TP_SERIES_TEMPORELLES

Mamadou Diouf 20203258 Saâd Qriouet 20171683

03/28/2021

0. Chargement des librairies

```
library(ggfortify)
```

```
## Loading required package: ggplot2
```

```
library(astsa)
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Registered S3 methods overwritten by 'forecast':
##   method                 from
##   autoplot.Arima         ggfortify
##   autoplot.acf           ggfortify
##   autoplot.ar            ggfortify
##   autoplot.bats          ggfortify
##   autoplot.decomposed.ts ggfortify
##   autoplot.ets           ggfortify
##   autoplot.forecast      ggfortify
##   autoplot.stl           ggfortify
##   autoplot.ts            ggfortify
##   fitted.ar              ggfortify
##   fortify.ts             ggfortify
##   residuals.ar           ggfortify
```

```
##
## Attaching package: 'forecast'
```

```
## The following object is masked from 'package:astsa':
##
##     gas
```

```
library(tseries)
library(stats)
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(TTR)
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v tibble  3.0.4     v purrr   0.3.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x MASS::select()  masks dplyr::select()
```

1. Chargement du jeu de données et résumé

```r
data(USAccDeaths)
?USAccDeaths
summary(USAccDeaths)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6892    8089    8728    8789    9323   11317
```

Le jeu de données est une série temporelle donnant le nombre d'accidents mortels par mois aux Etats-Unis. Nous avons affiché le résumé statistiques de cette série à l'aide de la fonction "summary".

Nous allons regarder de plus près la série temporelle et y faire une analyse descriptive :

```r
print(USAccDeaths)
```

```
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 1973  9007  8106  8928  9137 10017 10826 11317 10744  9713  9938  9161  8927
## 1974  7750  6981  8038  8422  8714  9512 10120  9823  8743  9129  8710  8680
## 1975  8162  7306  8124  7870  9387  9556 10093  9620  8285  8466  8160  8034
## 1976  7717  7461  7767  7925  8623  8945 10078  9179  8037  8488  7874  8647
## 1977  7792  6957  7726  8106  8890  9299 10625  9302  8314  8850  8265  8796
## 1978  7836  6892  7791  8192  9115  9434 10484  9827  9110  9070  8633  9240
```

```r
str(USAccDeaths) # ts de 1973 jusqu'à 1978
```

```
##  Time-Series [1:72] from 1973 to 1979: 9007 8106 8928 9137 10017 ...
```

```r
class(USAccDeaths) #ts
```

```
## [1] "ts"
```

```r
start(USAccDeaths) # 01/1973
```

```
## [1] 1973    1
```

```r
end(USAccDeaths) # 12/1978
```

```
## [1] 1978   12
```

```r
frequency(USAccDeaths) #12 => 12 obs par an (donc une obs par mois)
```

```
## [1] 12
```

```r
deltat(USAccDeaths) #1/12
```

```
## [1] 0.08333333
```

```r
cycle(USAccDeaths) # ce sont bien des données mensuelles
```

```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 1973   1   2   3   4   5   6   7   8   9  10  11  12
## 1974   1   2   3   4   5   6   7   8   9  10  11  12
## 1975   1   2   3   4   5   6   7   8   9  10  11  12
## 1976   1   2   3   4   5   6   7   8   9  10  11  12
## 1977   1   2   3   4   5   6   7   8   9  10  11  12
## 1978   1   2   3   4   5   6   7   8   9  10  11  12
```

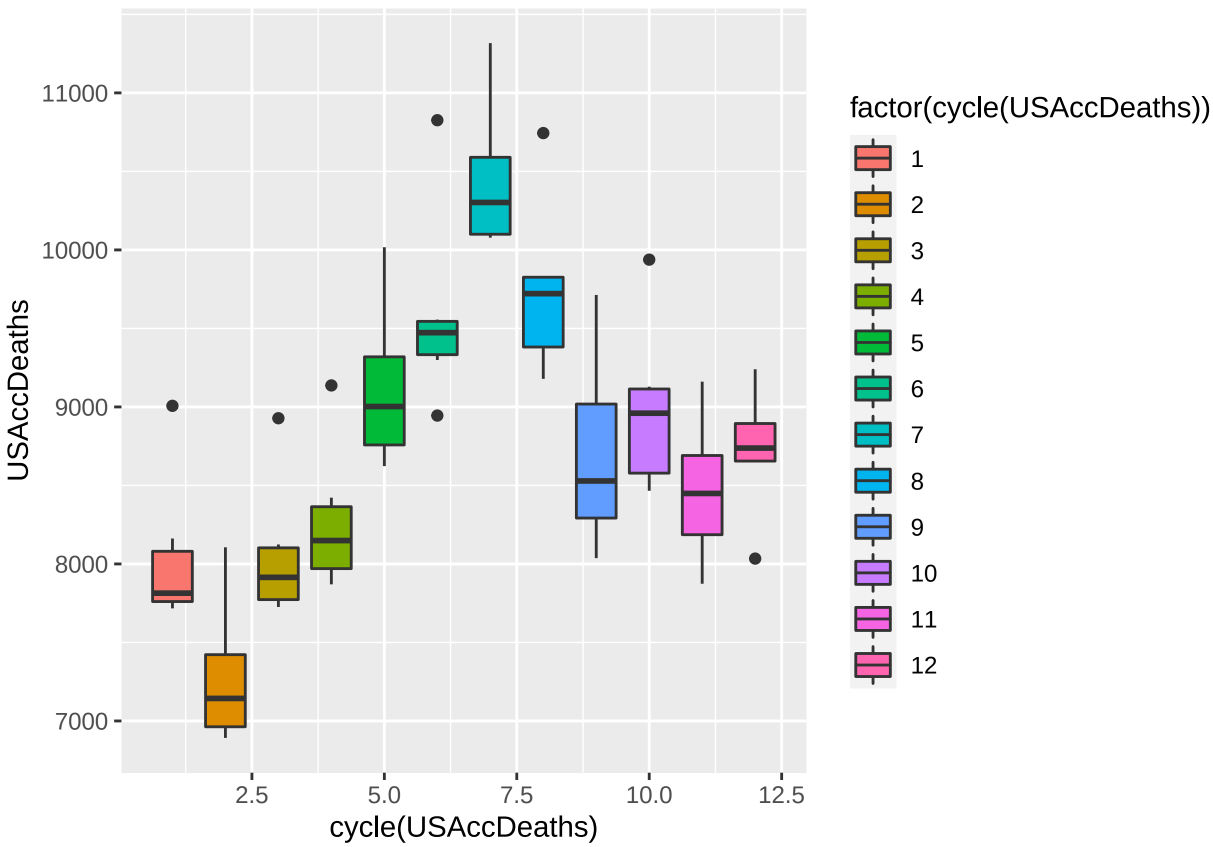
```r
mean(USAccDeaths) # Environ 8789 deces en moyenne par mois
```

```
## [1] 8788.792
```

