Séries Temporelles 1 (2A) - ENSAI

TP 3 : Mise en œuvre des outils du cours (Une Proposition de Solution)

José Gregorio GOMEZ-GARCIA

1er décembre 2023

Exercice 1 : Prévision de la qualité de l'air

Le but de cet exercice est d'étudier la qualité de l'air dans les stations de RER parisiennes. En particulier, on va se concentrer sur la station Auber. Les données sont disponibles sur moodle sous le nom `qualite-de-lair-auber.csv`. Vous pouvez aussi télécharger les données actualisées sur le site https://air-interieur.ratp.fr/details/AUBEA .

Le tableau de données contient plusieurs variables comme la date/heure (aaaa-mm-jj-hh), le volume de monoxide d'azote (CO), le volume de dyoxide d'azote (CO2), dyoxide de carbone (CO2), etc. Pour les deux premières parties, nous nous concentrerons sur les mesures de volume de dioxyde d'azote (NO2) jusqu'au **31 décembre 2017** car entre les 01/01/2017 et 23/12/2021 il n'y a pas de registres.

- Chargement des données :

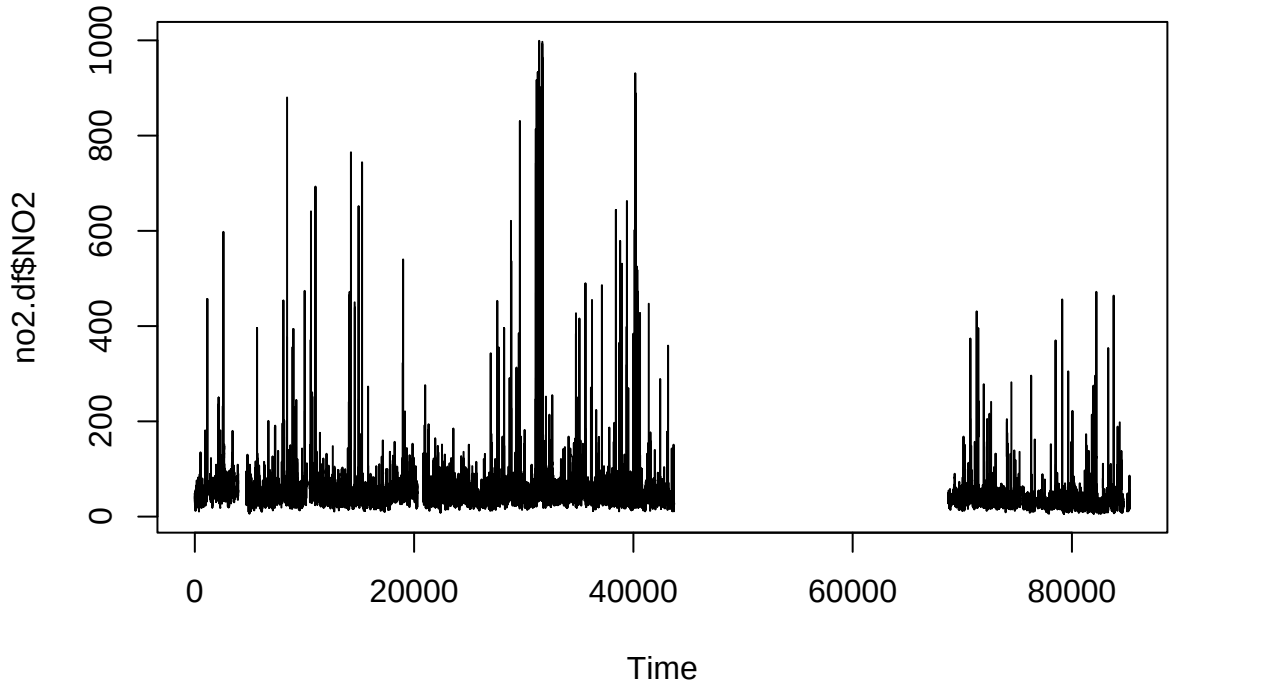
```
auber <- read.csv("qualite-de-lair-auber.csv", header = TRUE, sep = ";", dec = ",")
auber <- auber[1:85293,] #Pour supprimer les lignes vides et avec ND.
auber <- auber[nrow(auber):1,] #Rangement des données en temps progréssif (du passé au futur)

# Transformation de la date en format aaaa-mm-jj hh :
library(lubridate)
dates <- data.frame(date = auber$DATE.HEURE)
dates$date <- as.POSIXct(dates$date, format = "%Y-%m-%dT%H") # Transforme la colonne
# "date" en objet de classe POSIXct

# Extraction des composantes de la date
dates$annee <- year(dates$date)
dates$mois <- month(dates$date)
dates$jour <- day(dates$date)
dates$heure <- hour(dates$date)

# Inclusion des dates de format POSIXcs dans le tableu de données
auber <- cbind(dates, auber[,-1])

# Série temporelle d'intérêt (NO2) :
no2.df <- auber[, c(1:5,7)]
no2.df$NO2 <- as.numeric(no2.df$NO2)
head(no2.df)
```

```
                  date annee mois jour heure NO2
85293 2013-01-04 17:00:00  2013    1    4    17  30
85292 2013-01-04 18:00:00  2013    1    4    18  49
85291 2013-01-04 19:00:00  2013    1    4    19  48
```

```
85290 2013-01-04 20:00:00  2013    1    4    20  41
85289 2013-01-04 21:00:00  2013    1    4    21  42
85288 2013-01-04 22:00:00  2013    1    4    22  41
```

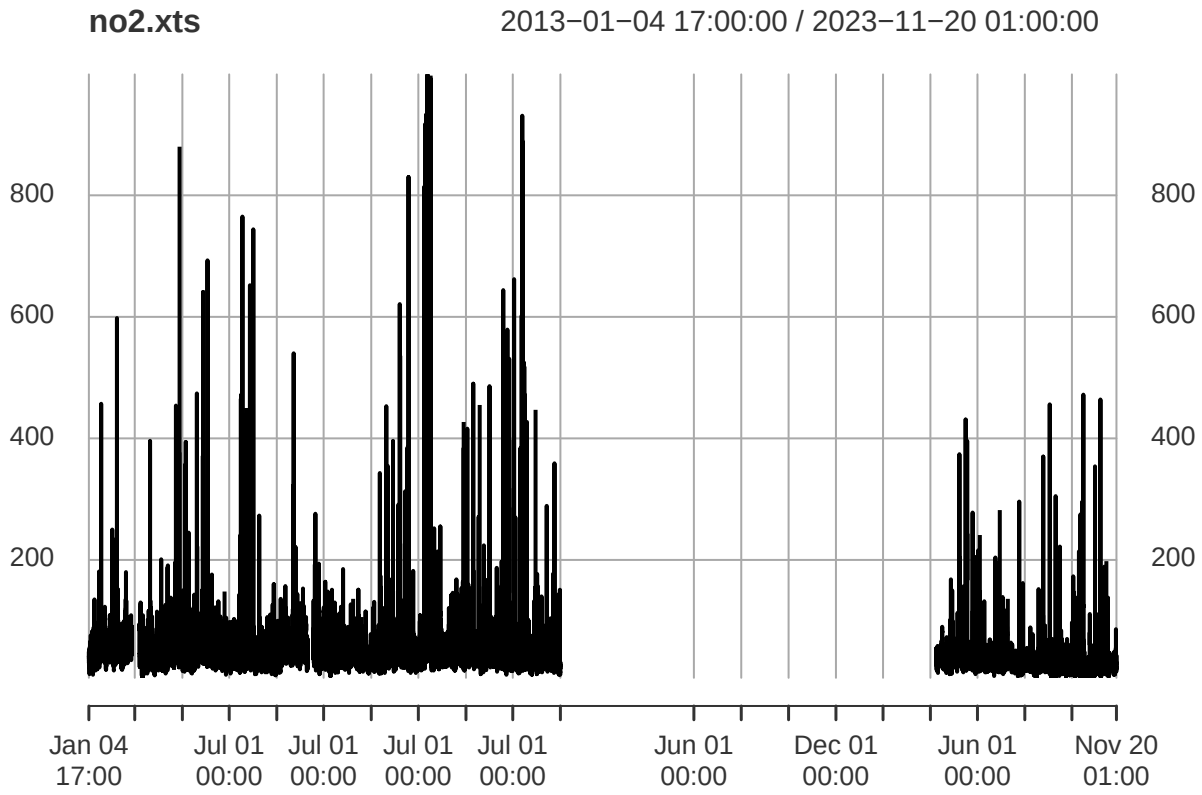
Première vue de la série temporelle :

```
plot.ts(no2.df$NO2)
```



Ou dans le format `xts` :

```
library("xts")
no2.xts <- xts(no2.df$NO2, no2.df$date)
plot(no2.xts)
```



Première partie : modélisation et prévision de la moyenne journalière de NO2

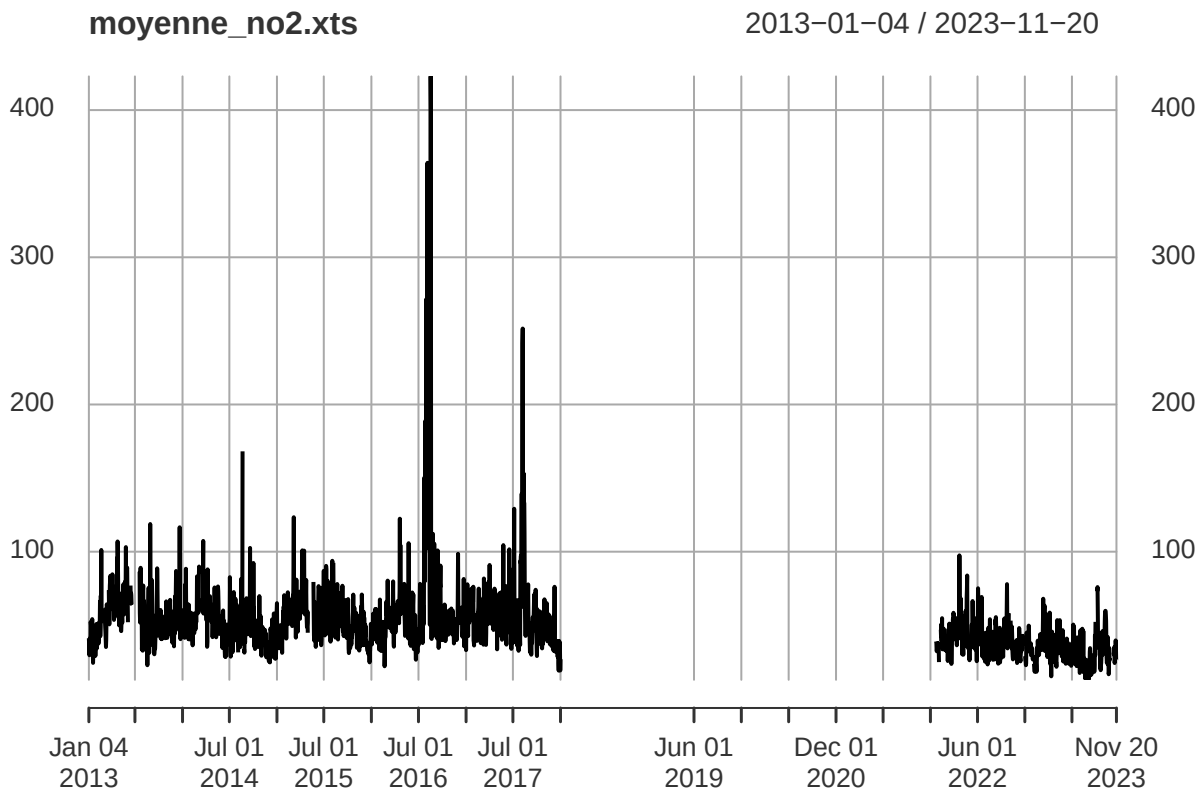
1.1.- Construire un tableau de données contenant les mesures moyennes journalières de NO2.

```
library(dplyr)
moyenne_journaliere_no2 <- no2.df %>%
  group_by(annee, mois, jour) %>%
  summarize(moyenne_no2 = mean(NO2, na.rm = TRUE))
summary(moyenne_journaliere_no2)
```

```
     annee           mois             jour        moyenne_no2
 Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   : 12.96
 1st Qu.:2015   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 37.92
 Median :2017   Median : 7.000   Median :16.00   Median : 47.42
 Mean   :2018   Mean   : 6.529   Mean   :15.71   Mean   : 50.95
 3rd Qu.:2021   3rd Qu.: 9.000   3rd Qu.:23.00   3rd Qu.: 58.38
 Max.   :2023   Max.   :12.000   Max.   :31.00   Max.   :422.90
                                                 NA's   :1185
```

Regardons la trayectoire observée de la série temporelle (log)“moyenne journalière de NO2” :

```
date <- paste(moyenne_journaliere_no2$annee,
              moyenne_journaliere_no2$mois,
              moyenne_journaliere_no2$jour,
              sep = "-")
date <- as.Date(date)
moyenne_no2.xts <- xts(moyenne_journaliere_no2$moyenne_no2, date)
plot(moyenne_no2.xts)
```

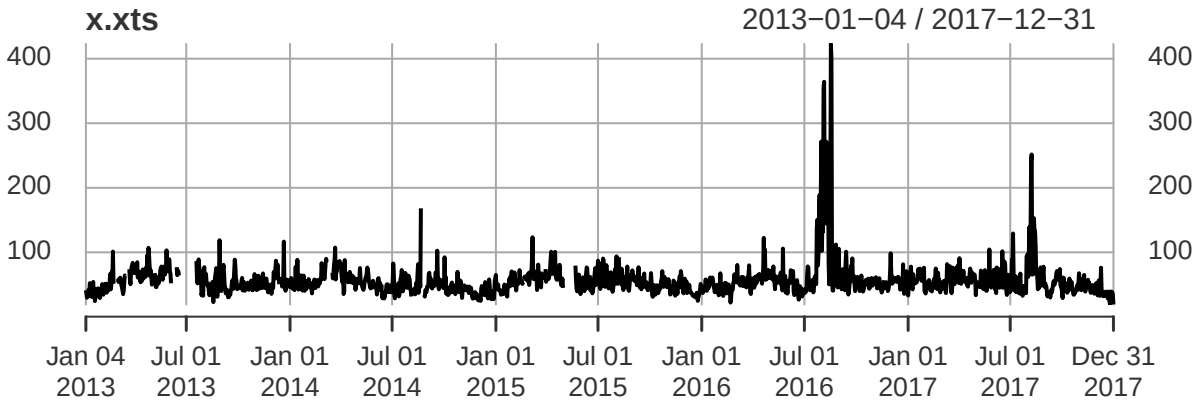


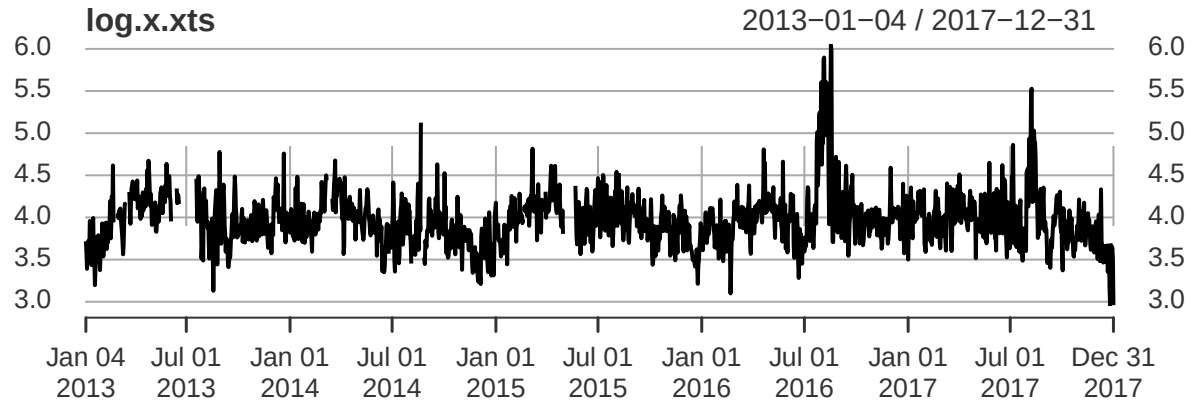
1.2.- Appliquer la méthode de Box-Jenkins pour proposer un modèle (S)ARIMA pour cette série. N'oubliez pas de choisir un échantillon de test pour vérifier la qualité de prédiction.

```
moyenne_journaliere_no2.2017 <- moyenne_journaliere_no2[1:1823,] #On se concentre sur
#les données jusqu'en 2017.
summary(moyenne_journaliere_no2.2017)
```

```
     annee           mois             jour        moyenne_no2
 Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   : 19.21
 1st Qu.:2014   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 43.35
 Median :2015   Median : 7.000   Median :16.00   Median : 51.92
 Mean   :2015   Mean   : 6.533   Mean   :15.75   Mean   : 56.27
 3rd Qu.:2016   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.: 62.04
 Max.   :2017   Max.   :12.000   Max.   :31.00   Max.   :422.90
                                                 NA's   :98
```

```
x <- moyenne_journaliere_no2.2017$moyenne_no2
log.x <- log(x) #On applique une transformation logarithmique pour plus ou moins
# régulariser les explosions (extrêmes) et s'approcher à nos hypothèses de bruit
# à variance finie.
x.xts <- xts(x, date[1:1823])
log.x.xts <- xts(log.x, date[1:1823])
par(mfrow=c(2,1))
plot(x.xts)
plot(log.x.xts)
```

