training set error measures:
                     ME    RMSE      MAE        MPE     MAPE    MASE       ACF1
Training set -0.4237798 7.94078 4.791944 -0.2784618 2.180156 0.57578 0.01656161
```

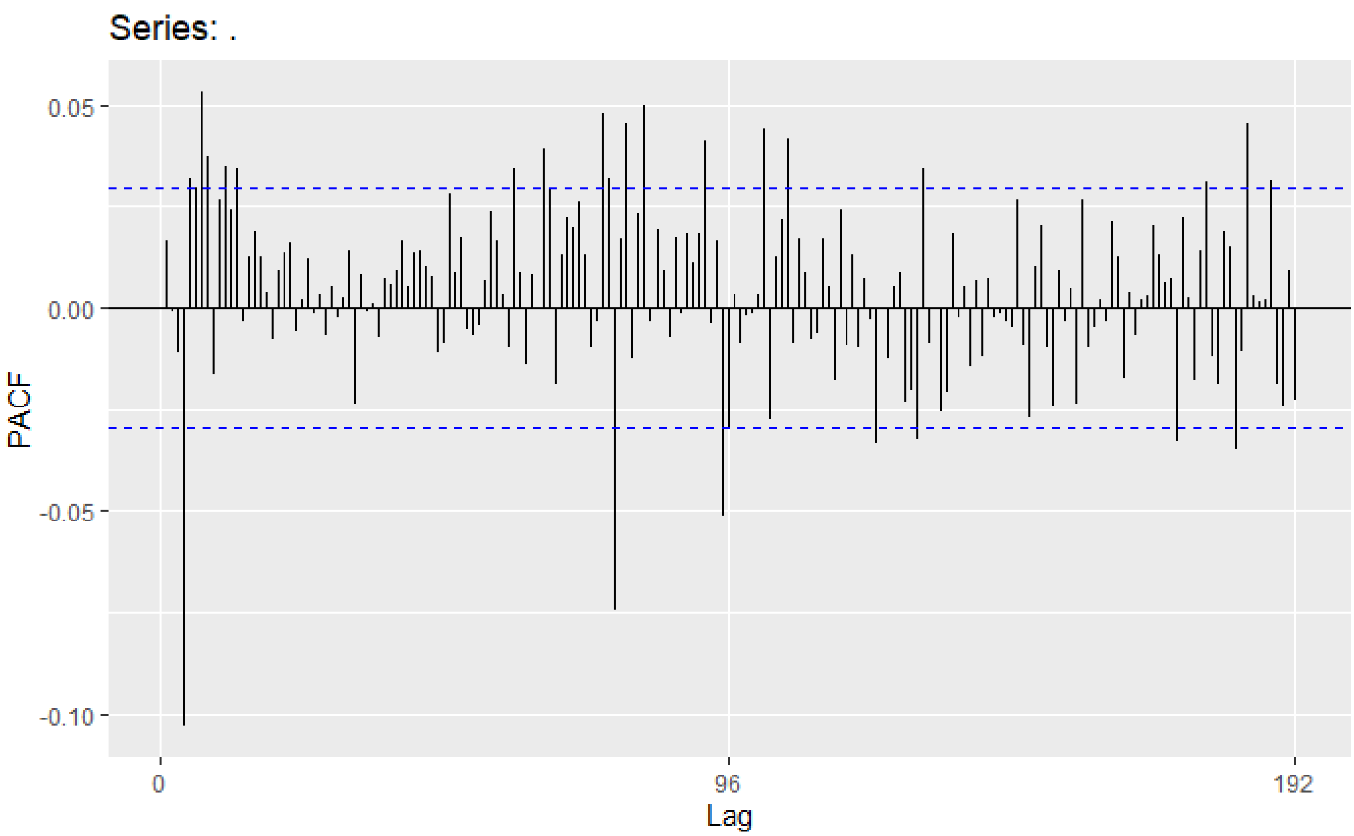
Hide

```
prevfit400Temp= forecast(fit400Temp, xreg=elecTemperature_test[,2])
cat("RMSE", sqrt(mean(elecTemperature_test[,1]-prevfit400Temp$mean)^2))
```