Il s'agit d'une série temporelle allant de Janvier 1973 jusqu'à Décembre 1978. On observe grâce à la fonction frequency qu'il y à 12 observations par unité de temps, donc 12 par ans soit une observation par mois, chôse que l'on vérifie grâce a la fonction deltat qui nous donne une période d'environ 0,0833.. entre deux observations successives (ce qui correspond à un mois) et grâce à la fonction cycle qui nous permet de bien voir si ce sont des données mensuelles. Nous remarquons également qu'il y à en moyenne environ 8789 décès par mois.

```
# boxplot
ggplot(data=USAccDeaths, aes(cycle(USAccDeaths),USAccDeaths)) + geom_boxplot(aes(fill = factor(cycle(US
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



Nous observons qu'en moyenne, le mois contenant le nombre d'accidents le plus faible est Février avec environ 7200 décès; tandis que le plus élevé est Juillet avec presque 10500 décès (cela est probablement du à l'afflux de trafic routier durant la periode estivale avec les vacances). Nous remarquons que la série est croissante entre Fevrier et Juillet, et ensuite décroit jusqu'à Septembre avant de se stabiliser (avec une légère décroissance tout de même) jusqu'en Janvier et baisser d'environ un dixième en Février.

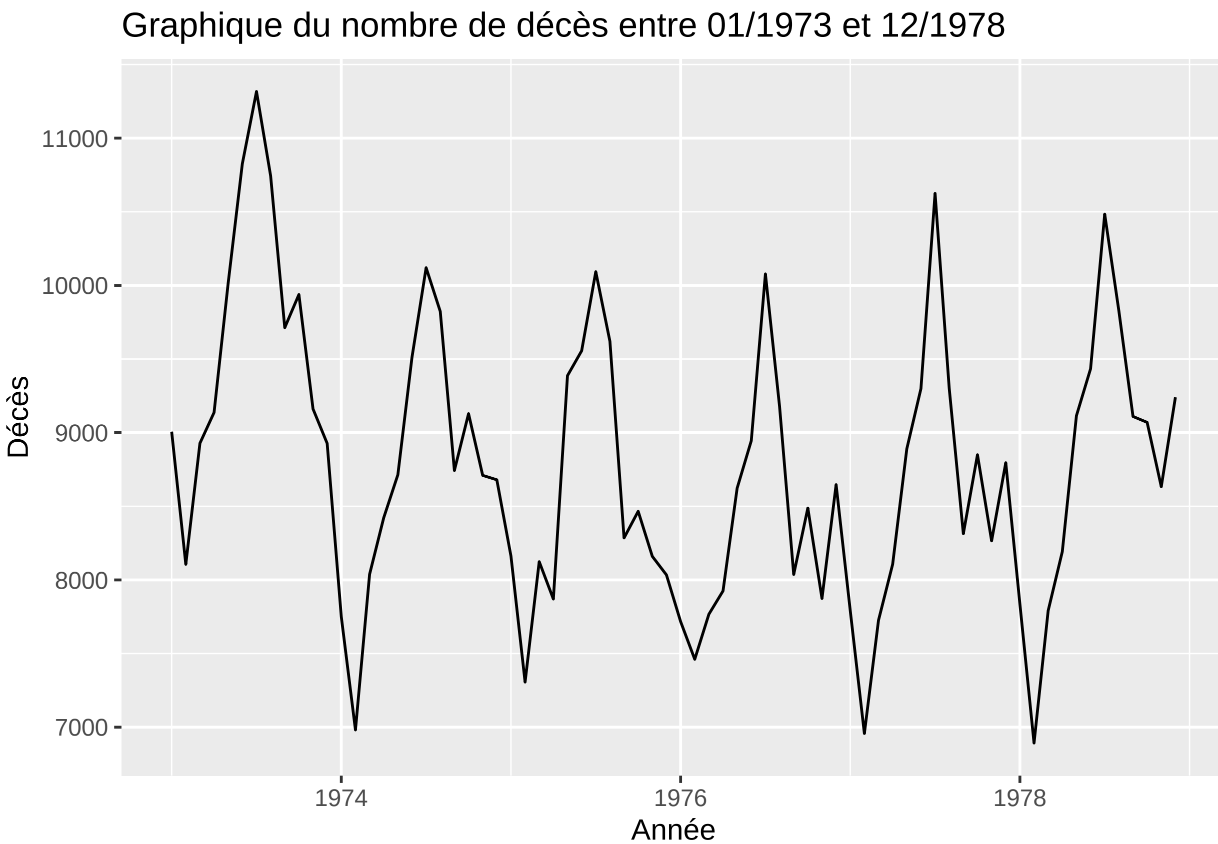
2. Visualisation des données

Nous allons nous intéresser à la représentation graphique de la série afin de réaliser d'éventuelles transformations.

a) Chronogramme :

Afin d'étudier une série temporelle, nous devons commencer par son chronogramme. En effet, ce dernier nous montre certains aspects de la série comme des valeurs anormales, des ruptures, changements de dynamique de la série,. . .