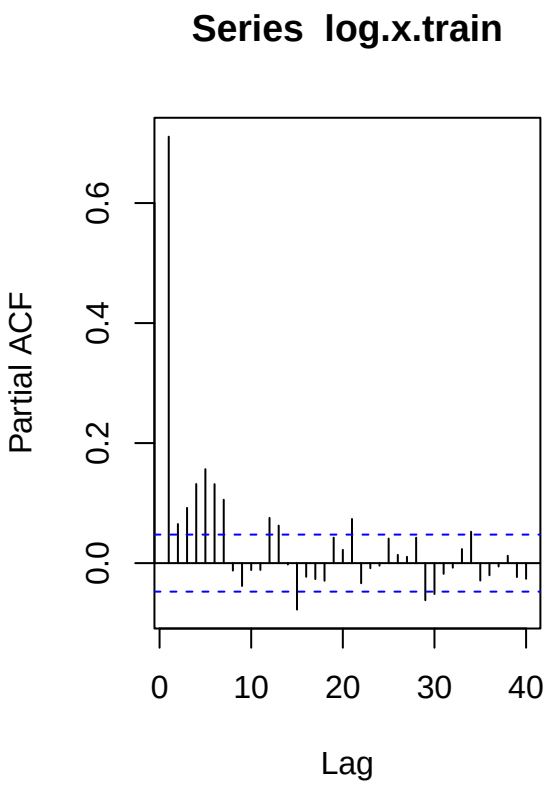


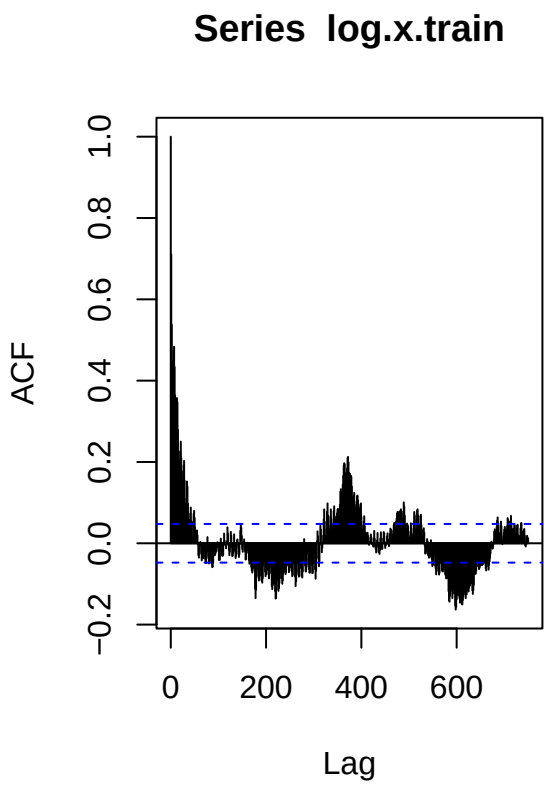



```r
x.train <- x[1:1700] #Serie d'entrainement
x.test <- x[1701:1823] #Serie de test
log.x.train <- log(x.train)
log.x.test <- log(x.test)
# Au format xts
x.xts.train <- x.xts[1:1700]
x.xts.test <- x.xts[1701:1823]
log.x.xts.train <- log(x.xts.train)
log.x.xts.test <- log(x.xts.test)
```

On réalise maintenant un diagnostic de stationnarité : acf

```r
par(mfrow=c(1,2))
pacf(log.x.train, na.action = na.pass, lag.max = 40)
acf(log.x.train, na.action = na.pass, lag.max = 750)
```

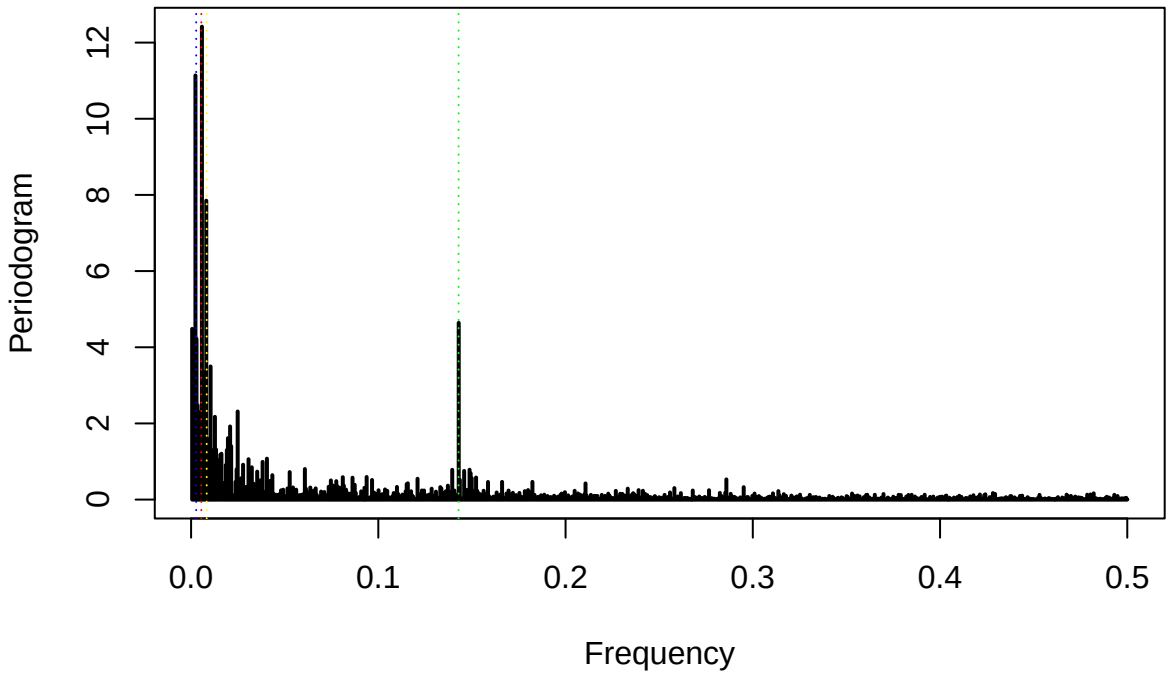





```
#abline(v=365, col="red")
#abline(v=730, col="red")
```

Selon l'acf, la série ne semble pas être stationnaire (pas de décroissance exponentielle de l'acf vers zéro). En effet, on peut constater un comportement vaguement sinusoïdal de l'acf avec une période pas bien définie. On pourrait vérifier cela avec le périodogramme *(non vu en cours)*

```
library(TSA)
x.imputed <- na.aggregate(log.x.train, FUN = mean) #Pour faire marcher le periodogram de TSA
p <- TSA::periodogram(x.imputed)
abline(v=1/7, col="green", lty=3) #Pour une période hebdomadaire
abline(v=1/120, col="yellow", lty=3) #Pour une période quadrimestriel
abline(v=1/183, col="red", lty=3) #Pour une période semestriel
abline(v=1/365, col="blue", lty=3) #Pour une période annuel
```

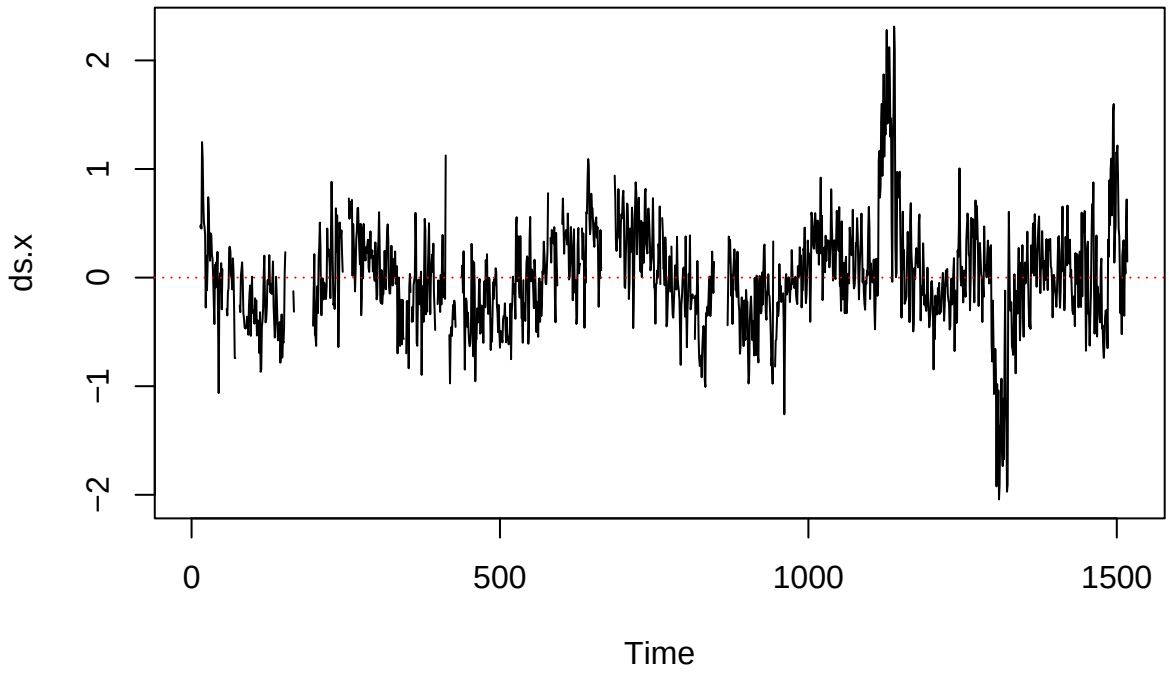



```
detach(package:TSA)
```

En fait, on observe 4 cycles remarquables : hebdomadaire, quadrimestriel, semestriel et annuel. Cependant, le plus significatif est le cycle semestriel.

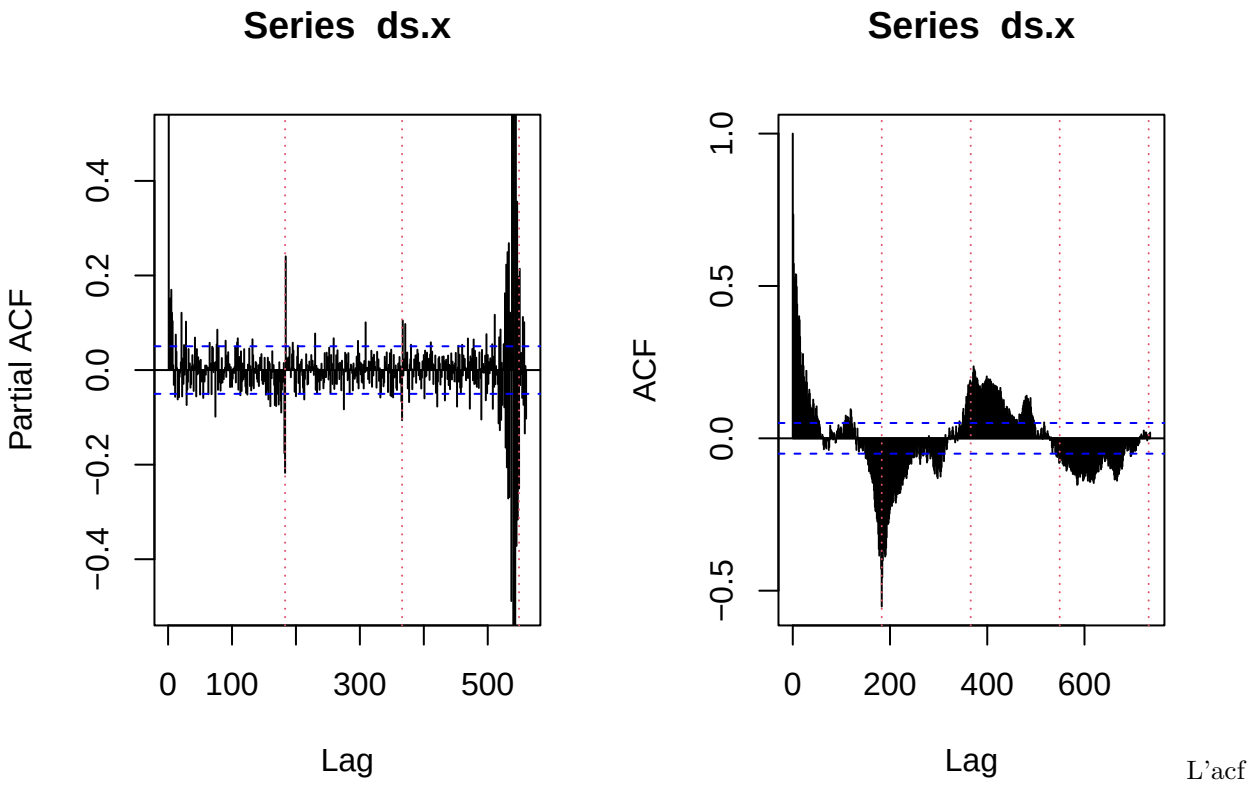
On va alors supposer l'existence d'une période semestriel ($s = 183$). Dans ce cas, on différencie par rapport à cette période pour que le choix du modèle soit plus simple.

```
ds.x <- diff(log.x.train, lag = 183) # Opetateur (I-B^183)
plot.ts(ds.x)
abline(h=0, col="red", lty=3)
```



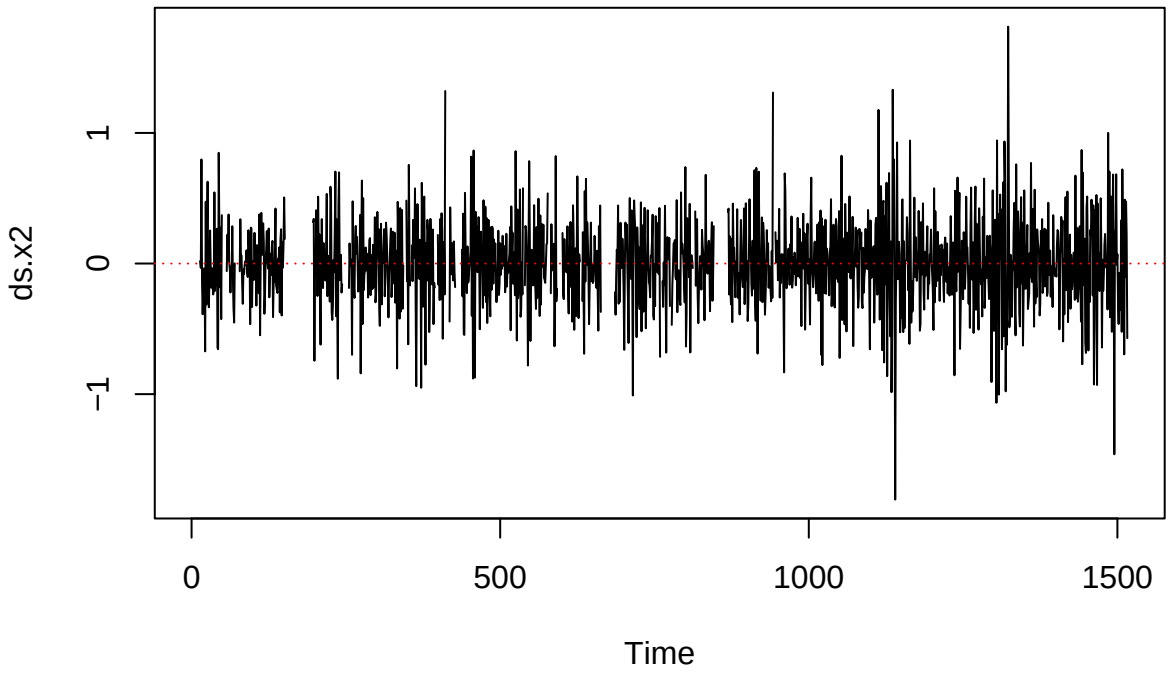
La série différenciée ne semble pas encore stationnaire. On peut observer cela en regardant l'acf :

```
par(mfrow=c(1,2))
pacf(ds.x, na.action = na.pass, lag.max = 560, ylim=c(-0.5, 0.5))
abline(v=c(183,366, 549, 732), col=2, lty=3)
acf(ds.x, na.action = na.pass,lag.max = 735)
abline(v=c(183,366, 549, 732), col=2, lty=3)
```



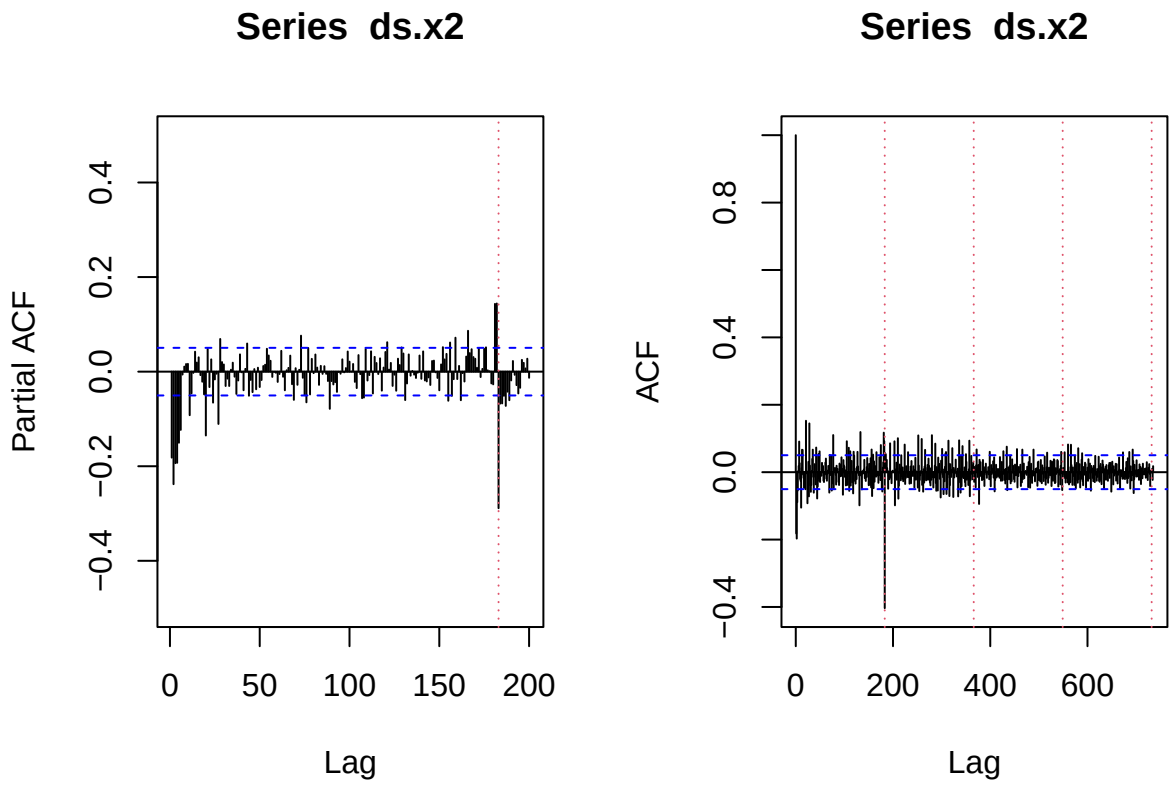
L'acf $\rho(h)$ reste encore trop significative (trop grande par rapport aux pointillés) même si h est assez grand. Nous avons testé une deuxième différenciation saisonnière mais cela n'a rien changé. Même si l'on n'observe pas une tendance linéaire (ou polynomiale), on va tenter une différentiation du type $\Delta(B) = I - B$:

```
ds.x2 <- diff(ds.x)
plot.ts(ds.x2)
abline(h=0, col="red", lty=3)
```