```
RMSE 3.380815
```

Hide

```
fit400Temp %>% residuals() %>% ggPacf()
```



On observe toujours un pic sur le lag 4 on va essayer d'en prendre un autre pic ici avec le lag 7

Hide

```
fit700Temp=Arima(elecTemperature_train[,1], order=c(7,0,0), seasonal=c(0,1,1), xreg = elecTem
perature_train[,2])#96
summary(fit700Temp)
```

```
Series: elecTemperature_train[, 1]
Regression with ARIMA(7,0,0)(0,1,1)[96] errors

Coefficients:
         ar1     ar2     ar3      ar4     ar5     ar6     ar7     sma1    xreg
      0.7136  0.0821  0.0913  -0.2222  0.0803  0.0347  0.0243  -0.8614  0.5014
s.e.  0.0154  0.0189  0.0189   0.0187  0.0191  0.0191  0.0157   0.0087  0.2253

sigma^2 = 61.29:  log likelihood = -14729.48
AIC=29478.96   AICc=29479.01   BIC=29542.43

Training set error measures:
                     ME     RMSE      MAE        MPE     MAPE      MASE         ACF1
Training set -0.3477228 7.733109 4.693706 -0.2438774 2.143879 0.5753418 -0.001330753
```

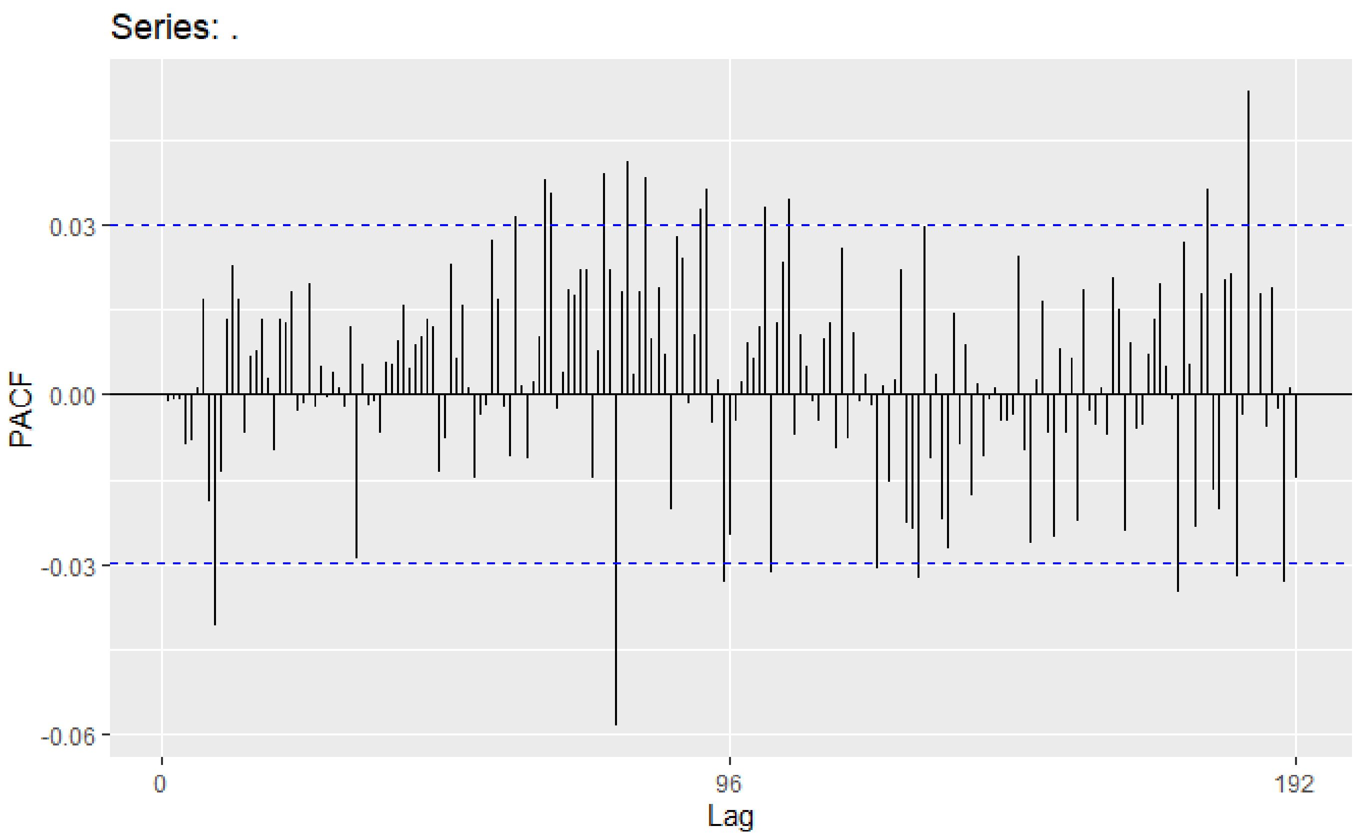
Hide