```
## Plot de la série :
autoplot(USAccDeaths, xlab='Année',ylab='Décès',main='Graphique du nombre de décès entre 01/1973 et 12/
```



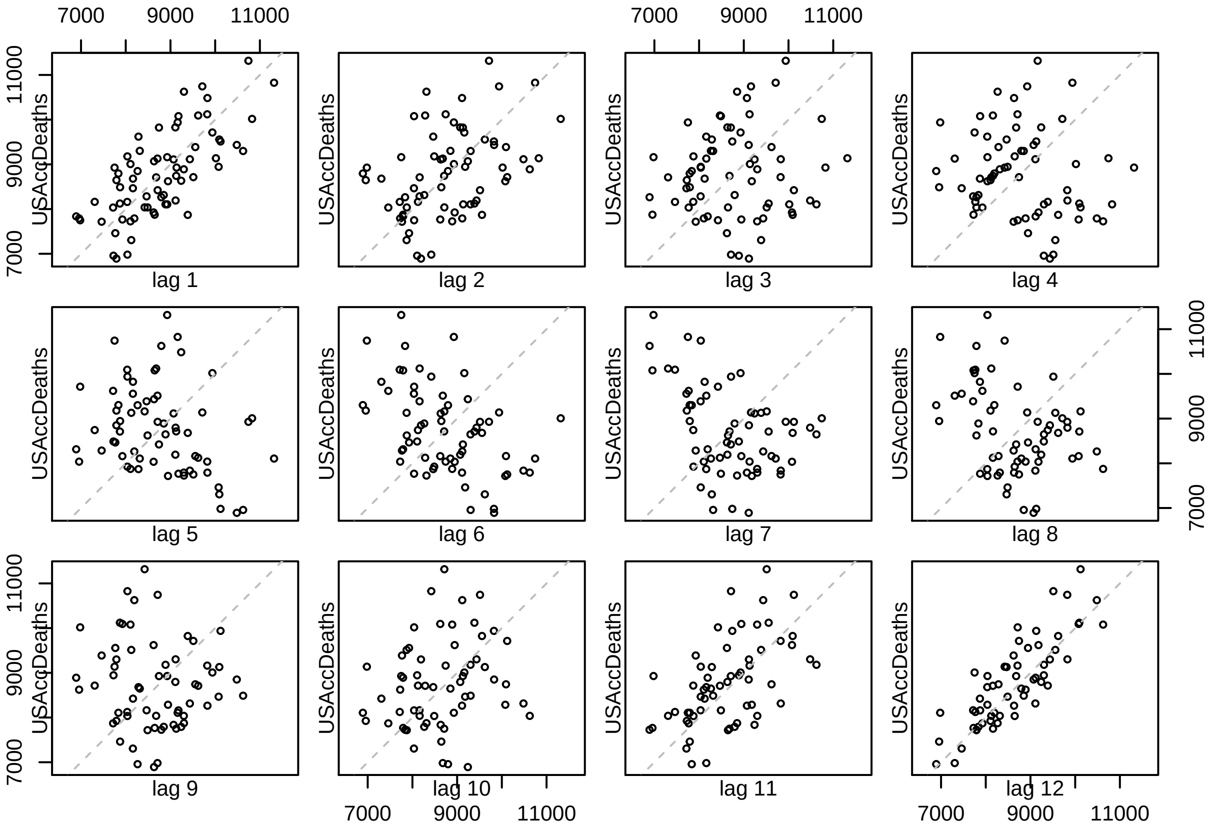
Comme nous l'avons remarqués avec le boxplot plus haut, nous observons une composante saisonnière dans cette série assez élevées au début et à la fin de la série.De plus, on observe une tendance qui décroit jusqu'à 1976 avant de croître (c'est d'ailleurs presque linéaire). La série n'est donc pas stationnaire.

Afin de vérifier nos hypothèses nous allons regarder d'autres graphiques complémentaires au chronogramme, comme le lag plot et le month plot de cette série.

b) Lag plot :

Le lag plot d'une série est un diagramme de dispersion des points ayant pour abscisse la série retardée de k instants et pour ordonnée la série non retardée. Le lag plot nous permet de comprendre la dépendance de la série par rapport à son passé.

```
lag.plot(USAccDeaths,lags=12,layout=c(3,4),do.lines=FALSE)
```

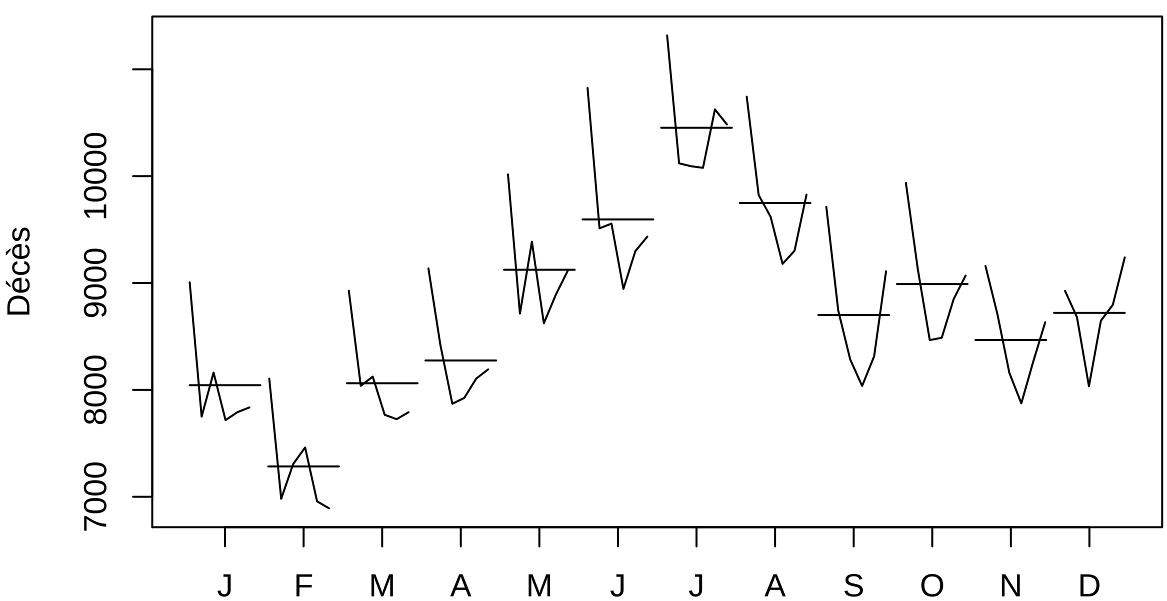


Ici, nous observons des autocorrélations très visibles jusqu'au retard 12.

c) Month plot :

Le month plot est une représentation simultanée des chronogrammes des séries associée à chaque mois, elle nous permet par exemple de vérifier s'il y à absence ou présence d'effet saisonnier.

```
monthplot(USAccDeaths, ylab = "Décès", main="", cex.main=1)
```

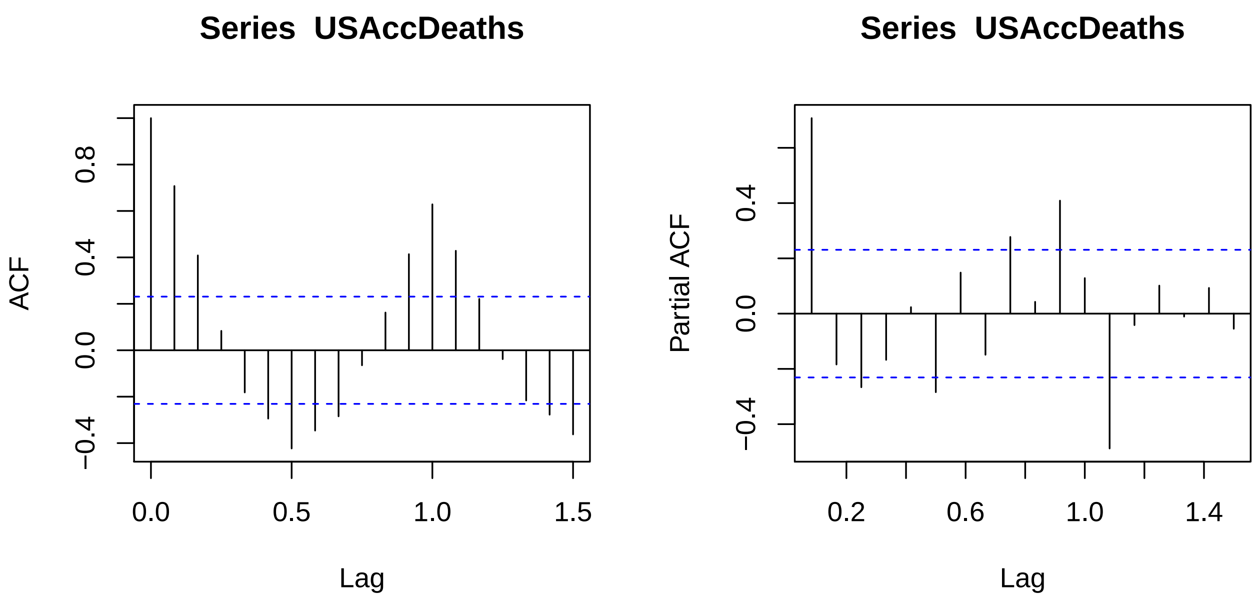


Ici, nous remarquons très clairement une variation saisonnière de la série, et des différences .

d) ACF et PACF

Les graphiques d'ACF et PACF nous permettent de determiner le degré du modèle ARMA associé.