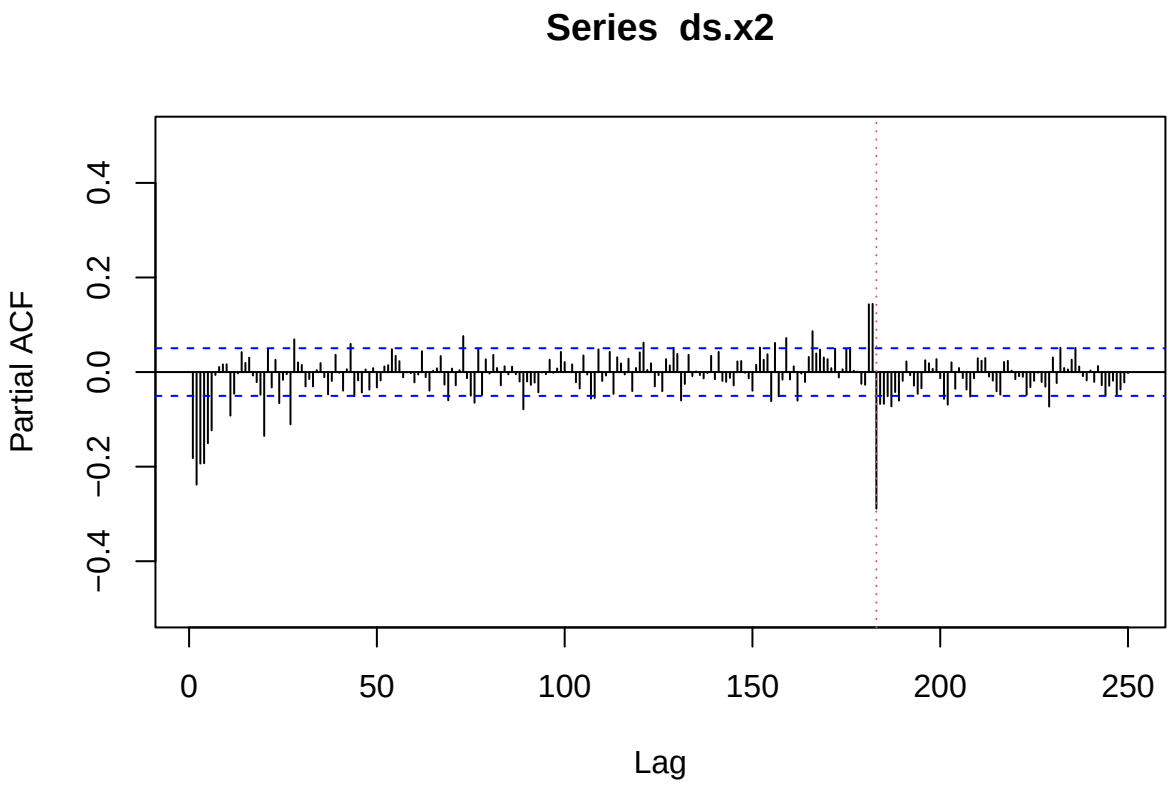
Cette fois-ci la série semble stationnaire. On va regarder maintenant sa acf et pacf.

```
par(mfrow=c(1,2))
pacf(ds.x2, na.action = na.pass, lag.max = 200, ylim=c(-0.5, 0.5))
abline(v=c(183,366, 549, 732), col=2, lty=3)
acf(ds.x2, na.action = na.pass,lag.max = 735)
abline(v=c(183,366, 549, 732), col=2, lty=3)
```



- Pour la pacf, par rapport à l'attendu de la théorie, on observe un comportement assez inusual pour les lags h supérieurs à 400. Cependant, on va ignorer ce comportement, et en regardant les lags multiples de la période $s = 183$, on peut choisir $P_{max} = 3$. La pacf entre les lags multiples de la période $s = 183$ ne sont pas bien définies. Pour simplifier le choix, on va se concentrer autour du lag=183 :

```
pacf(ds.x2, na.action = na.pass, lag.max = 250, ylim=c(-0.5, 0.5))
abline(v=c(183), col=2, lty=3)
```



Pour simplicité, on peut choisir $p_{max} = 6$.

- Concernant, l'acf, il semble clair que $Q_{max} = 1$ (acf fignificative sur un seul lag multiple de la période $s = 183$). Finalement, les autocorrélations entre les lags multiples de la période $s = 183$ se semblent pas bien définies mais en tout cas ne sont pas très grandes par rapport aux pointillés. On pourrait dire que $q_{max} = 0$. Cependant, pour nous rassurer de notre choix, on va choisir $q_{max} = 7$.

Nous devons dont estimer les paramètre de

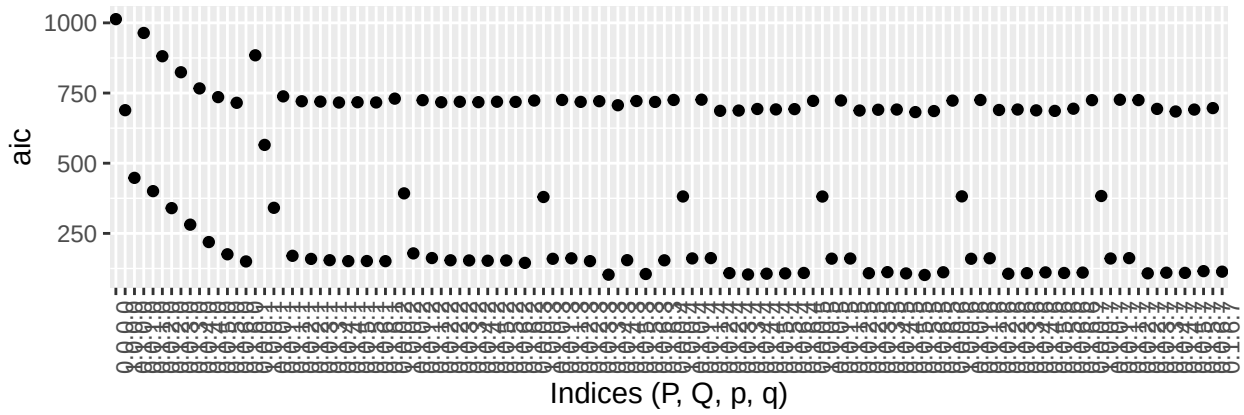
$$N = (P_{max} + 1) \times (Q_{max} + 1) \times (p_{max} + 1) \times (q_{max} + 1)$$

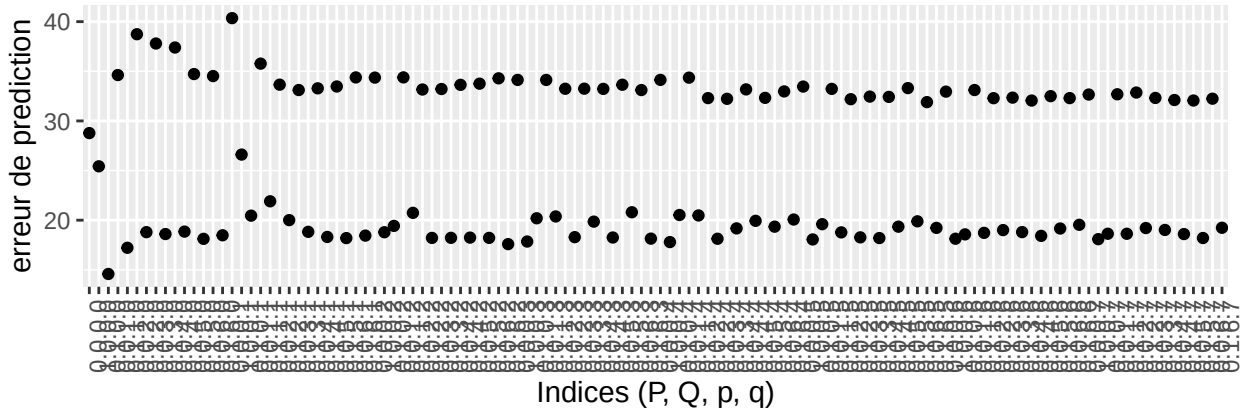
modèles.

```
log.x.train.ts <- ts(log.x.train, start = c(2013,01,04), frequency = 365.25)
set.seed(911)
s = 183
P.max = 3
Q.max = 1
p.max = 6
q.max = 7

Estimation = TRUE
if (Estimation == FALSE){
Resultats <- data.frame(P=NA, Q=NA, p=NA, q=NA, aic=NA, log.lik = NA, p.valeur=NA,
                        erreur.prediction = NA)
```

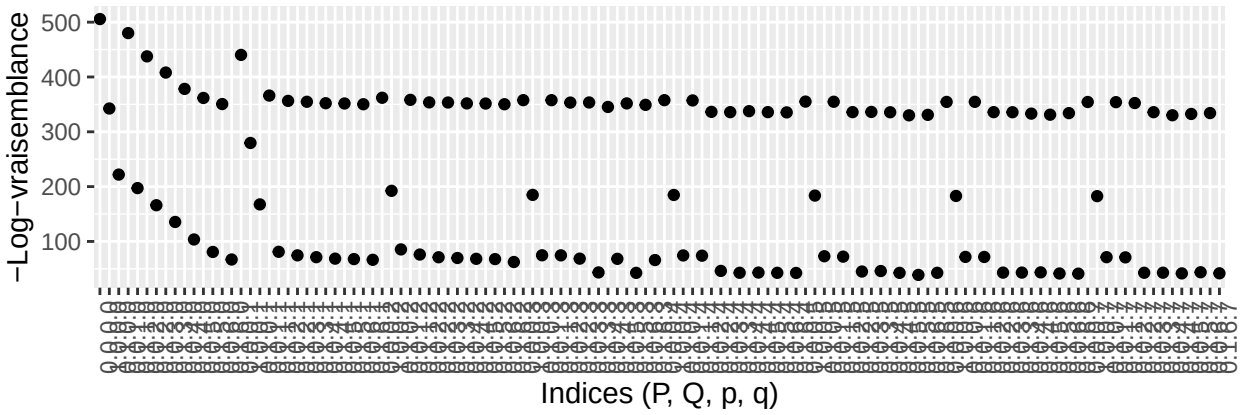
```
k = 1
print(c(((P.max+1)*(Q.max+1)*(p.max+1)*(q.max+1)), "modèles"))
for (P in 0:P.max){
  for (Q in 0:Q.max){
    for (p in 0:p.max){
      for (q in 0:q.max){
        #print(k)
        mod.aux <- arima(log.x.train.ts, order=c(p,1,q),
                         seasonal= list(order = c(P,1,Q), period = 183),
                         method = "CSS-ML" )
        pred.aux <- predict(mod.aux, n.ahead=length(x.test))
        erreur.aux <- sqrt(mean((exp(pred.aux$pred) - x.test)^2))
        Resultats[k,] <- c(P,Q,p,q, mod.aux$aic, - mod.aux$loglik,
                           Box.test(mod.aux$residuals)$p.value, erreur.aux)
        k <- k + 1
      }
    }
  }
}
save(Resultats, file = "Resultats_Exercice1_1_vf2.RData")
}else
  load("Resultats_Exercice1_1_vf2.RData")

cat("\f")
```

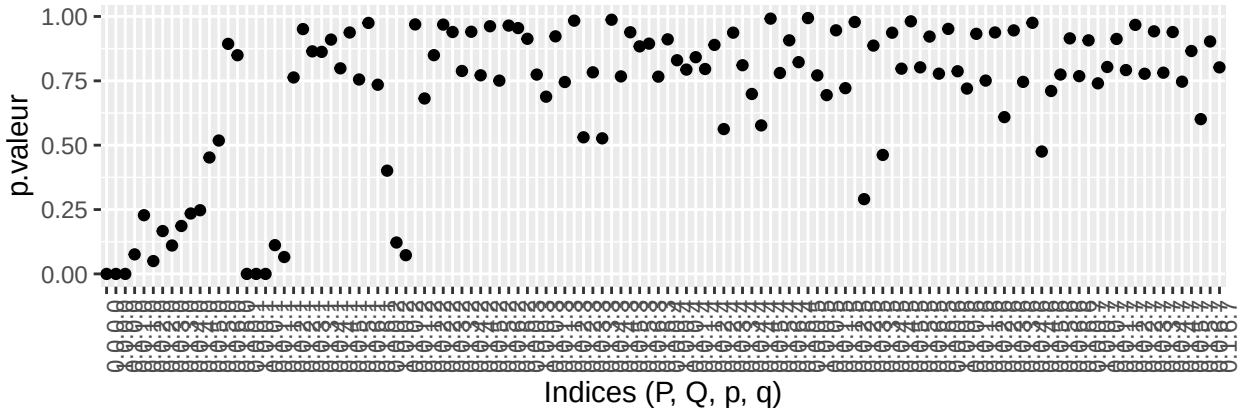
```r
library(ggplot2)
library(gridExtra)
p1 <- ggplot(Resultats, aes(x = interaction(P, Q, p, q), y = aic)) +
  geom_point() +
  labs(x = "Indices (P, Q, p, q)", y = "aic") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

p2 <- ggplot(Resultats, aes(x = interaction(P, Q, p, q), y = erreur.prediction)) +
  geom_point() +
  labs(x = "Indices (P, Q, p, q)", y = "erreur de prediction") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

grid.arrange(p1, p2, ncol = 1)
```



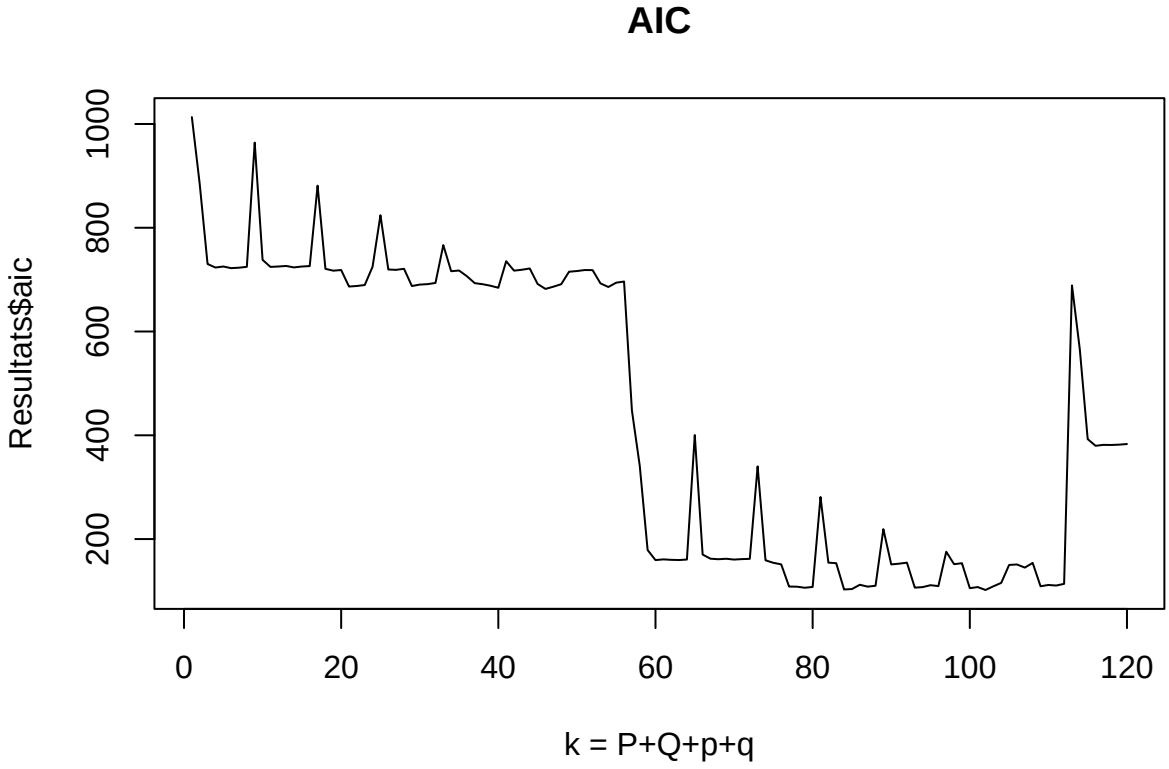



```r
p3 <- ggplot(Resultats, aes(x = interaction(P, Q, p, q), y = log.lik)) +
  geom_point() +
  labs(x = "Indices (P, Q, p, q)", y = "-Log-vraisemblance") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

p4 <- ggplot(Resultats, aes(x = interaction(P, Q, p, q), y = p.valeur)) +
  geom_point() +
  labs(x = "Indices (P, Q, p, q)", y = "p.valeur") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

grid.arrange(p3, p4, ncol = 1)
```