```
prevfit700Temp= forecast(fit700Temp, xreg=elecTemperature_test[,2])
cat("RMSE", sqrt(mean(elecTemperature_test[,1]-prevfit700Temp$mean)^2))
```

```
RMSE 4.657614
```

Hide

```
fit700Temp %>% residuals() %>% ggPacf()
```



On observe un pic sur le lag 9 on va essayer d'améliorer le modèle

Hide

```
fit900Temp=Arima(elecTemperature_train[,1], order=c(9,0,0), seasonal=c(0,1,1), xreg = elecTem
perature_train[,2])#96

summary(fit900Temp)
```

```
Series: elecTemperature_train[, 1]
Regression with ARIMA(9,0,0)(0,1,1)[96] errors

Coefficients:
         ar1     ar2     ar3      ar4    ar5     ar6     ar7      ar8      ar9     sma1    xr
eg
      0.6944  0.0838  0.1185  -0.2455  0.105  0.0108  0.0452  -0.0136  -0.0066  -0.8659  0.60
58
s.e.  0.0152  0.0186  0.0186   0.0187  0.019  0.0186  0.0186   0.0185   0.0153   0.0085  0.21
99

sigma^2 = 63.38:  log likelihood = -15136
AIC=30296.01   AICc=30296.08   BIC=30372.45

Training set error measures:
                     ME     RMSE      MAE        MPE     MAPE      MASE         ACF1
Training set -0.3546927 7.864058 4.771477 -0.2482636 2.169055 0.5733208 -0.001483967
```

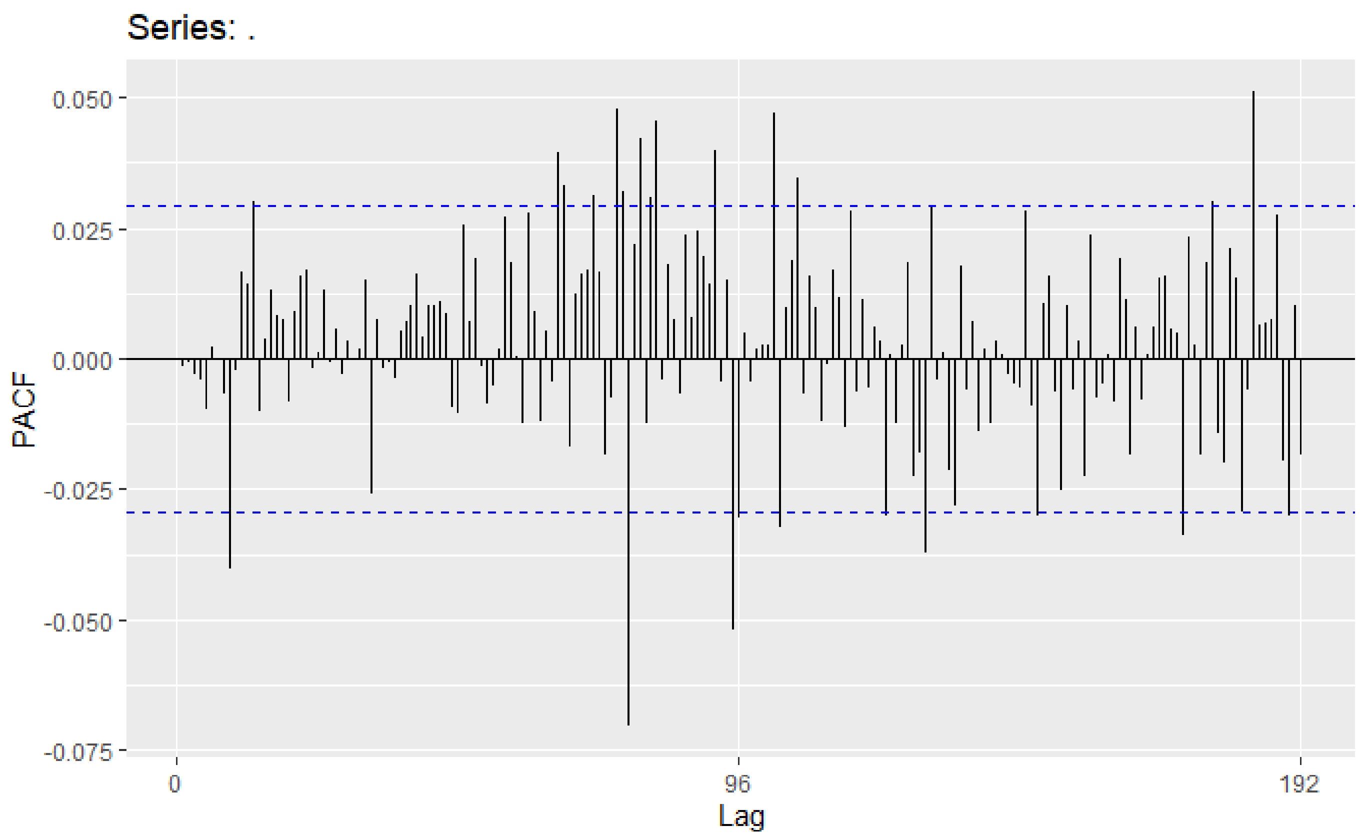
Hide