```
layout(matrix(1:2, nrow = 1, ncol = 2))
acf(USAccDeaths)
pacf(USAccDeaths)
```



Avec les graphiques de l'ACF (à gauche) et de la PACF (à droite), nous remarquons une tendance (avec une décroissance forte) et une saisonnalité (à travers une périodicité).

Les graphiques confirment donc la tendance qui croît et décroît, ainsi que la présence d'une périodicité de l'ordre de 12.

Nous avons vu que cette série n'est pas stationnaire et nécessite des transformations.

3. Premières transformations pour le rendre proche d'une série stationnaire et mettre en oeuvre des outils permettant de valider ou non la stationnarité de la série Après avoir expliqué ce que fait "decompose" de R, reprogrammer les différentes étapes de "decompose" et les comparer aux résultats donnés par "decompose".

Comme dit plus haut, nous allons réaliser des transformations afin de rendre notre série temporelle proche d'une série stationnaire, ensuite nous allons réaliser le test de Dickey-Fuller afin de vérifier que l'on à bien le résultat attendu.

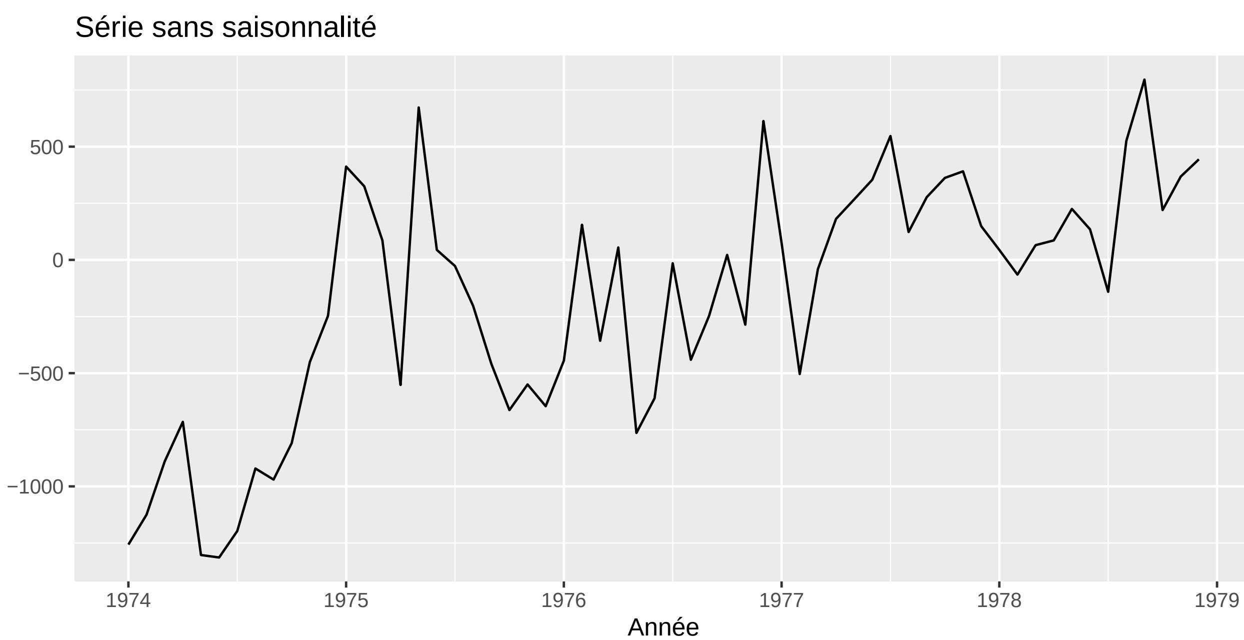
Dans cette partie, nous allons donc tenter de valider la stationnarité de cette série en réalisant au préalable des transformations, et dans un second temps, nous interesser à la fonction "decompose"

a) Transformations

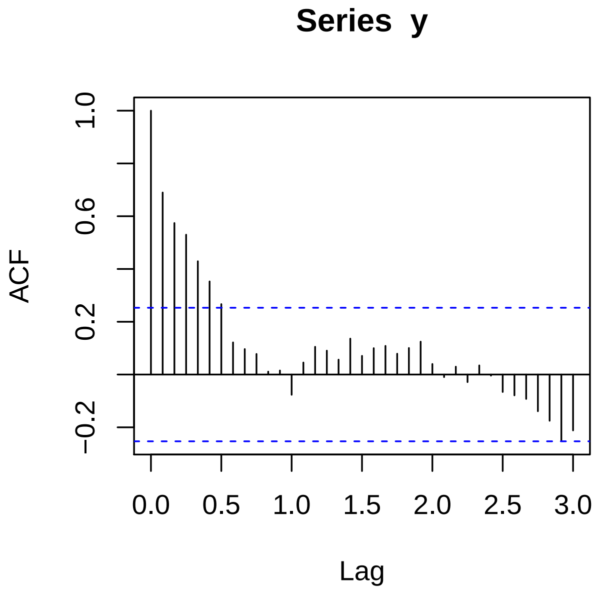
```
y=diff(USAccDeaths,lag=12,differences=1)
autoplot(y, xlab='Année',ylab='',main="Série sans saisonnalité", plot=F)
```

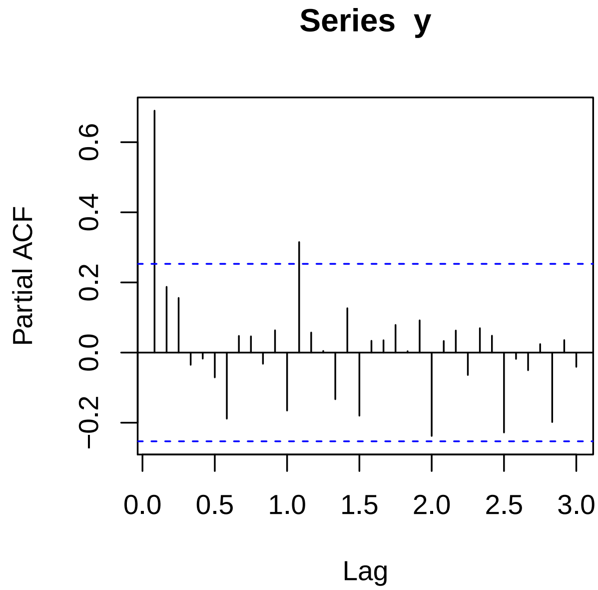
i) Retrait de la saisonnalité