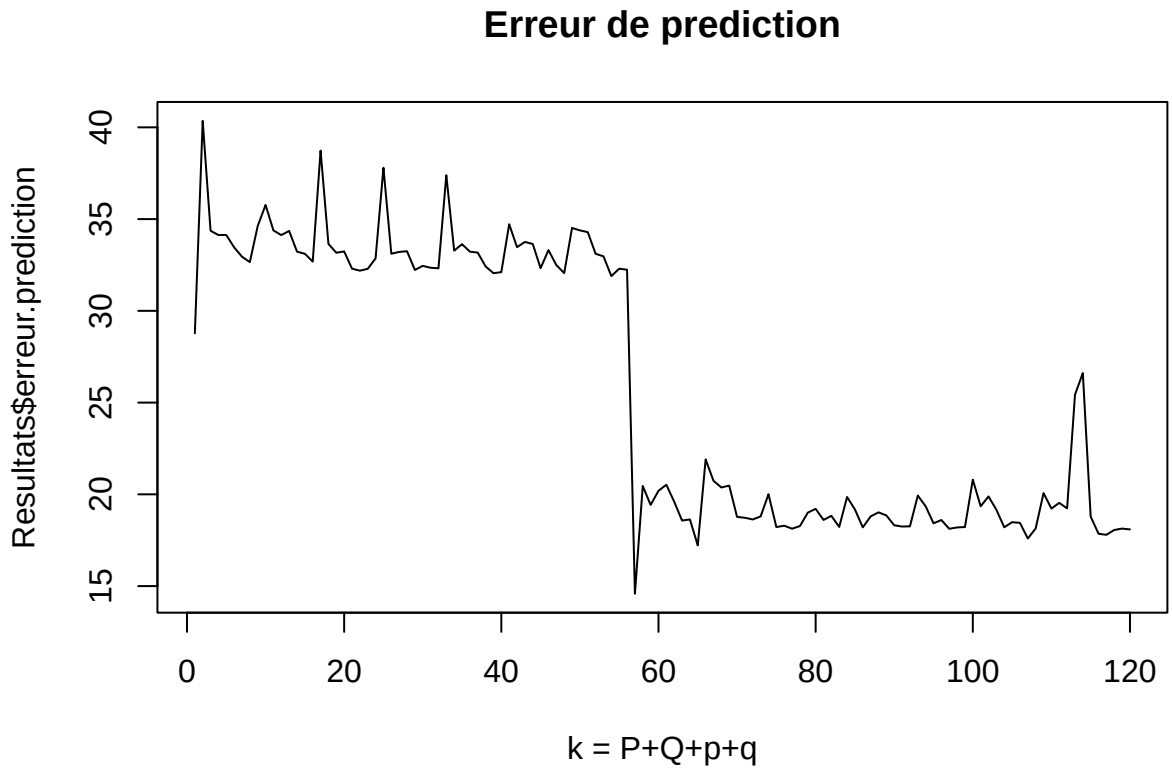




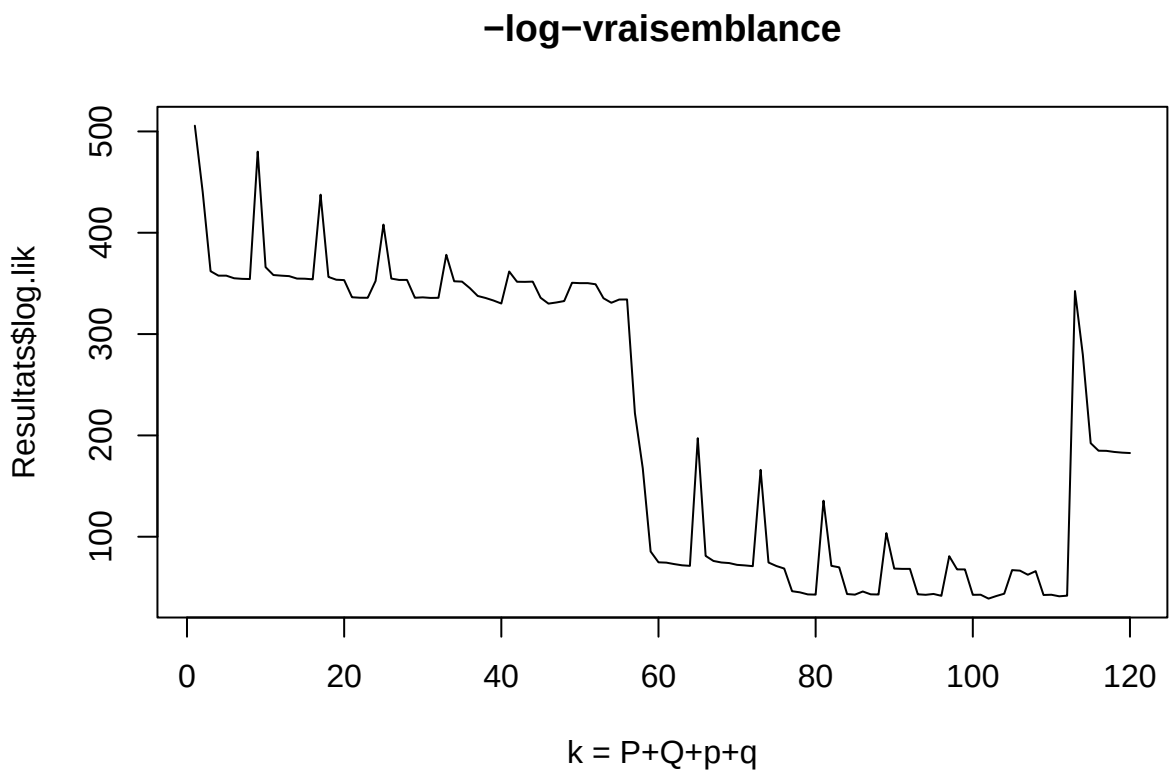

```
#par(mfrow=c(2,1))
plot.ts(Resultats$aic, main="AIC", xlab="k = P+Q+p+q")
```



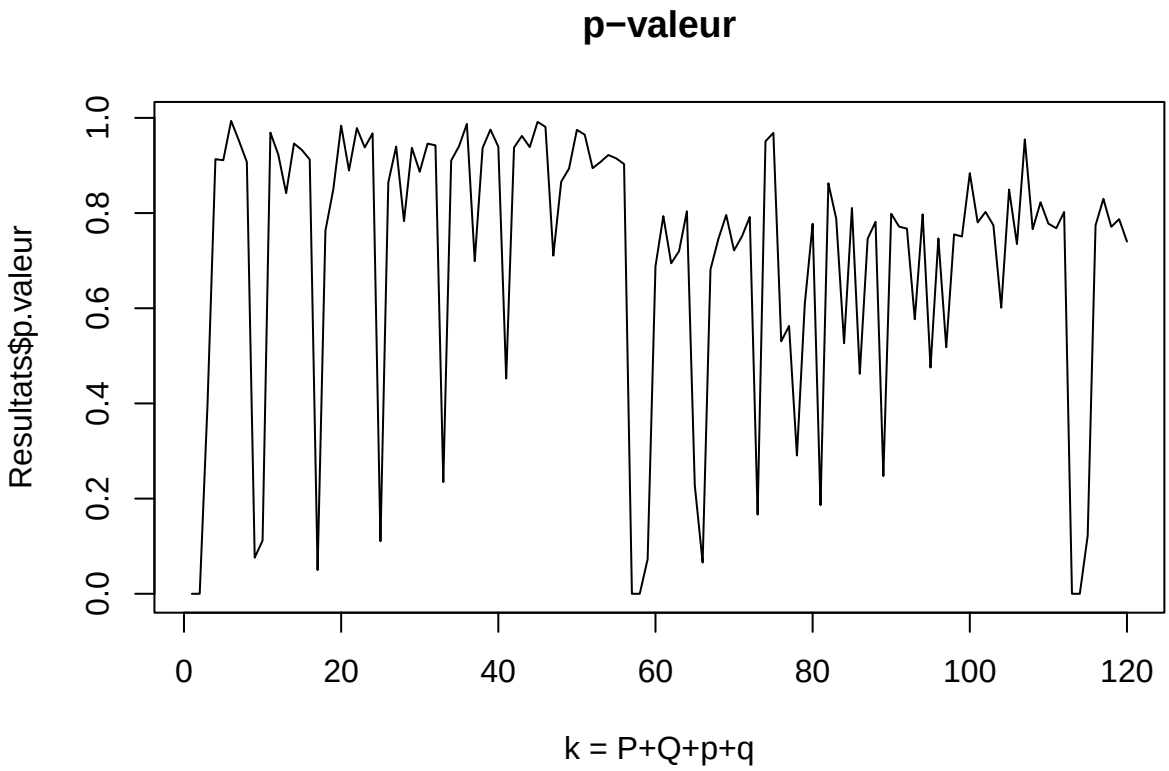
```
plot.ts(Resultats$erreur.prediction, main="Erreur de prediction", xlab="k = P+Q+p+q")
```




```
plot.ts(Resultats$log.lik, main="-log-vraisemblance", xlab="k = P+Q+p+q")
```



```
plot.ts(Resultats$p.valeur, main="p-valeur", xlab="k = P+Q+p+q")
```



Choix du modèle :

```
ordre = Resultats[which(Resultats$aic == min(Resultats$aic)),1:4]
print(ordre)
```

```
    P Q p q
102 0 1 5 5
```

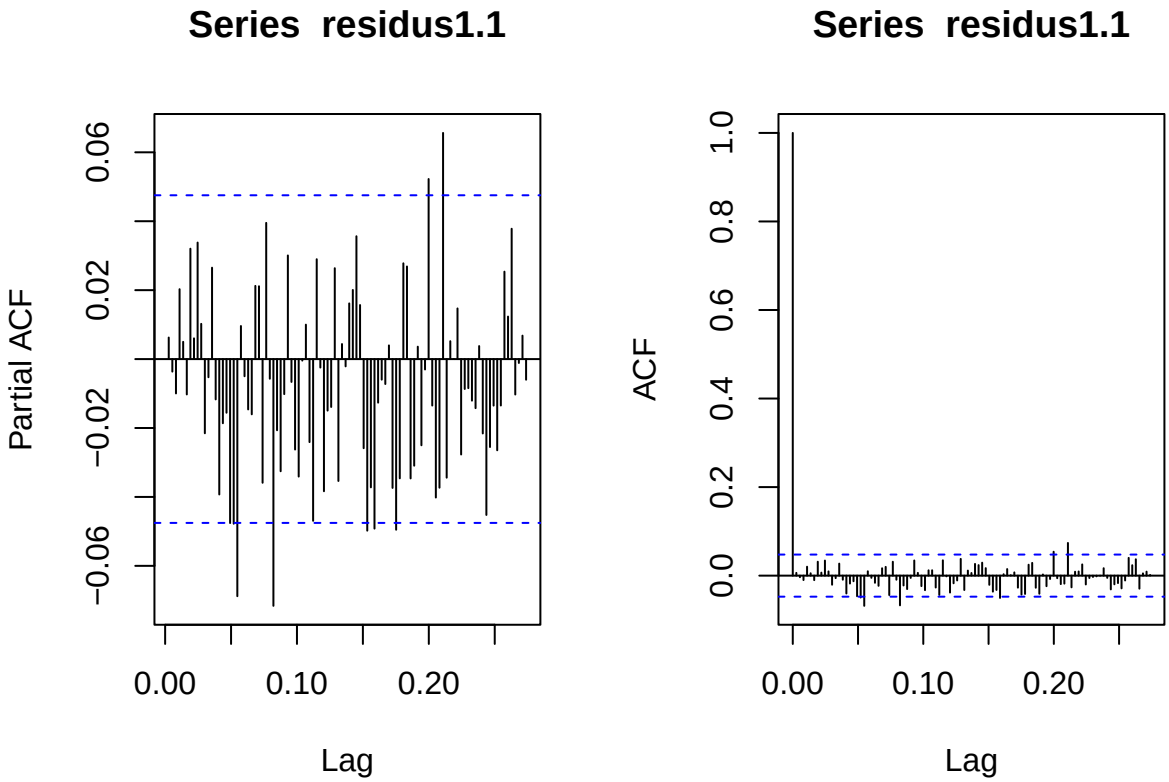
```
estimated_model = TRUE
if (estimated_model==FALSE){
  sarima_final <- arima(log.x.train.ts, order=c(ordre$p,1,ordre$q),
                        seasonal= list(order = c(ordre$P,1,ordre$Q), period = 183),
                        method = "CSS-ML" )
  save(sarima_final, file = "Sarima_Exercice1_1_vf2.RData")
}else
  load("Sarima_Exercice1_1_vf2.RData")

sarima_final
```

```
Call:
arima(x = log.x.train.ts, order = c(ordre$p, 1, ordre$q), seasonal = list(order = c(ordre$P,
    1, ordre$Q), period = 183), method = "CSS-ML")

Coefficients:
          ar1     ar2      ar3      ar4     ar5      ma1      ma2     ma3
      -0.1844  0.5994  -0.7995  -0.6572  0.3850  -0.2086  -0.9243  0.9048
s.e.   0.0474  0.0440   0.0226   0.0374  0.0475   0.0355   0.0375  0.0297
         ma4      ma5     sma1
```

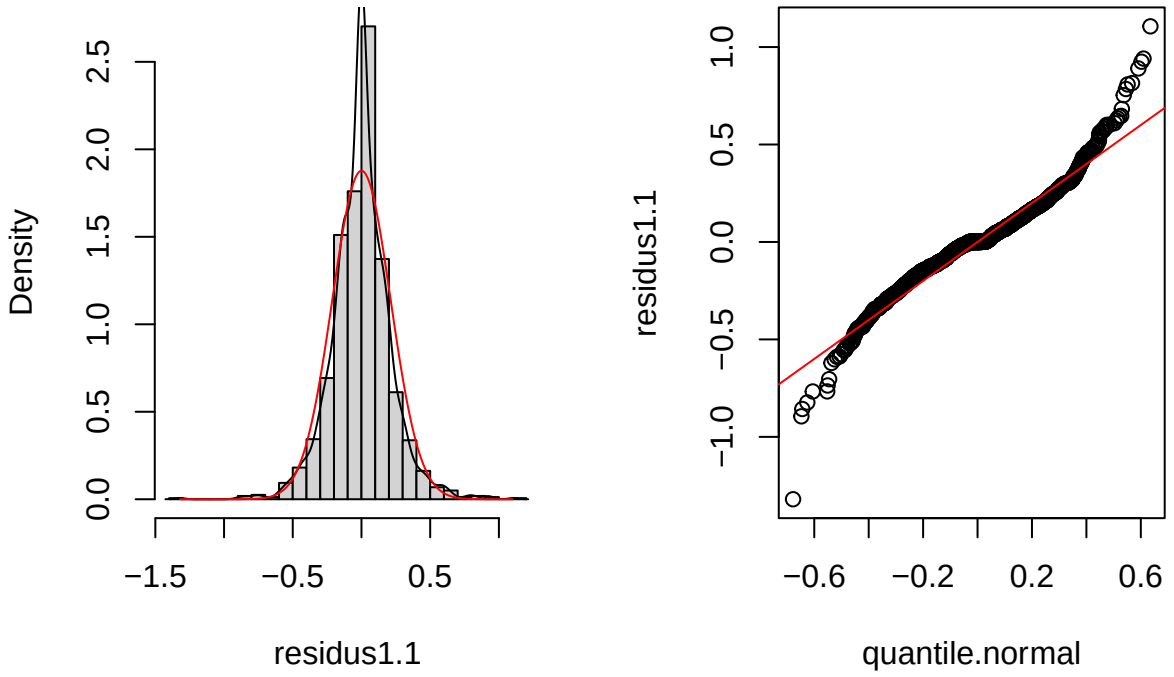
```
      0.4136  -0.7891  -0.8812
s.e.  0.0300   0.0403   0.0540

sigma^2 estimated as 0.05097:  log likelihood = -38.93,  aic = 101.86
```

On regarde le comportement des reridus :

```r
residus1.1 <- sarima_final$residuals
par(mfrow=c(1,2))
pacf(residus1.1, na.action = na.pass, lag.max = 100)
acf(residus1.1, na.action = na.pass, lag.max = 100)
```

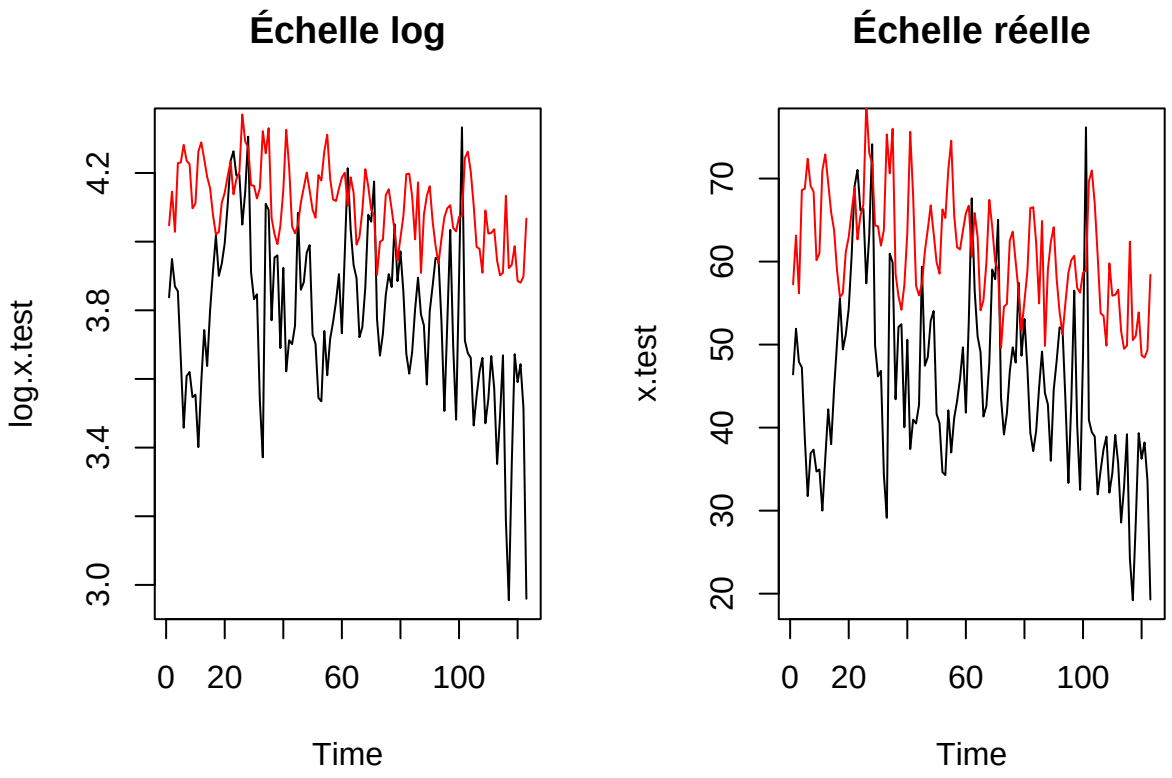



```r
par(mfrow=c(1,2))
hist(residus1.1, probability = TRUE, breaks = 20)
lines(density(residus1.1, na.rm = TRUE))
t <- seq(min(residus1.1, na.rm = TRUE), max(residus1.1, na.rm = TRUE), by=0.01)
lines(t,dnorm(t, sd=sd(residus1.1, na.rm = TRUE)), col="red")

set.seed(912)
quantile.normal <- rnorm(length(residus1.1),sd=sd(residus1.1, na.rm = TRUE), mean = mean(residus1.1, na
qqplot(quantile.normal, residus1.1)
abline(a=0, b=1, col="red")
```