```
prevfit900Temp= forecast(fit900Temp, xreg=elecTemperature_test[,2])
cat("RMSE", sqrt(mean(elecTemperature_test[,1]-prevfit900Temp$mean)^2))
```

```
RMSE 3.143429
```

Hide

```
fit900Temp %>% residuals() %>% ggPacf()
```



On observe un pic sur le lag 8 on va essayer d’améliorer le modèle

Hide

```
fit800Temp=Arima(elecTemperature_train[,1], order=c(8,0,0), seasonal=c(0,1,1), xreg = elecTem
perature_train[,2])#96
summary(fit800Temp)
```

```
Series: elecTemperature_train[, 1]
Regression with ARIMA(8,0,0)(0,1,1)[96] errors

Coefficients:
         ar1     ar2     ar3      ar4     ar5     ar6     ar7      ar8     sma1    xreg
      0.6946  0.0835  0.1184  -0.2461  0.1066  0.0100  0.0447  -0.0181  -0.8660  0.5986
s.e.  0.0152  0.0185  0.0186   0.0186  0.0186  0.0186  0.0185   0.0153   0.0085  0.2200

sigma^2 = 63.37:  log likelihood = -15136.1
AIC=30294.2   AICc=30294.26   BIC=30364.26

Training set error measures:
                     ME     RMSE      MAE        MPE     MAPE      MASE        ACF1
Training set -0.3522732 7.864132 4.770965 -0.2472987 2.168549 0.5732593 -0.00192004
```

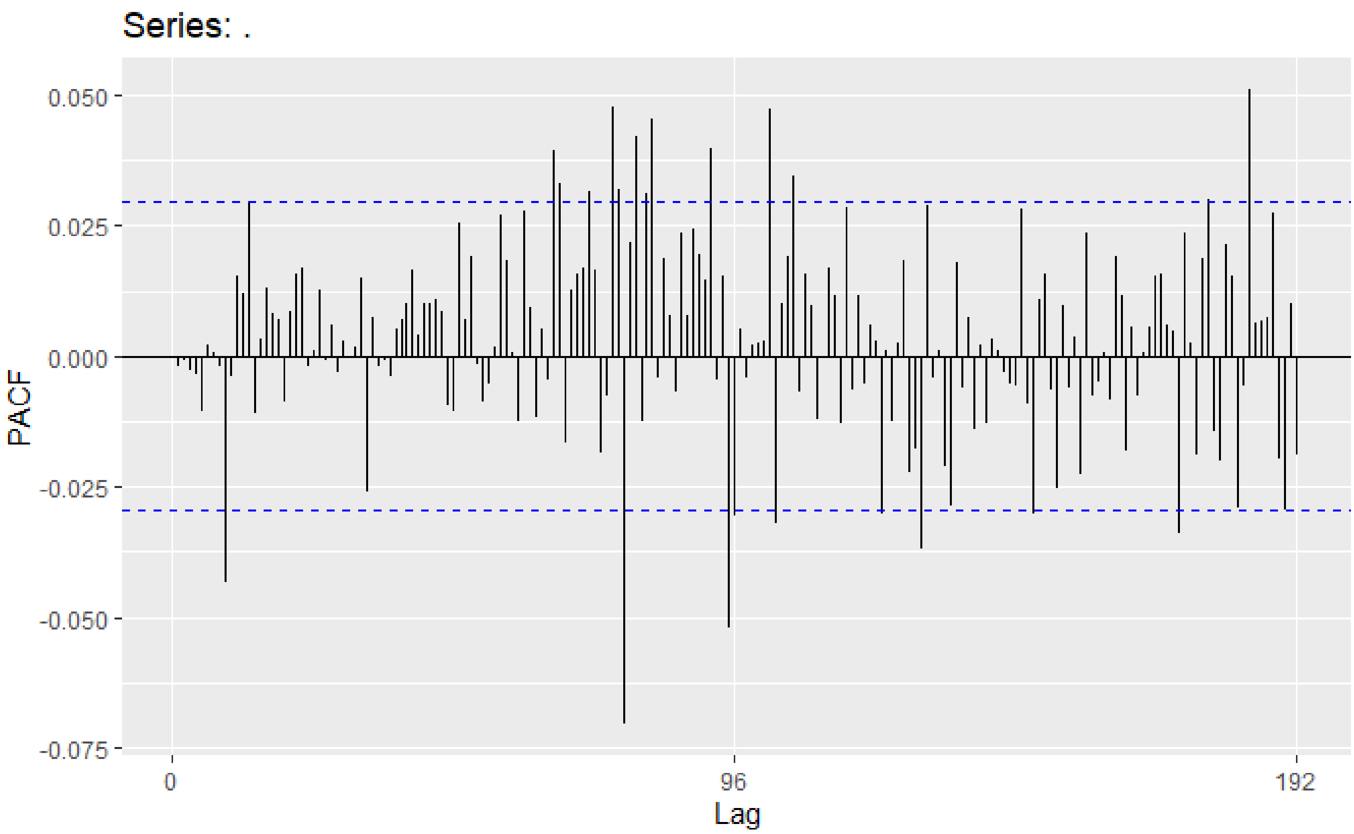
Hide

```
prevfit800Temp= forecast(fit800Temp, xreg=elecTemperature_test[,2])
cat("RMSE", sqrt(mean(elecTemperature_test[,1]-prevfit800Temp$mean)^2))
```