```
## Warning: Ignoring unknown parameters: plot
```




```
layout(matrix(1:2, nrow = 1, ncol = 2))
acf(y,lag.max=36)
pacfy=pacf(y,lag.max=36)
```



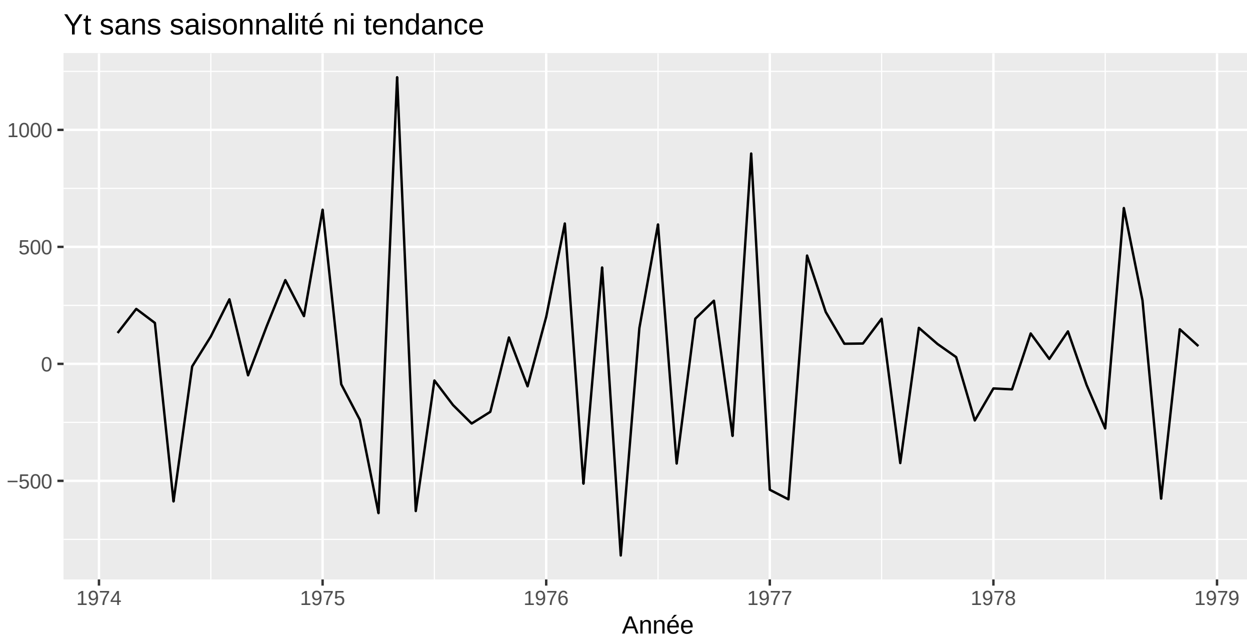


Après avoir retiré la saisonnalité, nous allons aussi y retirer la tendance

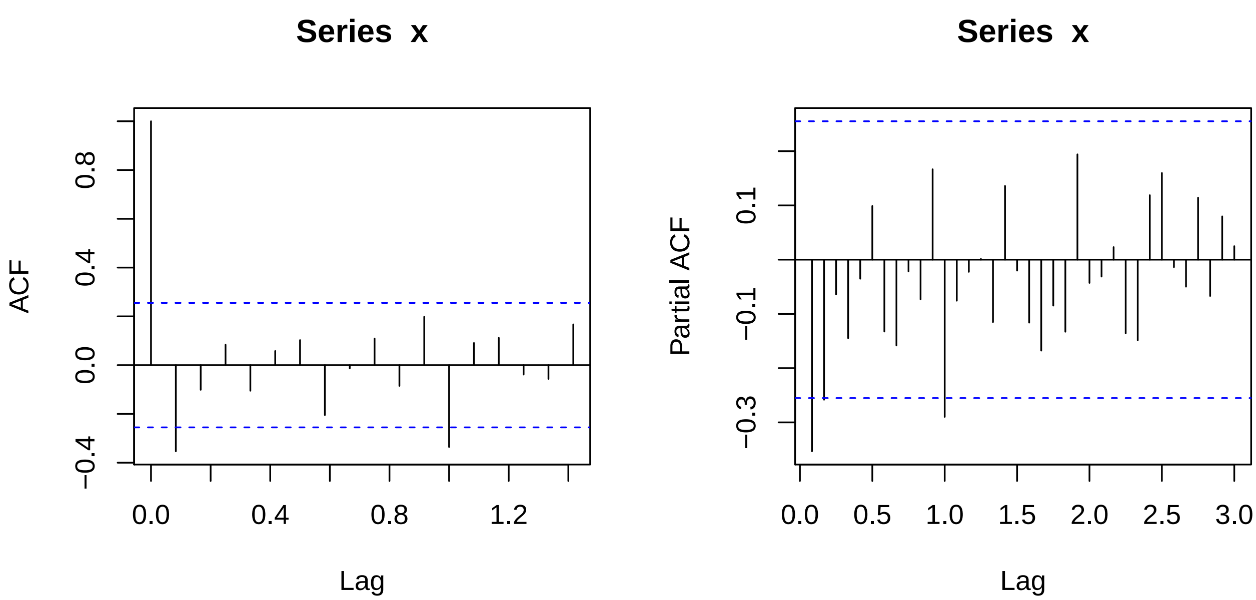
```r
x=diff(y,lag=1,differences=1)
autoplot(x, xlab='Année',ylab='',main="Yt sans saisonnalité ni tendance",plot=F)
```

ii) Retrait de la tendance

```
## Warning: Ignoring unknown parameters: plot
```




```r
layout(matrix(1:2, nrow = 1, ncol = 2))
acf(x)
pacfx=pacf(x,lag.max=36)
```



b) Test de la stationnarité :

Utilisons le test de Dickey-Fuller afin de voir si notre série est stationnaire. Dans ce test, nous allons tester l'hypothèse nulle H0 : "La série n'est pas stationnaire" contre l'hyopthèse alternative H1 : "La série est stationnaire". Le test sera réalisé à chacune des series (celle de base puis avec les deux étapes de transformation)

```
adf.test(USAccDeaths,alternative=c("stationary"),12)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  USAccDeaths
## Dickey-Fuller = -1.6398, Lag order = 12, p-value = 0.722
## alternative hypothesis: stationary
```

```
adf.test(y,alternative=c("stationary"),12)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  y
## Dickey-Fuller = -3.0751, Lag order = 12, p-value = 0.1402
## alternative hypothesis: stationary
```