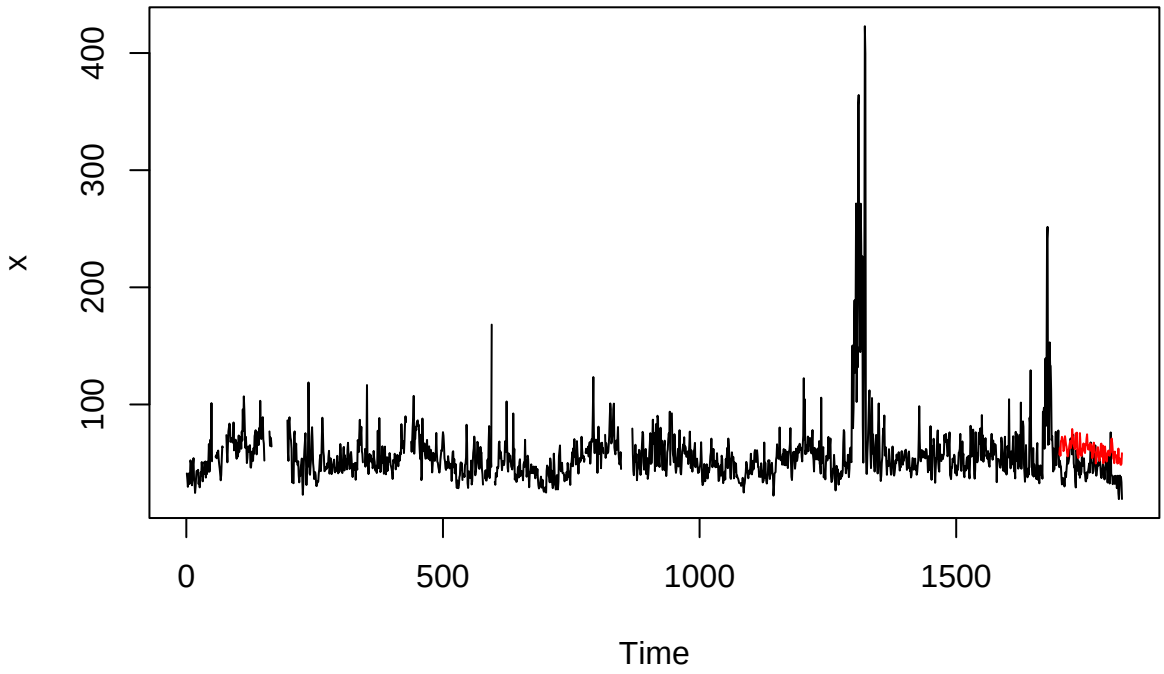
On regarde maintenant la qualité de prédiction :

```
RUN=FALSE
if (RUN==TRUE){
  log.pred1 <- predict(sarima_final, n.ahead=length(log.x.test))
  save(log.pred1, file = "Test_prediction_1_1_vf2.RData")
}else
  load("Test_prediction_1_1_vf2.RData")

par(mfrow=c(1,2))
plot.ts(log.x.test, main="Échelle log")
lines(as.numeric(log.pred1$pred), col="red")

plot.ts(x.test, main="Échelle réelle")
lines(exp(as.numeric(log.pred1$pred)), col="red")
```



Vue globale :

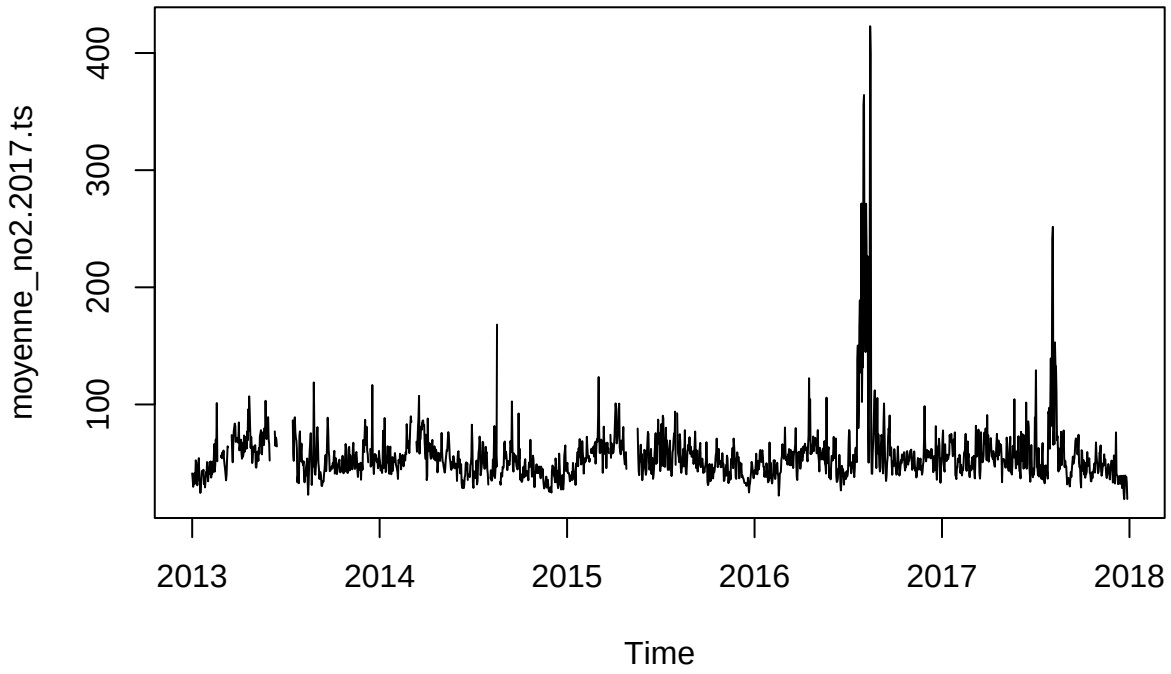
```
plot.ts(x)
lines(1701:1823, exp(as.numeric(log.pred1$pred)), col="red")
```



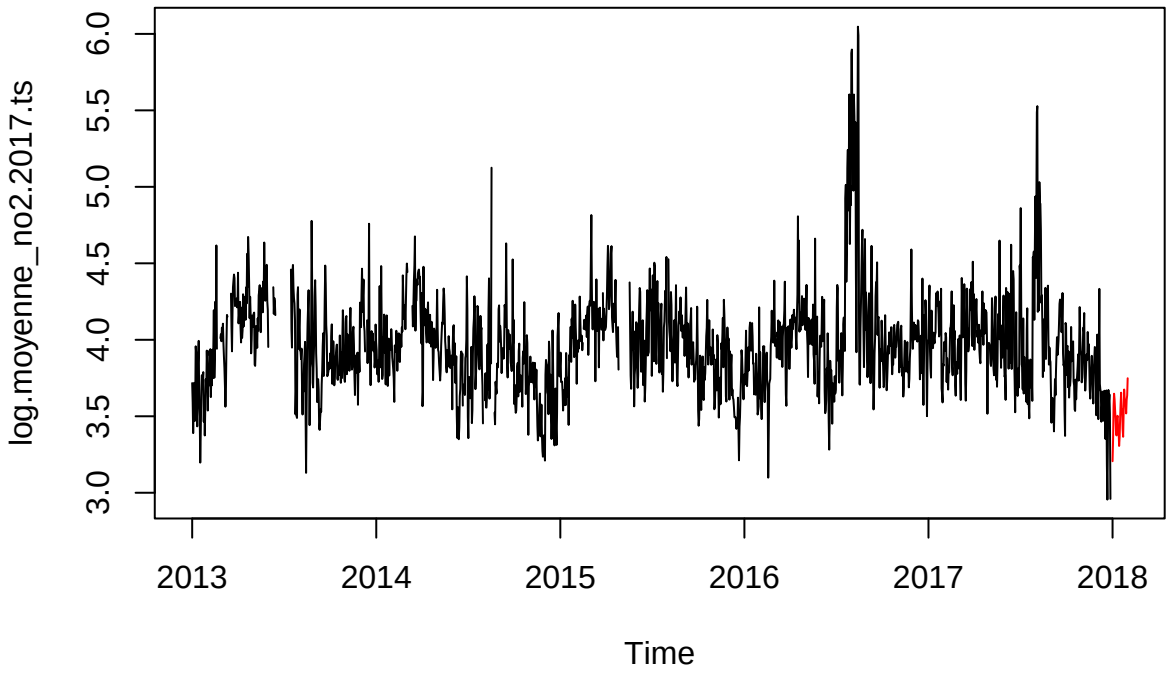
1.3.- Réaliser une prédiction des mesures moyennes journalières de NO2 pour janvier 2018.

```
moyenne_no2.2017 <- moyenne_journaliere_no2.2017$moyenne_no2
moyenne_no2.2017.ts <- ts(moyenne_no2.2017, start = c(2013,1,4), frequency = 365.25)
```

```r
plot(moyenne_no2.2017.ts)
```




```r
log.moyenne_no2.2017.ts <- log(moyenne_no2.2017.ts)
RUN = FALSE
if (RUN==TRUE){
  sarima.update <- arima(log.moyenne_no2.2017.ts, order=c(ordre$p,1,ordre$q),
                         seasonal= list(order = c(ordre$P,1,ordre$Q), period = 183),
                         method = "CSS-ML" )
  save(sarima.update, file = "sarima_update_vf2.RData")
  }else
    load("sarima_update_vf2.RData")

prediction.janvier2018 <- predict(sarima.update, n.ahead=31)
hat.no2.janvier2018 <- ts(as.numeric(prediction.janvier2018$pred), start = c(2018,01,01), frequency = 3
plot(log.moyenne_no2.2017.ts, xlim = c(2013,2018.08))
lines(hat.no2.janvier2018, col="red")
```

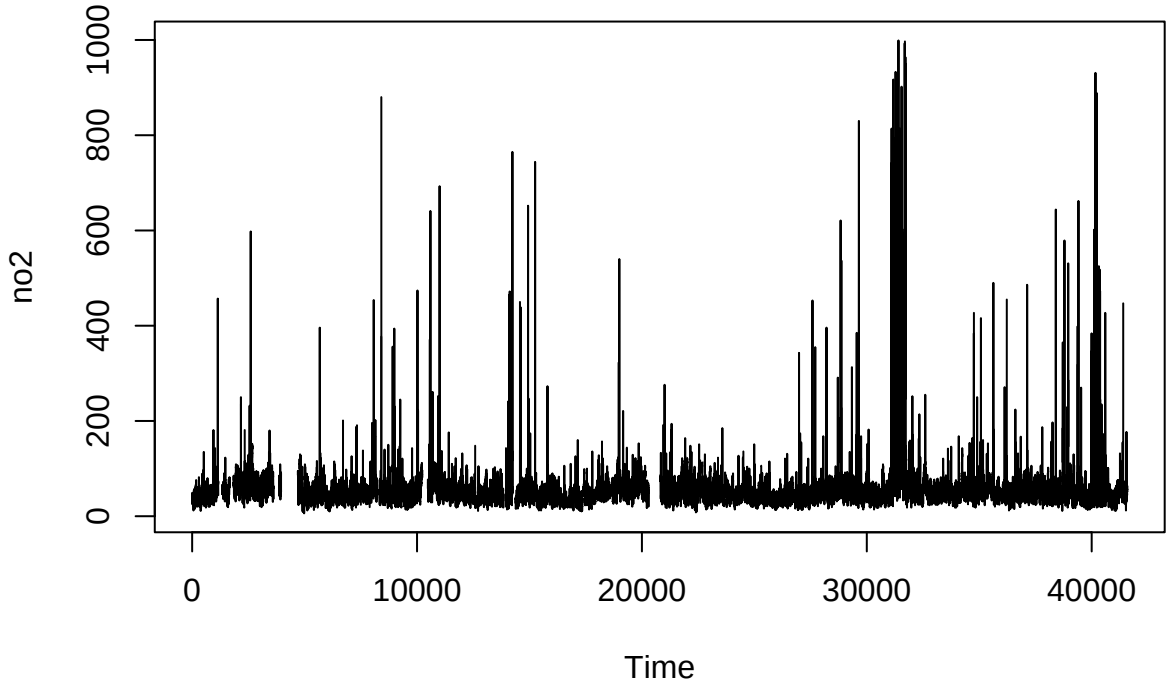


Deuxième partie : modélisation et prévision des données horaires de NO2

2.1.- Appliquer la méthode de Box-Jenkins pour proposer un modèle (S)ARIMA pour les mesures horaires de NO2. N'oubliez pas de choisir un échantillon de test pour vérifier la qualité de prédiction.

Remarque : les mesures horaires presentent une variance assez irréguilière, vous pouvez donc travailler avec l'échelle logarithmique afin d'obtenir une variance plus réguilière.

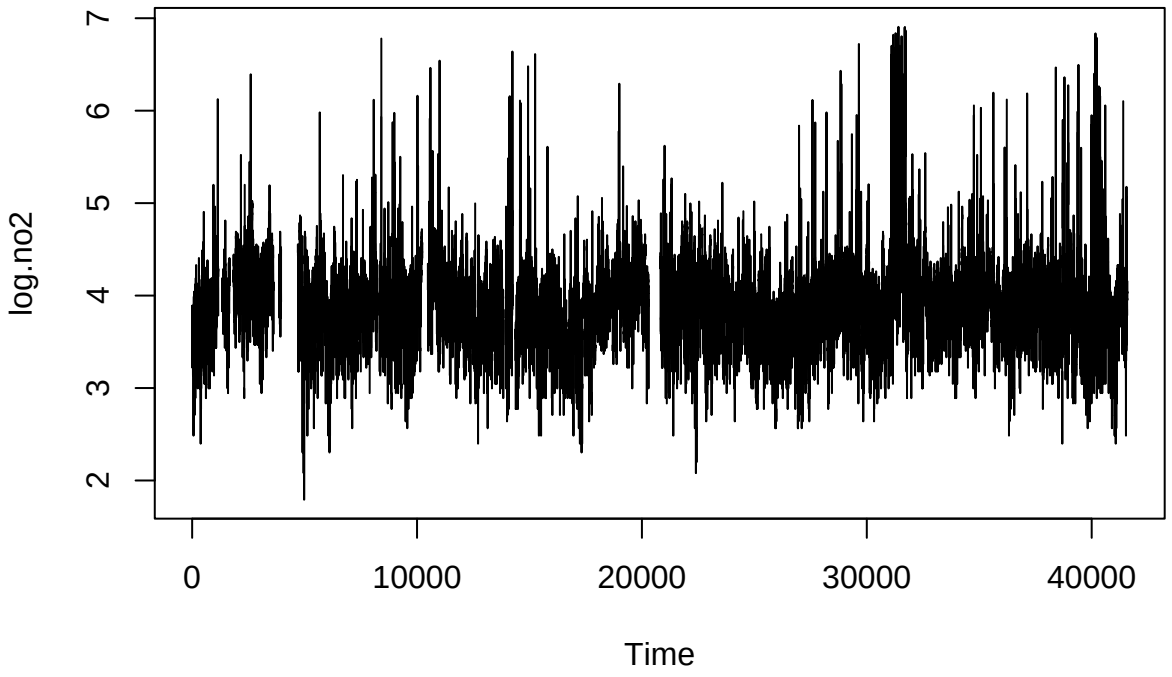
```
no2 <- no2.df$NO2[1:41584] #Données jusqu'au 31 décembre 2017 à 23h
plot.ts(no2)
```



Vari-

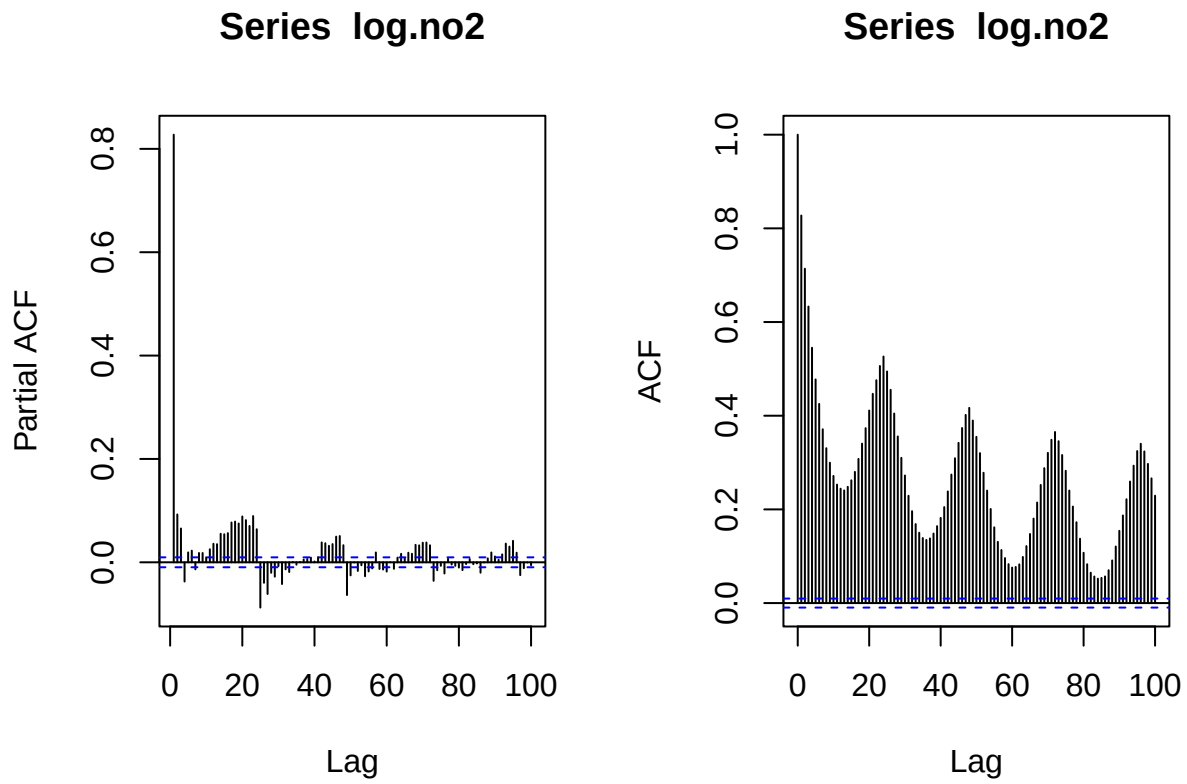
ance trop variable/irréguilière, on dirait qu'elle dépende du temps. On va alors travailler avec l'échelle logarithmique :

```
log.no2 <- log(no2)
plot.ts(log.no2)
```