```
RMSE 3.137962
```

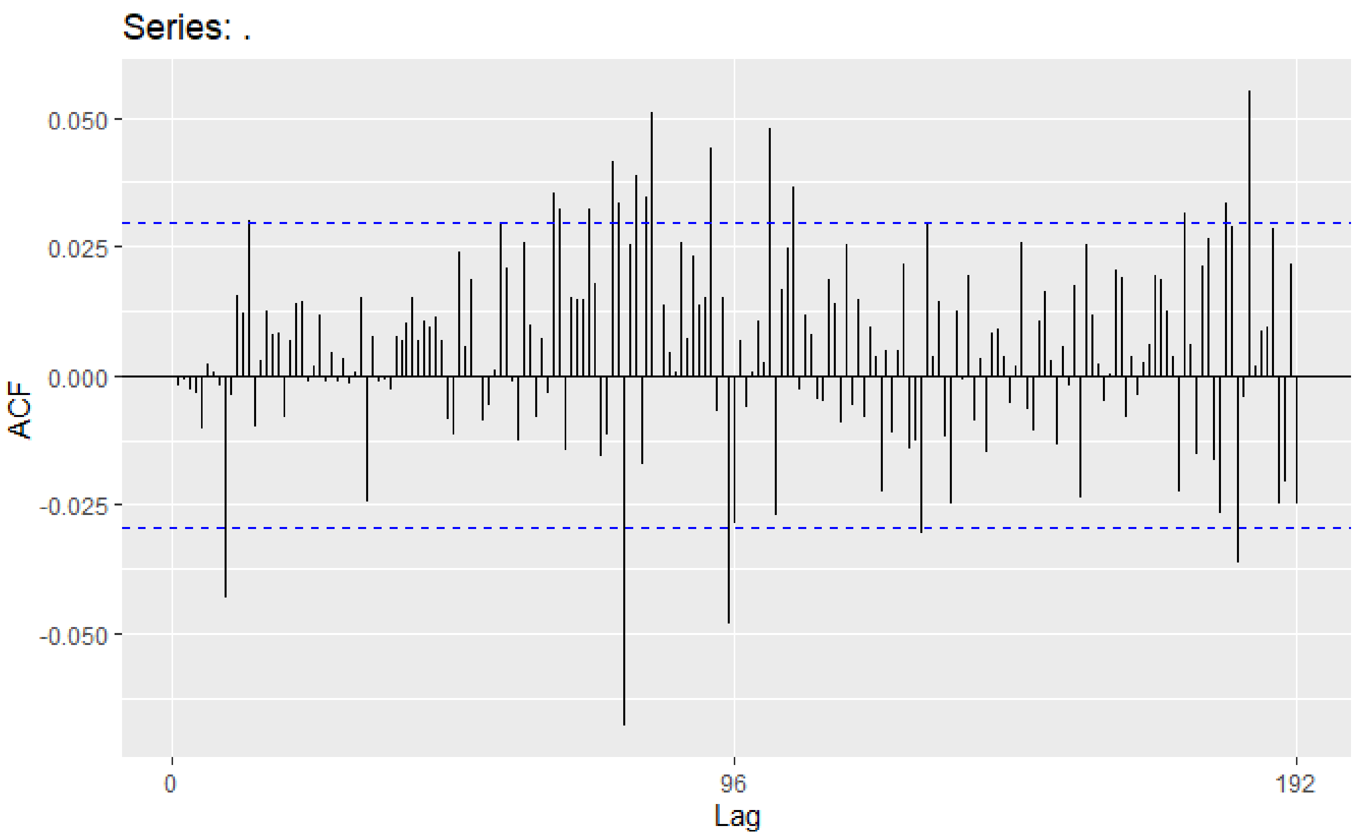
Hide

```
fit800Temp %>% residuals() %>% ggPacf()
```



Hide

```
fit800Temp %>% residuals() %>% ggAcf()
```



On arrive plus à stabiliser les pic des lag, cela nécessite trop de performances.

NEURAL NETWORK

On essaye de faire une prédiction avec un réseau de neurones.

Hide

```
NNfitTemp=nnetar(elecTemperature_train[,1], xreg = elecTemperature_train[,2])
summary(NNfitTemp)
```