```
adf.test(x,alternative=c("stationary"),12)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  x
## Dickey-Fuller = -3.4595, Lag order = 12, p-value = 0.05513
## alternative hypothesis: stationary
```

Ici, nous obtenons une p-value de 0.05513 qui est presque égale à 5%, nous considérons le test significatif. Ainsi nous rejetons donc l'hypothèse H0 à 5%, ie : la série est stationnaire.

Nous avons vus grâce à ce test que notre série est bien stationnaire, nous allons donc pouvoir utiliser un modèle ARIMA.

c) Decompose

La fonction "decompose" réduit une série temporelle en 3 composantes : **tendance, effets saisonniers et erreurs aléatoires.**

Nous allons l'implémenter et la comparer avec celle déjà présente sur R en utilisant notre série temporelle X_t,
$X_t = M_t + S_t + E_t$
M_t est la tendance, S_t est l'effet saison et $E_t$est une erreur aléatoire(résidus)

Donc, en estimant et en soustrayant les deux M_t et S_t De X_t, nous espérons avoir une série temporelle de résidus stationnaires E_t.

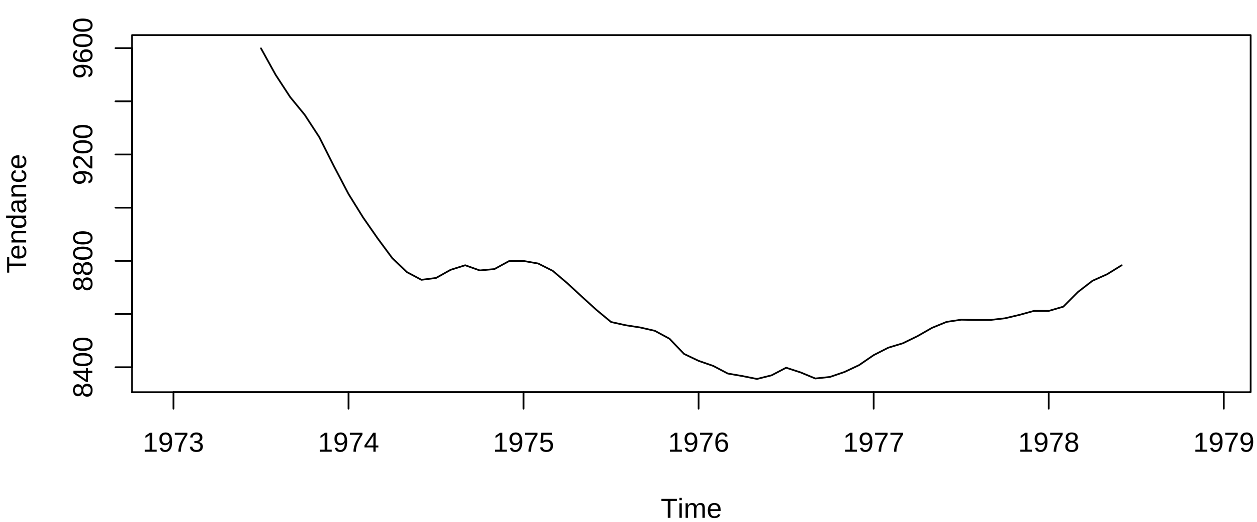
Estimation de la tendance: nous l'estimons avec la moyenne mobile:
$\hat{m}_t = \sum_{k=-a}^{a} \frac{1}{1+2a} X_{t+k}$

Nous avons des données mensuelles pour notre série alors nous choisissons une moyenne mobile de 12 points,$a = 6$.

```
decusd=decompose(USAccDeaths,type = 'additive')
filtre <- c(1/2, rep(1, times = 11), 1/2)/12
Mchap <- stats::filter(USAccDeaths,filtre)
Mchap
```

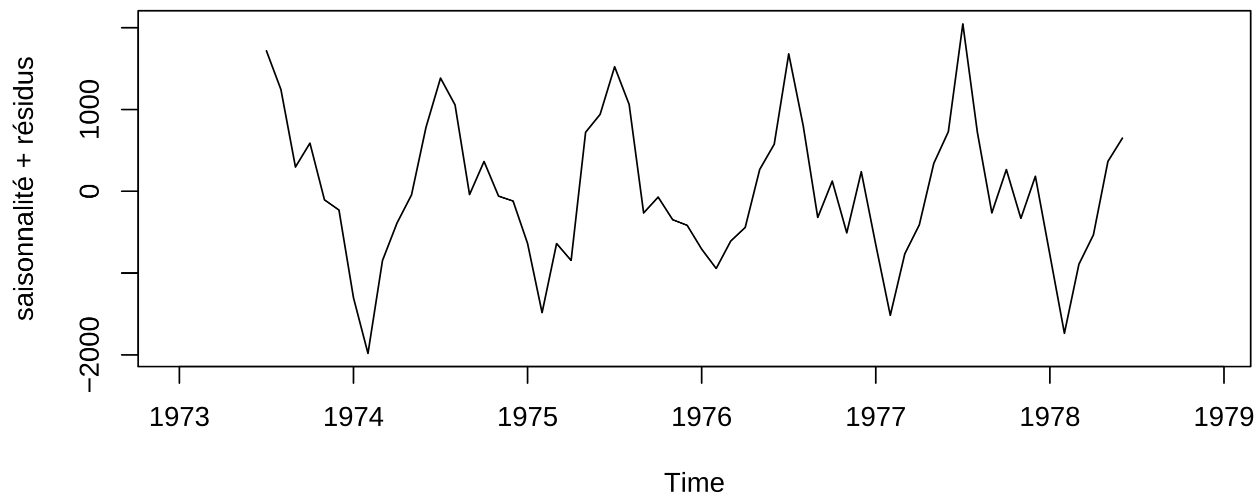
```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1973       NA       NA       NA       NA       NA       NA 9599.375 9500.125
## 1974 9051.542 8963.292 8884.500 8810.375 8757.875 8728.792 8735.667 8766.375
## 1975 8799.708 8790.125 8762.583 8715.875 8665.333 8615.500 8570.042 8557.958
## 1976 8424.042 8405.042 8376.333 8366.917 8355.917 8369.542 8398.208 8380.333
## 1977 8445.542 8473.458 8490.125 8516.750 8548.125 8570.625 8578.667 8577.792
## 1978 8611.792 8627.792 8682.833 8725.167 8749.667 8783.500       NA       NA
##           Sep      Oct      Nov      Dec
## 1973 9416.167 9349.292 9265.208 9156.167
## 1974 8783.500 8764.083 8769.125 8799.000
## 1975 8549.542 8536.958 8507.417 8450.125
## 1976 8357.625 8363.458 8382.125 8408.000
## 1977 8577.792 8584.083 8597.042 8612.042
## 1978       NA       NA       NA       NA
```