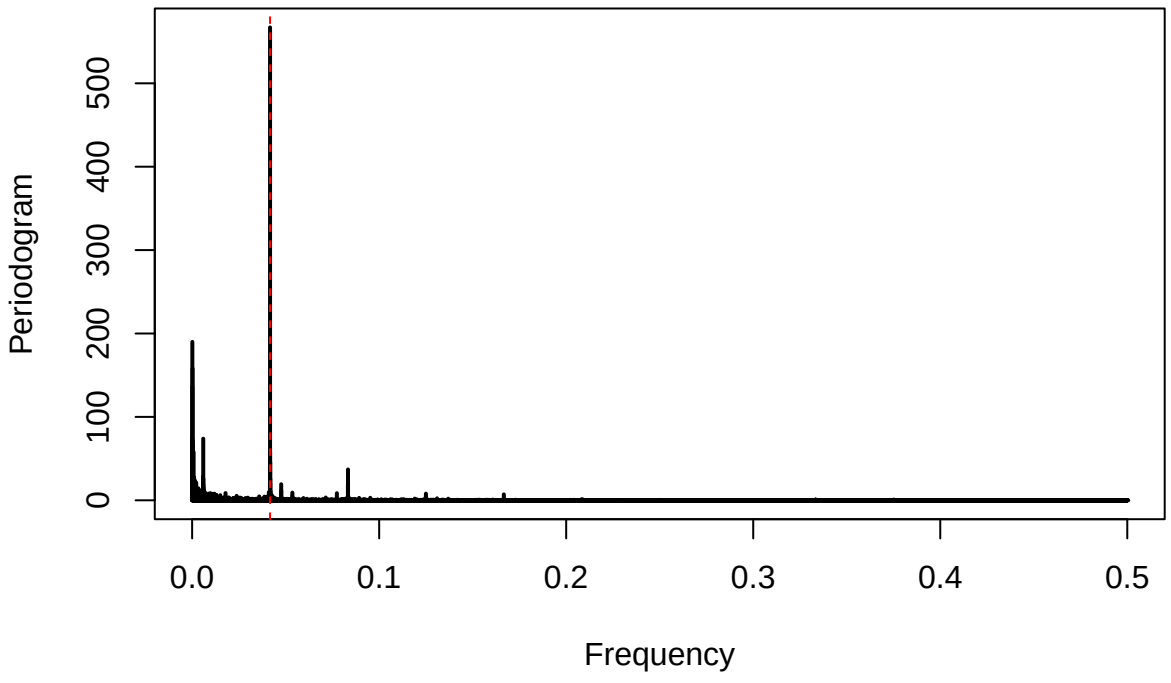
Diagnostic de stationnarité avec l'acf :

```
par(mfrow=c(1,2))
pacf(log.no2, na.action = na.pass, lag.max = 100)
acf(log.no2, na.action = na.pass, lag.max = 100)
```



L'acf montre bien la non-stationnarité de la série (pas de décroissance exponentielle vers zéro). De plus, on observe une forme sinusoïdale avec des sommets autour des multiples de 24. Il est donc probable que c'est dû à une périodicité journalière. On va le vérifier avec le périodogramme :

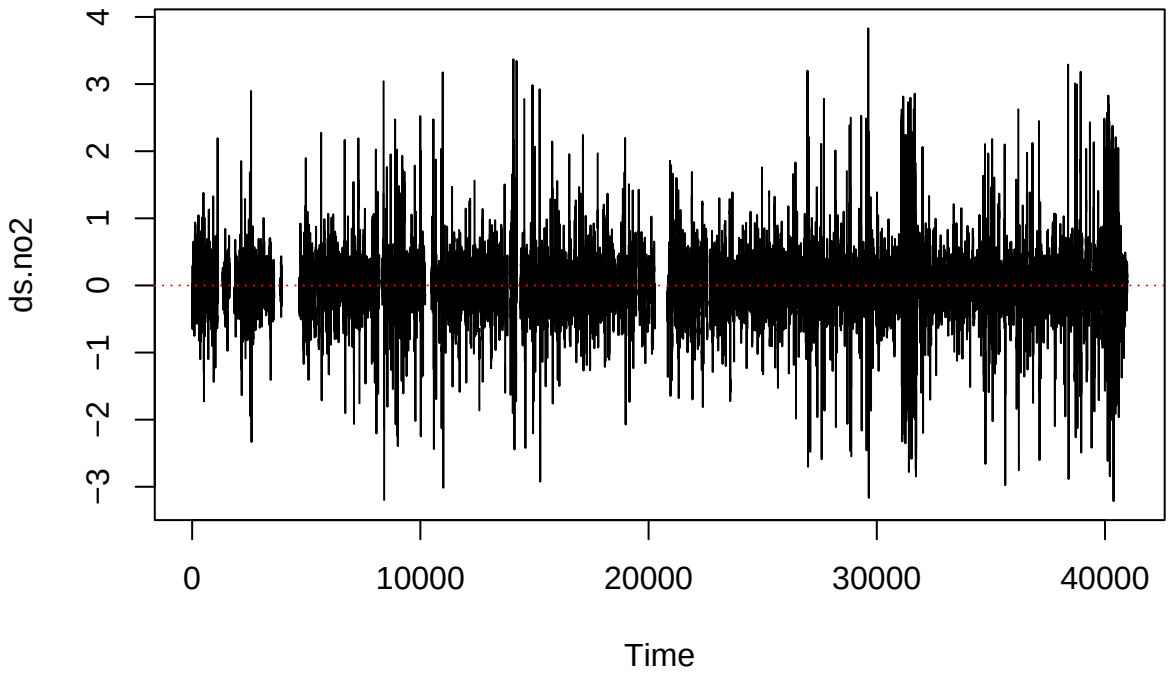
```
library(TSA)
x.imputed2 <- na.aggregate(log.no2, FUN = mean) #Pour faire marcher le periodogram de TSA
p <- TSA::periodogram(x.imputed2)
abline(v=1/24, col="red", lty=2)
```



```r
detach(package:TSA)
```

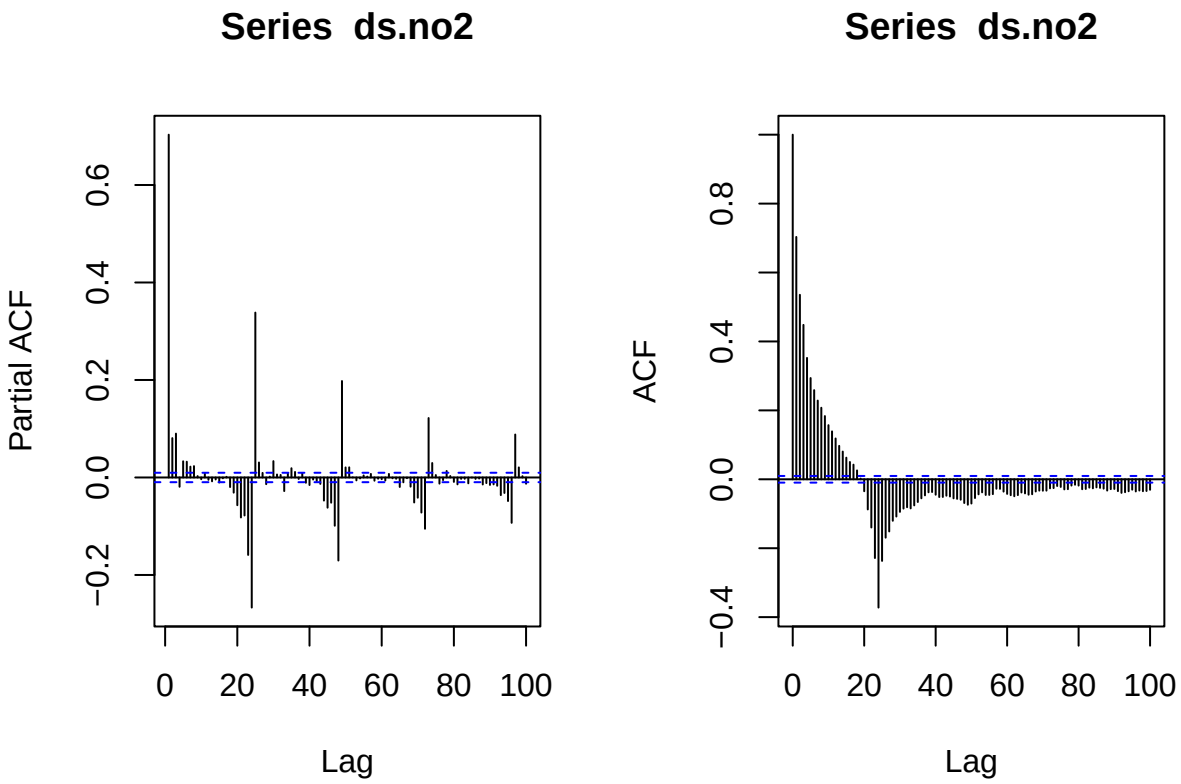
Cela confirme notre hypothèse : il y a une partie saisonnière dans la série de période $s = 24$. On va donc appliquer l'opérateur $\Delta_s(B) = (I - B^{24})$ afin de stationnariser la série :

```r
log.no2.ts <- ts(log.no2, start = c(2013,01,04,17), frequency = 24) #Données format ts
log.no2.train <- log.no2[1:41000] #Donnees d'entrainement (echelle log)
no2.test <- no2[41001:length(log.no2.ts)] #Donnees de test
log.no2.train.ts <- ts(log.no2.train, start = c(2013,01,04,17), frequency = 24) #Donnees d'entrainement
log.no2.test <- log.no2.ts[41001:length(log.no2.ts)] #Donnees de test echelle log.
ds.no2 <- diff(log.no2.train, lag=24)
plot.ts(ds.no2)
abline(h=0, col="red", lty=3)
```



Cela semble stationnaire. On regarde maintenant l'acf et pacf :

```r
par(mfrow=c(1,2))
pacf(ds.no2, na.action = na.pass, lag.max = 100)
acf(ds.no2, na.action = na.pass, lag.max = 100)
```



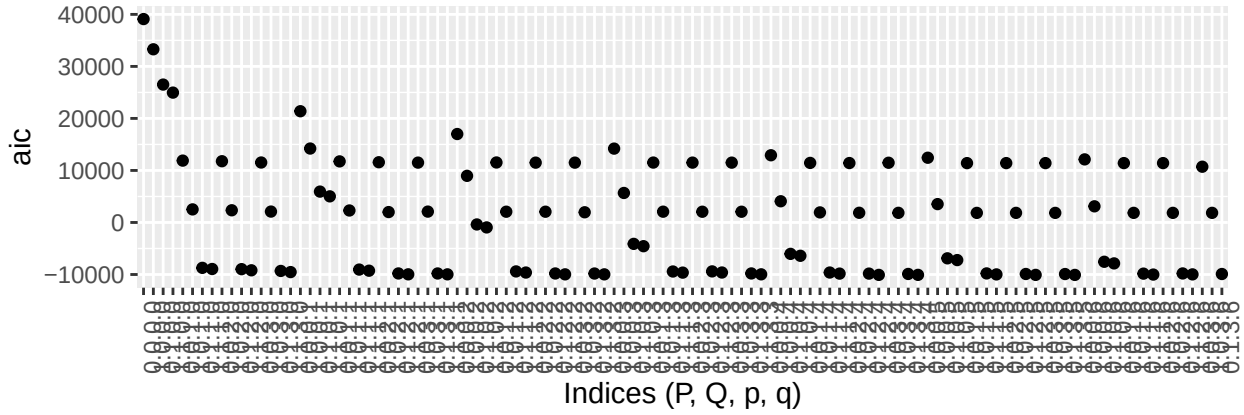
- En regardant la pacf, on observe des corrélations partielles significatives sur les lags multiples de 24. Le graphe suggère le choix de $P_{max} = 8$ (pour pouvez élargir la fenetre pour voir les autres valeurs avec `lag.max`). Par ailleurs, autour de chaque lag h multiple de 24, on observe que les corrélations partielles deminuent rapidement vers zéro (plus au moins à 6 pas autour de $h = 24k$). Cela veut dire qu'on peut choisir $p_{max} = 6$.
- En regardant l'acf, on observe une seule grande autocorrélation sur $h = 24$. On peut alors choisir $Q_{max} = 1$. Autour de cette grande corrélation, les autres autocorrélations restent assez significatives, ce qui nous force à choisir un $q_{max} = 12$.

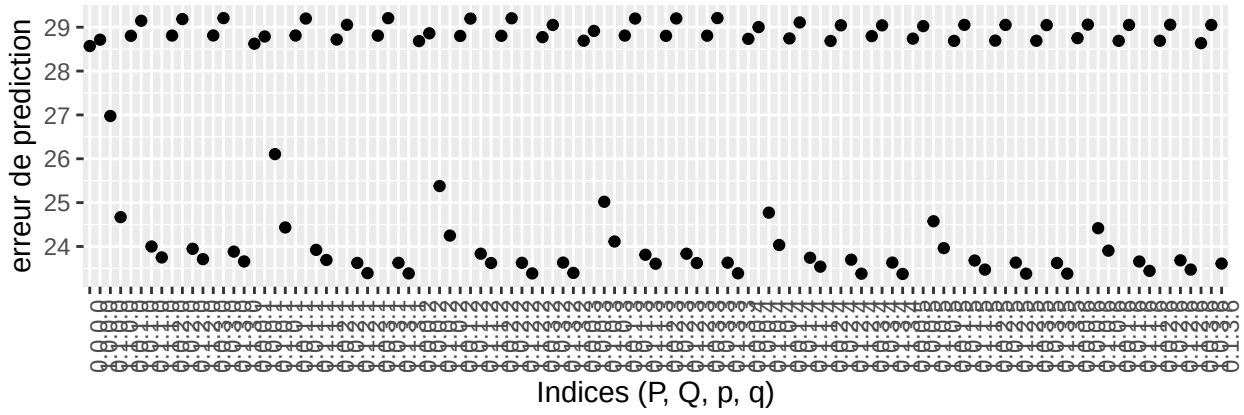
Tout cela implique que nous devons estimer 1512 modèles, ce qui n'est pas faisable dans un temps raisonable avec un ordinateur classique. Vous pouvez essayer d'optimiser l'algorithme suivant avec une parallélisation pour réduire le temps de calcul (Exercice).

```
set.seed(0115)
s = 24
#On va choisir P.max, p.max et q.max plus petit pour réduire le temps de calcul. Vous pouvez aller plus
P.max = 2 #8
Q.max = 1
p.max = 3 #6
q.max = 6 #12

Estimation = TRUE
if (Estimation == FALSE){
Resultats2 <- data.frame(P=NA, Q=NA, p=NA, q=NA, aic=NA, log.lik = NA, p.valeur=NA,
                         erreur.prediction = NA)
k = 1
print(c(((P.max+1)*(Q.max+1)*(p.max+1)*(q.max+1)), "modèles"))
for (P in 0:P.max){
  for (Q in 0:Q.max){
```

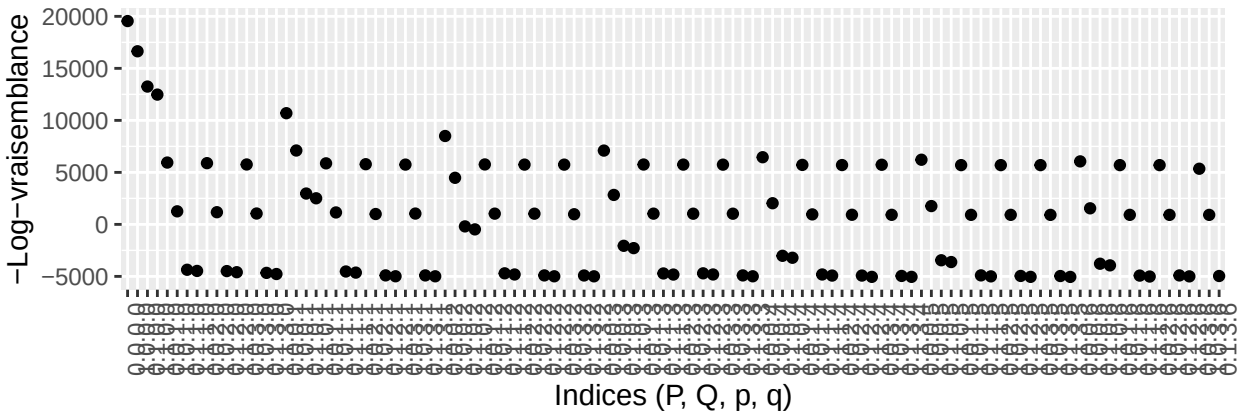
```
    for (p in 0:p.max){
      for (q in 0:q.max){
        #print(k)
        mod.aux <- arima(log.no2.train.ts, order=c(p,0,q),
                         seasonal= list(order = c(P,1,Q), period = 24),
                         method = "CSS-ML" )
        pred.aux <- predict(mod.aux, n.ahead=length(log.no2.test))
        erreur.aux <- sqrt(mean((exp(pred.aux$pred) - no2.test)^2, na.rm = TRUE))
        Resultats2[k,] <- c(P,Q,p,q, mod.aux$aic, - mod.aux$loglik,
                            Box.test(mod.aux$residuals)$p.value, erreur.aux)
        k <- k + 1
      }
    }
  }
}
save(Resultats2, file = "Resultats_Exercice1_2_vf.RData")
}else
  load("Resultats_Exercice1_2_vf.RData")
```