```
          Length Class        Mode
x         4411   ts           numeric
m            1   -none-       numeric
p            1   -none-       numeric
P            1   -none-       numeric
scalex       2   -none-       list
scalexreg    2   -none-       list
size         1   -none-       numeric
xreg      4411   -none-       numeric
subset    4411   -none-       numeric
model       20   nnetarmodels list
nnetargs     0   -none-       list
fitted    4411   ts           numeric
residuals 4411   ts           numeric
lags        17   -none-       numeric
series       1   -none-       character
method       1   -none-       character
call         3   -none-       call
```

Hide

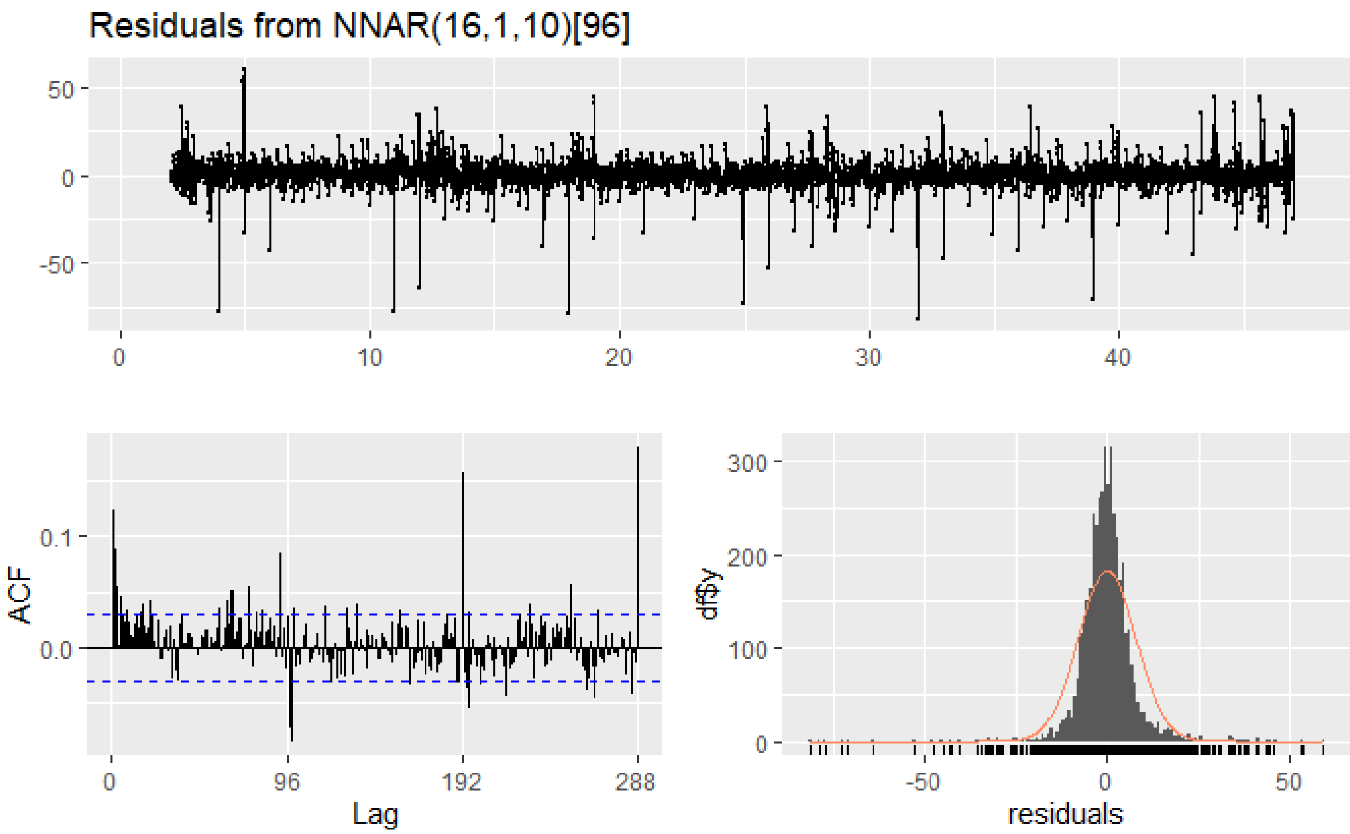
```
prevNNfitTemp= forecast(NNfitTemp, xreg=elecTemperature_test[,2])
cat("RMSE", sqrt(mean(elecTemperature_test[,1]-prevNNfitTemp$mean)^2))
```

```
RMSE 1.208937
```

Hide

```
checkresiduals(NNfitTemp)
```

```
Warning in modeldf.default(object) :
  Could not find appropriate degrees of freedom for this model.
```



Le résultat est très satisfesant comparé aux RMSE du SARIMA

Pour la prévision avec l’utilisation de la température extérieure

Les prédictions semblent vraiment similaires avec ou sans la variable température, on a aperçu une petite différence de performances entre les modèles.
Le modèle utilisant un réseau de neurone s’est démarqué avec un RMSE bizarrement bas de 1.208937.

Exportation des résultats

Hide

```
fitFinaltemp = nnetar(elecTemperature_prev[1:4507,1], xreg = elecTemperature_prev[1:4507,2])
prevFinalTemp= forecast(fitFinaltemp, xreg=elecTemperature_prev[4508:4603,2])
write.xlsx(as.data.frame(prevFinalTemp$mean), "exportPowerTemp.xlsx")
```

A noter que dans le fichier prédiction :
La première colonne contient les prédictions SANS la variable température.
La deuxième colonne contient les prédictions AVEC la variable température.