```
plot.ts(Mchap, ylab = "Tendance", cex = 1)
```



Estimation de la saisonnalité

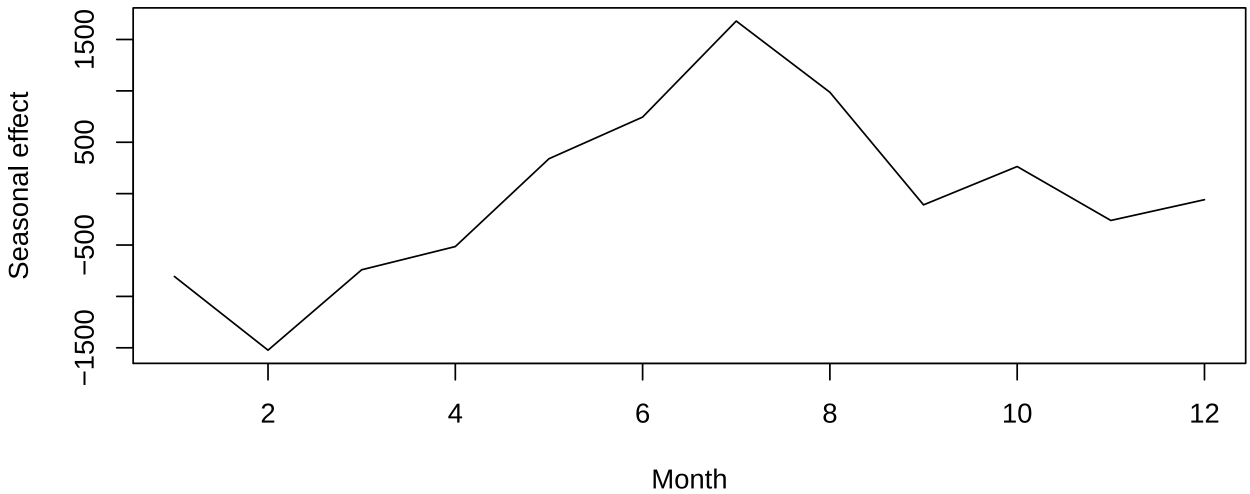
```r
D <- USAccDeaths - Mchap  #saisonnalité + résidus
plot.ts(D,ylab = "saisonnalité + résidus", cex = 1)
```



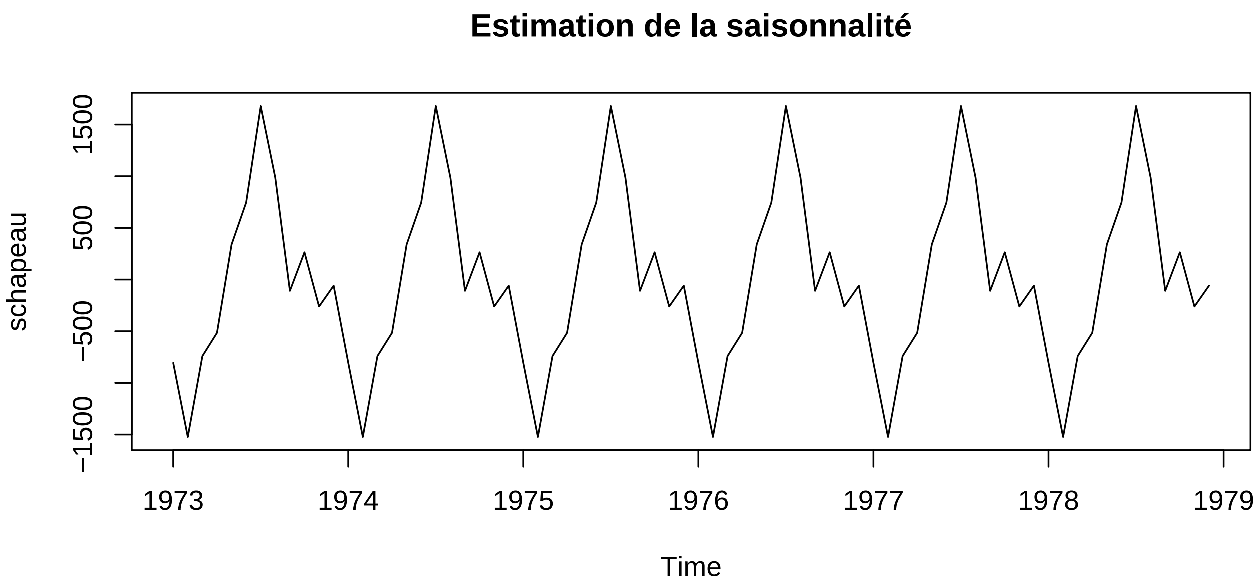

```r
## longueur de la ts
ll <- length(D)
## frequency (ie, 12)
ff <- frequency(D)
## nombre de périodes (years); %/% is integer division
periods <- ll%/%ff
periods
```

```
## [1] 6
```

```r
## index of cumulative month
index <- seq(1, ll, by = ff) - 1
## get mean by month
stild <- numeric(ff)
for (i in 1:ff) {
    stild[i] <- mean(D[index + i], na.rm = TRUE)
}
## subtract mean to make overall mean = 0
schape <- stild - mean(stild)
## plot the monthly seasonal effects
plot.ts(schape, ylab = "Seasonal effect", xlab = "Month", cex = 1)
```

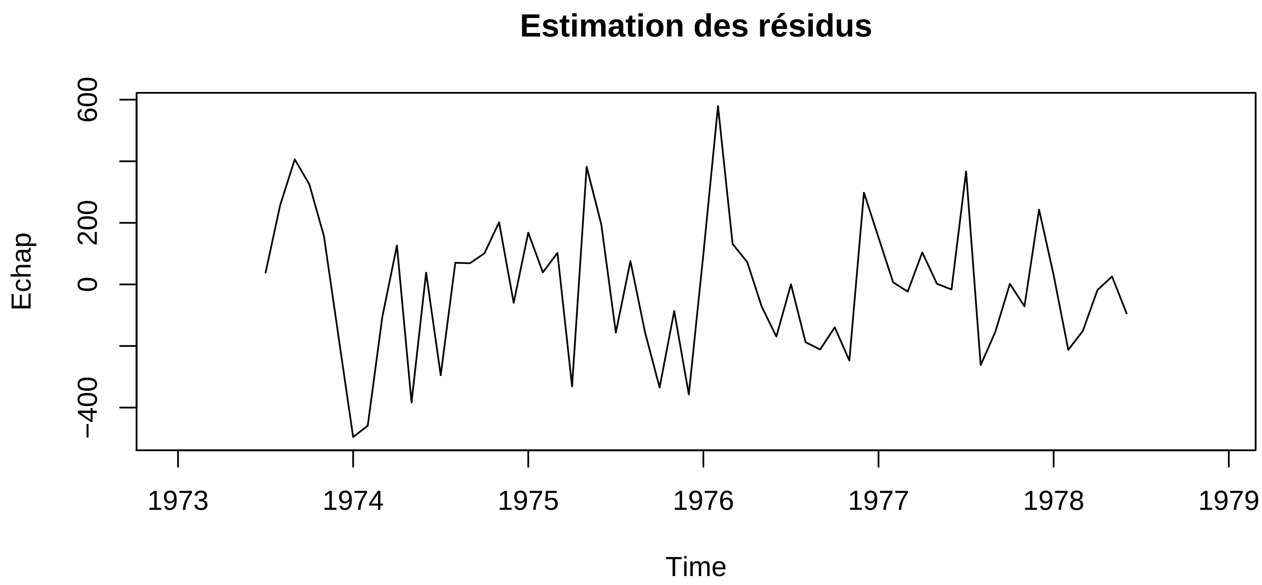