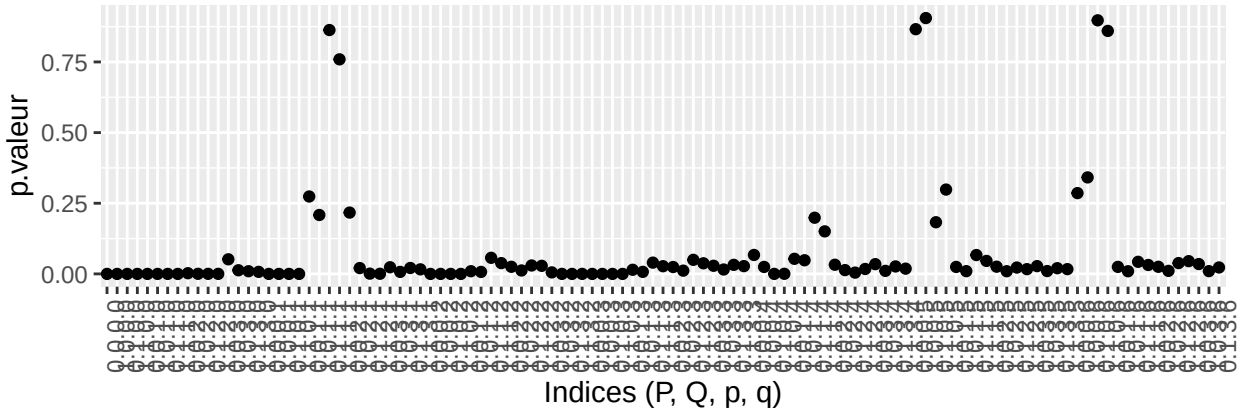
```
library(ggplot2)
library(gridExtra)
p1 <- ggplot(Resultats2, aes(x = interaction(P, Q, p, q), y = aic)) +
  geom_point() +
  labs(x = "Indices (P, Q, p, q)", y = "aic") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

p2 <- ggplot(Resultats2, aes(x = interaction(P, Q, p, q), y = erreur.prediction)) +
  geom_point() +
  labs(x = "Indices (P, Q, p, q)", y = "erreur de prediction") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

grid.arrange(p1, p2, ncol = 1)
```



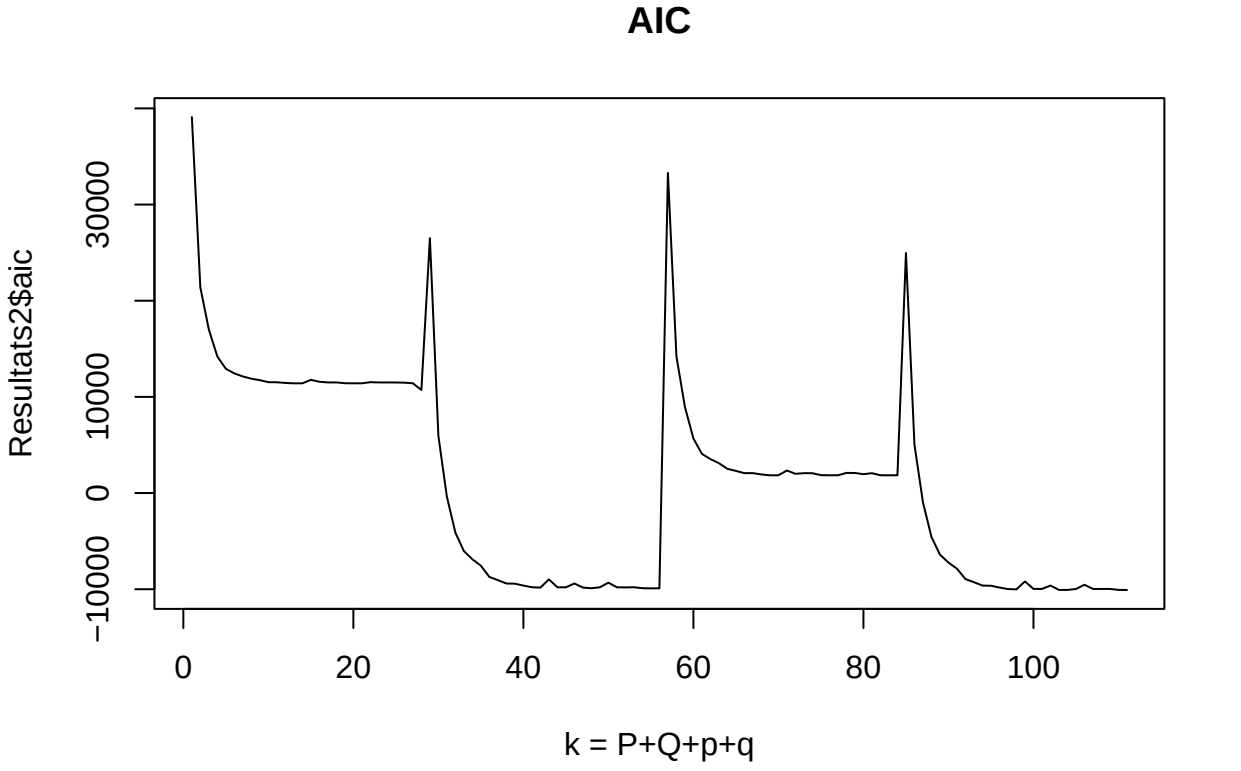



```r
p3 <- ggplot(Resultats2, aes(x = interaction(P, Q, p, q), y = log.lik)) +
  geom_point() +
  labs(x = "Indices (P, Q, p, q)", y = "-Log-vraisemblance") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

p4 <- ggplot(Resultats2, aes(x = interaction(P, Q, p, q), y = p.valeur)) +
  geom_point() +
  labs(x = "Indices (P, Q, p, q)", y = "p.valeur") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

grid.arrange(p3, p4, ncol = 1)
```

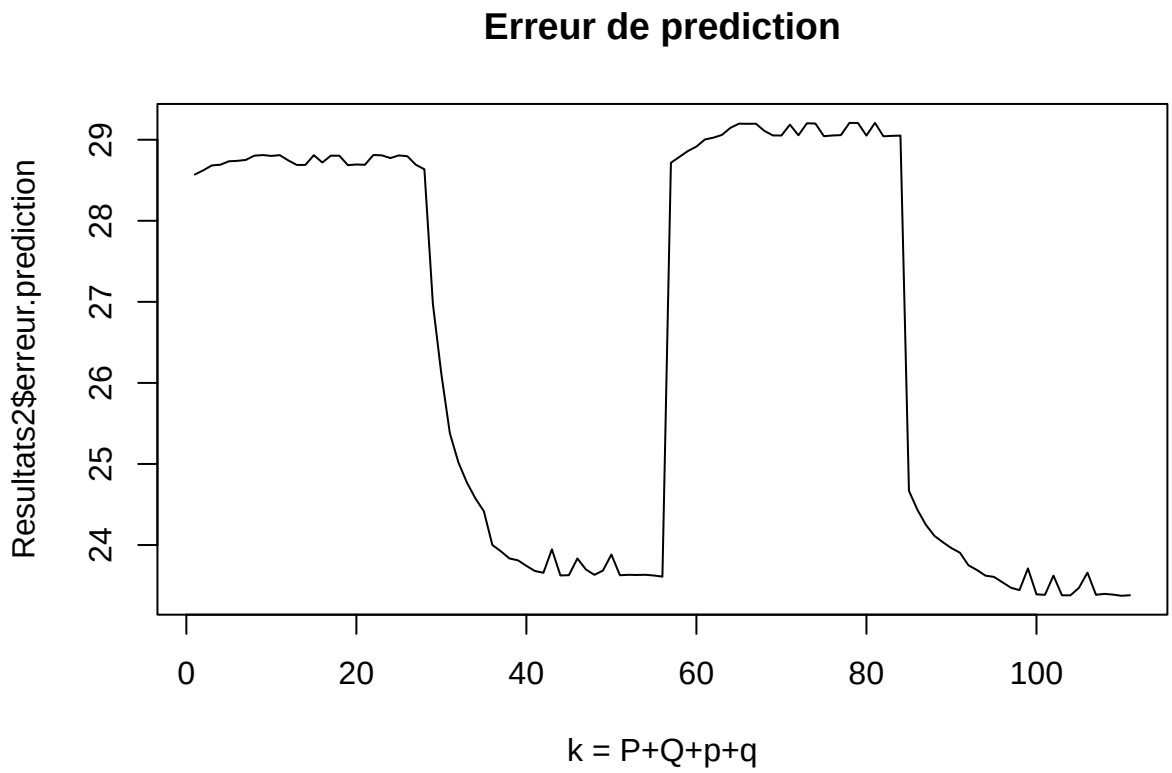




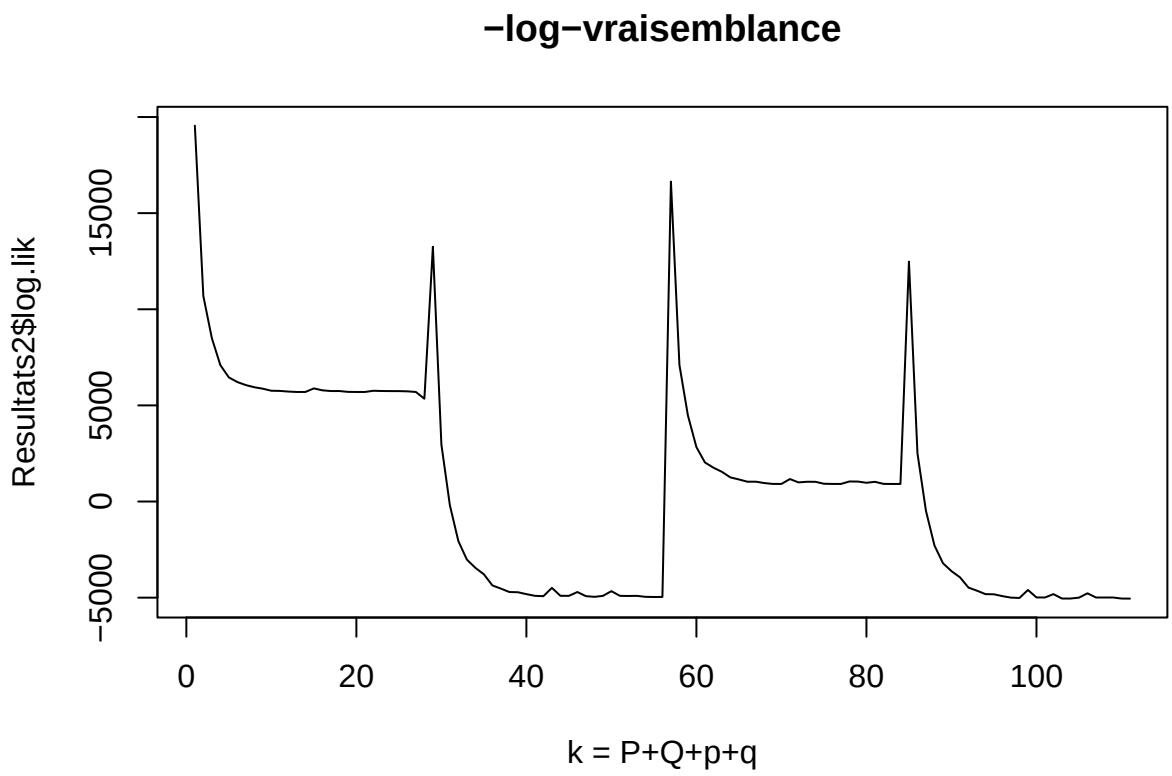

```
#par(mfrow=c(2,1))
plot.ts(Resultats2$aic, main="AIC", xlab="k = P+Q+p+q")
```



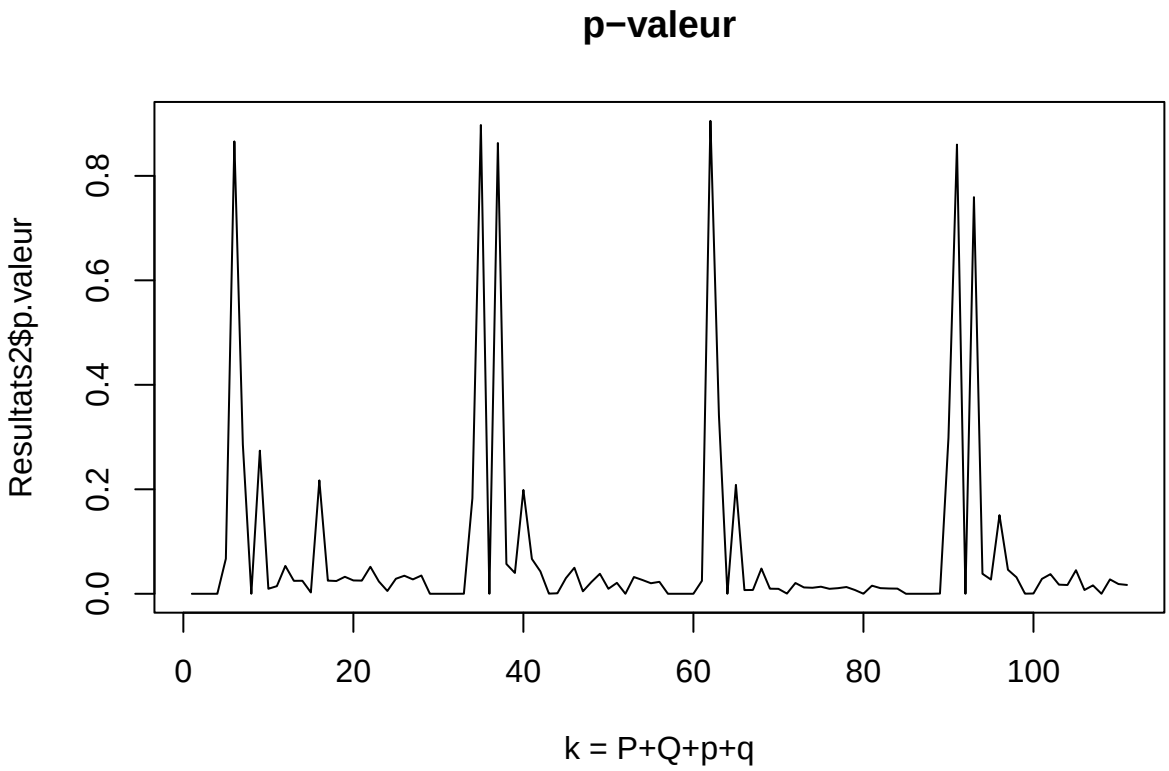
```r
plot.ts(Resultats2$erreur.prediction, main="Erreur de prediction", xlab="k = P+Q+p+q")
```




```r
plot.ts(Resultats2$log.lik, main="-log-vraisemblance", xlab="k = P+Q+p+q")
```



```
plot.ts(Resultats2$p.valeur, main="p-valeur", xlab="k = P+Q+p+q")
```



Choix du modèle :

```
K = which(Resultats2$aic == min(Resultats2$aic))
ordre = Resultats2[K,1:4]
K
```

```
[1] 111
```

```
print(ordre)
```

```
    P Q p q
111 1 1 3 5
```

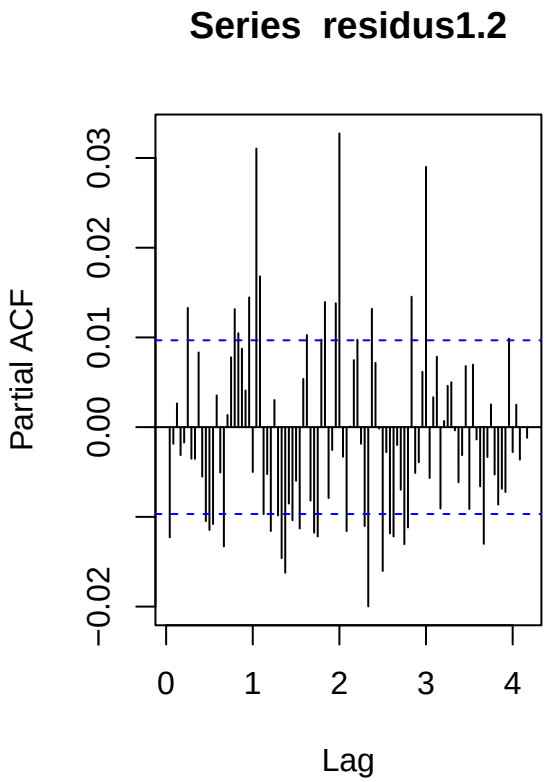
```
estimated_model = FALSE
if (estimated_model==TRUE){
  sarima_final2 <- arima(log.no2.train.ts, order=c(ordre$p,0,ordre$q),
                         seasonal= list(order = c(ordre$P,1,ordre$Q), period = 24),
                         method = "CSS-ML" )
  save(sarima_final2, file = "Sarima_Exercice1_2_vf.RData")
}else
  load("Sarima_Exercice1_2_vf.RData")

sarima_final2
```

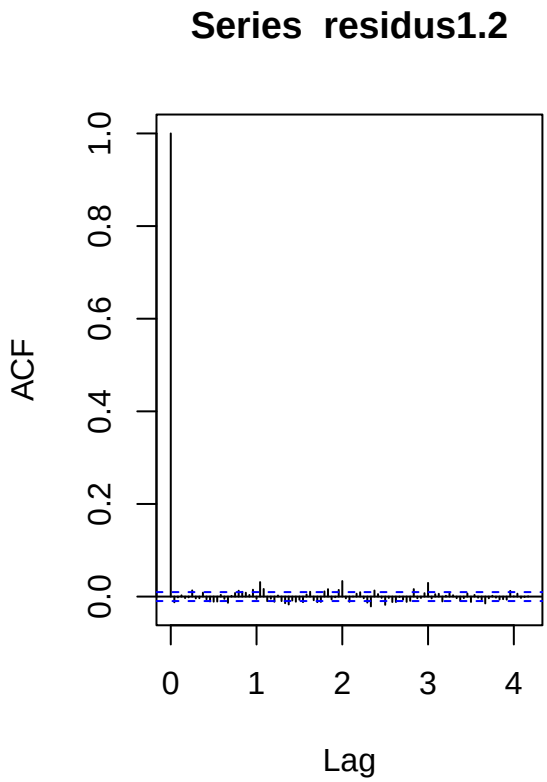
```
Call:
arima(x = log.no2.train.ts, order = c(ordre$p, 0, ordre$q), seasonal = list(order = c(ordre$P,
    1, ordre$Q), period = 24), method = "CSS-ML")

Coefficients:
```

```
         ar1     ar2      ar3     ma1      ma2     ma3     ma4      ma5    sar1
      0.6531  0.9120  -0.5819  0.0552  -0.8697  0.0455  0.0081  -0.0573  0.0757
s.e.  0.0379  0.0541   0.0321  0.0381   0.0369  0.0118  0.0103   0.0071  0.0058
         sma1
      -0.9617
s.e.   0.0022

sigma^2 estimated as 0.04453:  log likelihood = 5049.95,  aic = -10077.91
```

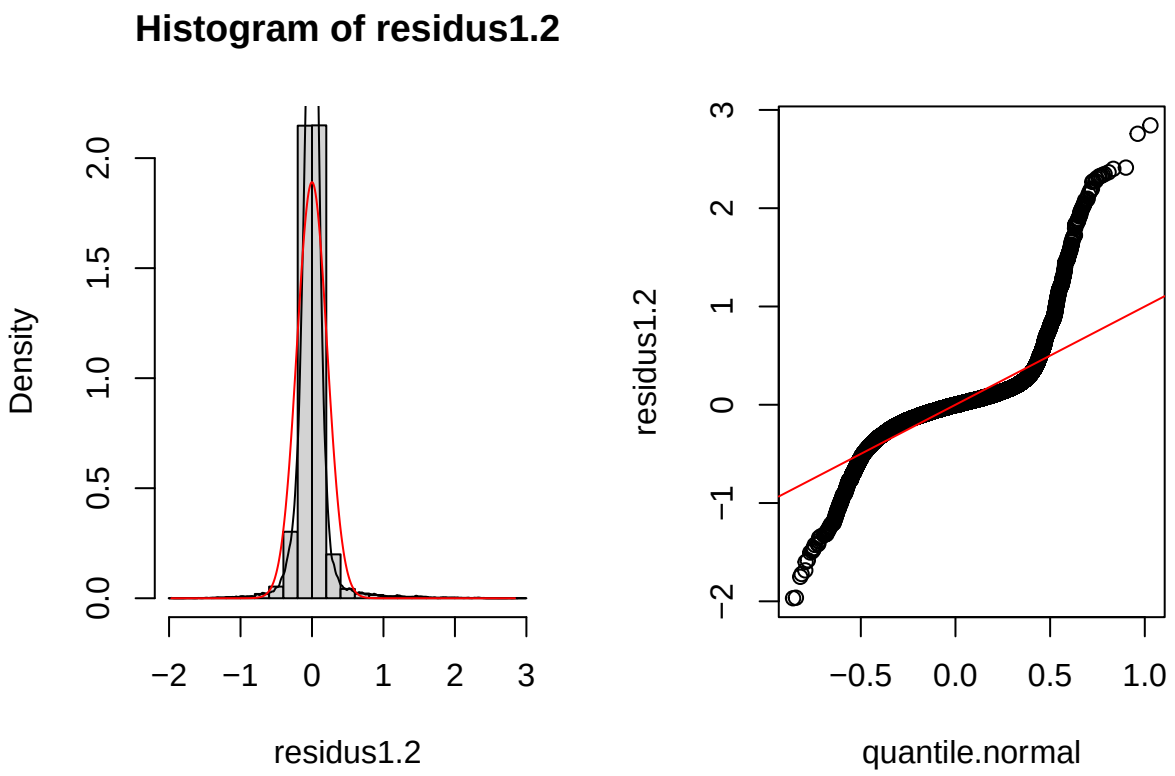
On regarde le comportement des reridus :

```
residus1.2 <- sarima_final2$residuals
par(mfrow=c(1,2))
pacf(residus1.2, na.action = na.pass, lag.max = 100)
acf(residus1.2, na.action = na.pass, lag.max = 100)
```





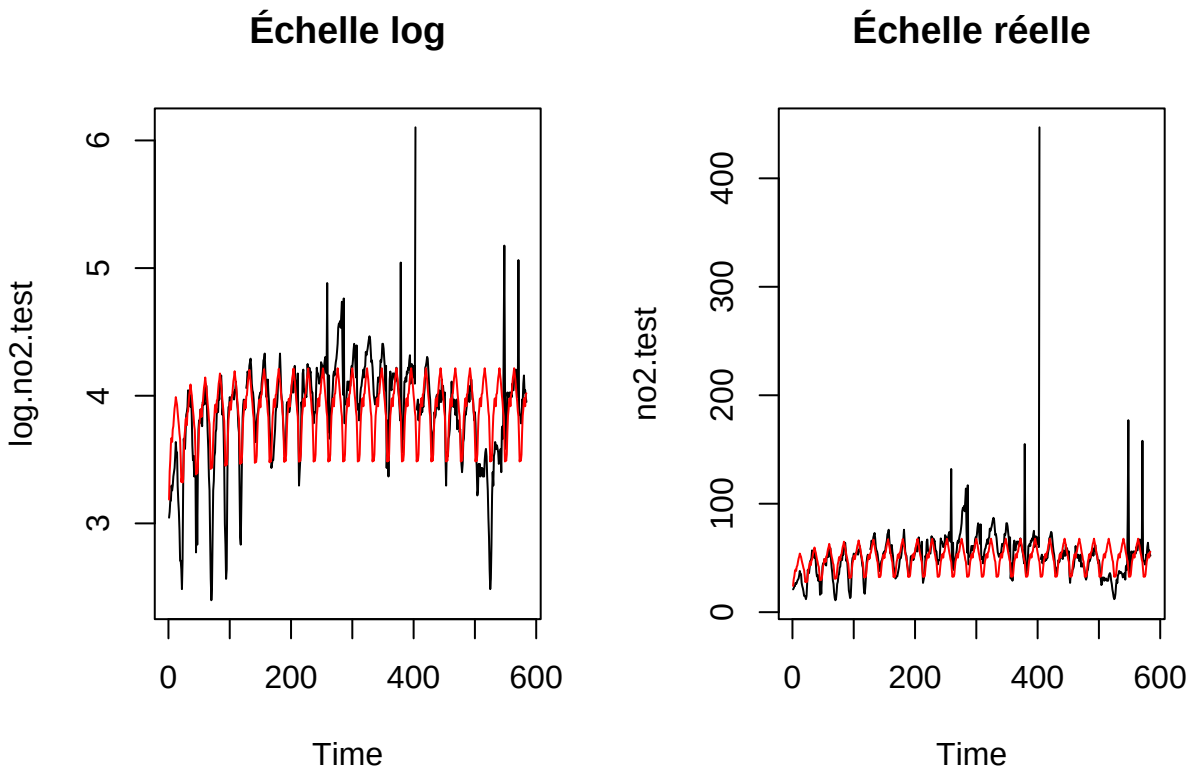

```
par(mfrow=c(1,2))
hist(residus1.2, probability = TRUE, breaks = 20)
lines(density(residus1.2, na.rm = TRUE))
t <- seq(min(residus1.2, na.rm = TRUE), max(residus1.2, na.rm = TRUE), by=0.01)
lines(t,dnorm(t, sd=sd(residus1.2, na.rm = TRUE)), col="red")

set.seed(912)
quantile.normal <- rnorm(length(residus1.2),sd=sd(residus1.2, na.rm = TRUE), mean = mean(residus1.2, na
qqplot(quantile.normal, residus1.2)
abline(a=0, b=1, col="red")
```



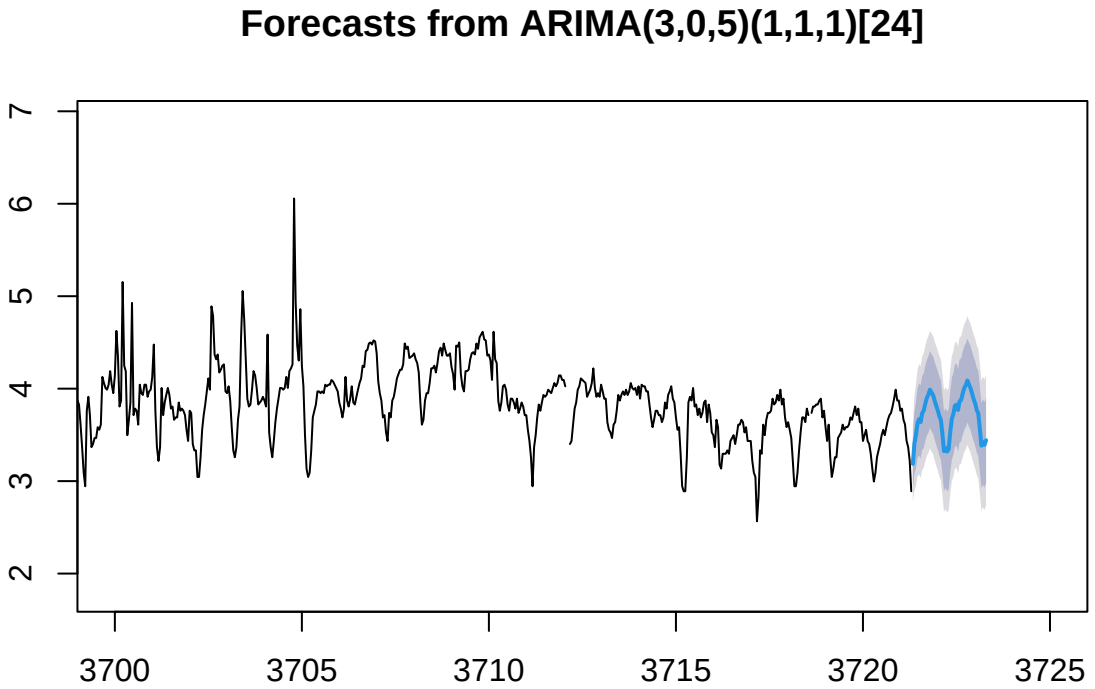
Les residus ne sont pas gaussiens (ce qui n'est pas grave car on n'a pas besoin de cette condition). Cependant, on pourrait aussi essayer de connaître la distribution de ces résidus estimés. *Exercice, essayer de trouver une distribution pour les residus : vous pouvez utiliser la commande* `fitdist` *du package* `fitdistplus`.

On regarde maintenant la qualité de prédiction :

```
par(mfrow=c(1,2))
plot.ts(log.no2.test, main="Échelle log")
log.pred2 <- predict(sarima_final2, n.ahead=length(log.no2.test))
lines(as.numeric(log.pred2$pred), col="red")

plot.ts(no2.test, main="Échelle réelle")
lines(exp(as.numeric(log.pred2$pred)), col="red")
```



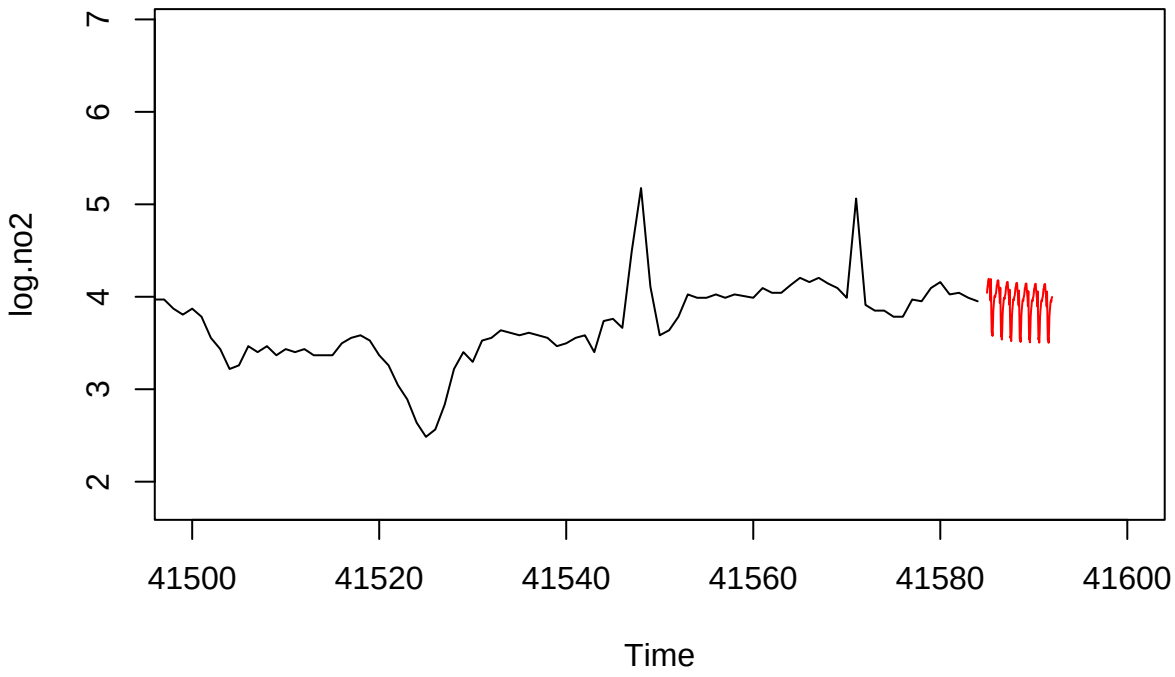

```
library(forecast)
plot(forecast(sarima_final2), xlim=c(3700, 3725))
```



2.2.- Réaliser une prédiction des mesures horaires de NO2 pour la première semaine de 2018.

```
RUN = FALSE
if (RUN==TRUE){
  sarima.update2 <- arima(log.no2.ts, order=c(ordre$p,0,ordre$q),
```

```
                        seasonal= list(order = c(ordre$P,1,ordre$Q), period = 24),
                        method = "CSS-ML" )
  save(sarima.update2, file = "sarima.update2_vf.RData")
  }else
  load("sarima.update2_vf.RData")
prediction.semaine1.2018 <- predict(sarima.update2, n.ahead=24*7)
hat.no2.semaine1.2018 <- ts(as.numeric(prediction.semaine1.2018$pred), start = 41585, frequency = 24)
plot.ts(log.no2, xlim=c(41500, 41600))
lines(hat.no2.semaine1.2018, col="red")
```



Bonus : Questions pratiques

B.1.- Répéter les points 1.2 et 2.1 en utilisant les données fournies après le 23 décembre 2021. Les modèles ont-ils changé ? Comparez ces deux périodes et posez-vous des questions sur l'évolution de NO2 dans cette station. Est-ce que le projet de *désenfumage et de renouvellement de l'air des espaces souterrains parisiens* (2021-2024) a donné des résultats significativement positifs à la station Auber ?

B.2.- On a constaté la présence de valeurs manquantes dans le jeu de données : (i) données manquantes isolées, (ii) données manquantes sur courte période et (iii) données manquantes sur longues périodes. Pour les cas (i) et (ii) on a réussi à estimer plusieurs paramètres à l'aide des options `omit.na` et '`na.action = na.pass`.

B.2.1.- Expliquer brièvement comment les estimateur empiriques de l'ACF, la moyenne, etc fonctionnnent sous la présence de données manquantes.

B.2.2.- Imaginons que la RATP a caché quelques données et qu'un organisme d'arbitrage vous demande d'imputer (i.e. compléter) les données manquantes. Proposer une démarche pour imputer les données manquantes. Pour cela, vous devez :

- Choisir une échantillon d'entraînement et de test
- Utiliser tous les modèles que vous considérez pertinent pour imputer ces données. Vous pouvez aussi utiliser les modèles du cours d'Apprentissage Supervisé (Quels seraient les modèles adaptés dans ce cadre ?).

- Comparer les modèles dans ce cadre en commentant leurs faiblesses et leur robustesse.

Exercice 2 : Prix du Bitcoin

On sait que le prix du Bitcoin a des comportements assez complexes. Cette complexité est encore plus marquée lorsque les registres sont donnés à des intervalles de temps plus courts, en particulier à cause de la volatilité du prix. Dans cet exercice, on va essayer de modéliser le prix de clôture (en $) en utilisant les modèles vus en cours et en allant plus loin avec des modèles que vous pourriez trouver dans la littérature et/ou en utilisant ChatGPT.

Le jeu de données se trouve sur la page moodle du cours sous le nom `BTC-USD.csv`. Vous pouvez charger les données via la commande suivante :

```
library(readr)
bitcoin <- read_csv("BTC-USD.csv")
```

Première partie : Analyse des prix moyens mensuels de clôture.

1.1.- Construire un tableau de données contenant les prix moyens mensuels de clôture. Dans ce tableau, inclure aussi une colonne contenant le logarithme des prix moyens mensuels de clôture.

1.2.- Appliquer la méthode de Box-Jenkins pour proposer un modèle (S)ARIMA pour la série *log des prix moyens mensuels de clôture*. Pourquoi on ne travaille pas directement avec *les prix moyens mensuels de clôture* ?

1.3.- Donner une prédiction du prix moyen du Bitcoin pour décembre 2023.

Seconde partie : Analyse des prix de clôture.

Dans cette partie vous avez la liberté d'utiliser le.s modèle.s (vus en cours, supports supplémentaires du cours, littérature, ChatGPT, etc) que vous voulez afin de modéliser le prix de clôture du Bitcoin. Ne vous limitez pas. Une fois validé votre modèle, réalisez une prédiction pour la semaine du 27 novembre et 1er décembre 2023. Au moment de ce TP, vous aurez déjà les prix et on pourra les comparer avec votre prédiction.