```
schapeau <- ts(rep(schape, periods + 1)[seq(ll)], start = start(D),
    frequency = ff)
plot.ts(schapeau, main="Estimation de la saisonnalité")
```

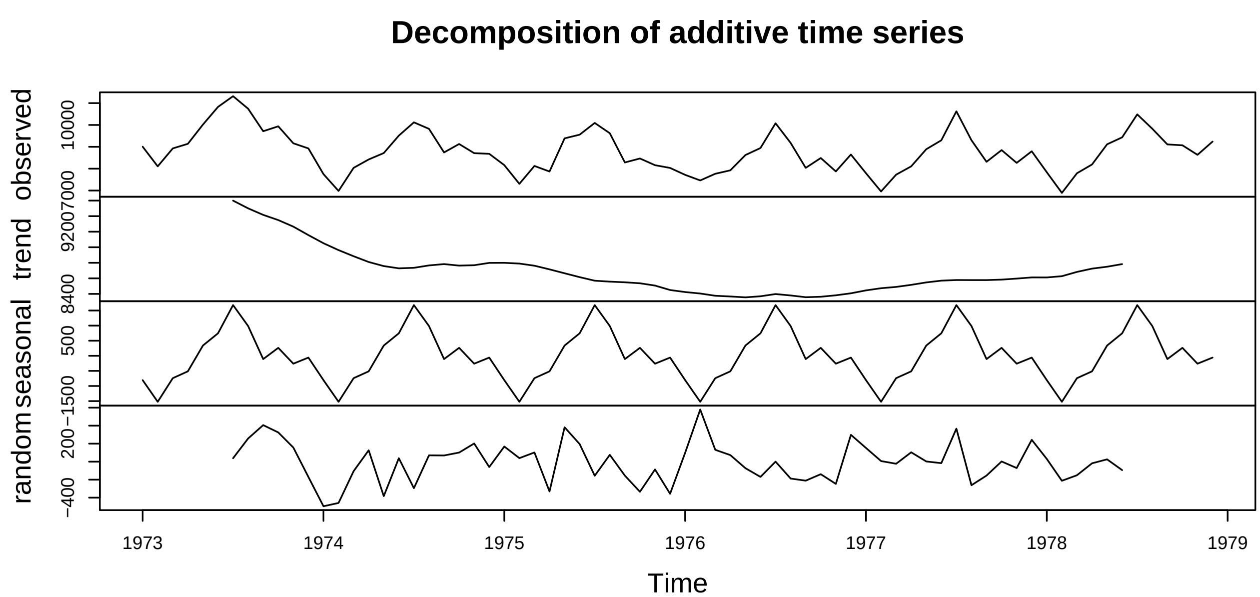


Estimation des résidus

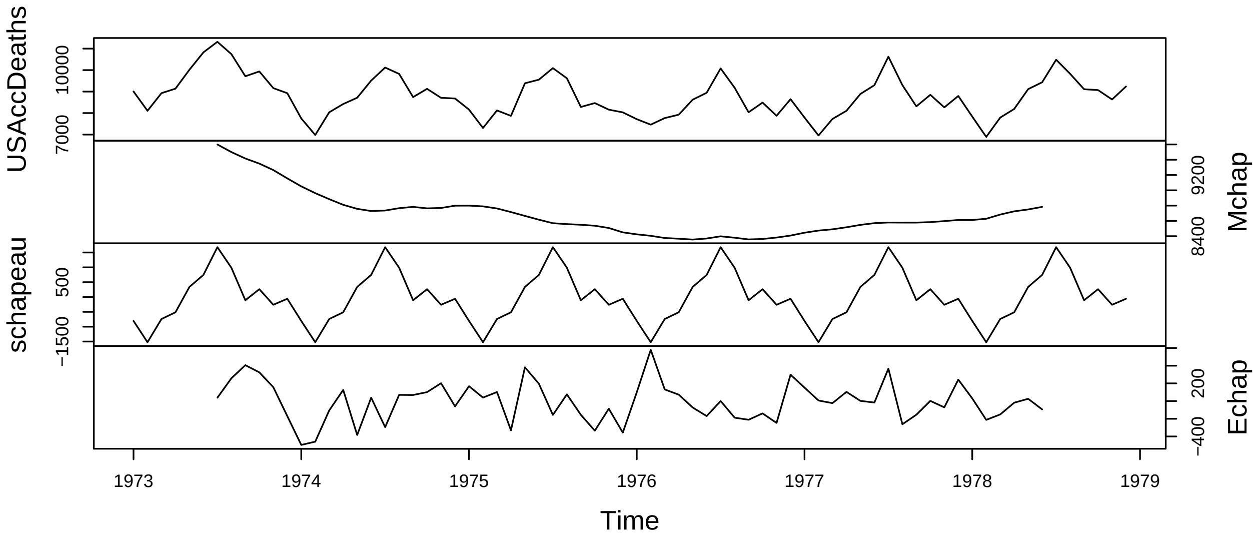
```
Echap = USAccDeaths - Mchap - schapeau
plot.ts(Echap,main="Estimation des résidus")
```




```r
par(mfrow = c(2,1))
plot(decusd)
```




```r
plot(cbind(USAccDeaths, Mchap, schapeau, Echap), main = "",
    yax.flip = TRUE)
```



Ainsi, les resultats donnés par decompose et celle que l'on a implémenté sont identiques.

4. Proposer 4 méthodes descriptives ou non permettant de prédire la série en la comparant avec la fin de la série. Présenter les différentes méthodes de prédiction.

Tout d'abord, nous allons séparer la série temporelle en deux avec une partie pour entraîner les modèles et une pour prédire la série.

```
USA_train <- window(USAccDeaths,end=c(1977,12))
USA_test <- window(USAccDeaths,start=c(1978,1))
x <- as.vector(time(USAccDeaths))
y <- as.vector(USAccDeaths)
```

Après avoir subdivisé notre série en deux séries (une d'entraînement et une de prédiction), nous allons utiliser différentes méthodes afin de prédire la série. Nous allons utiliser un modèle trigonométrique, un modèle ARIMA ainsi que deux méthodes de lissage exponentiel que sont la méthode de lissage exponentiel simple et la méthode de Holt-Winters

a) Modèle trigonométrique

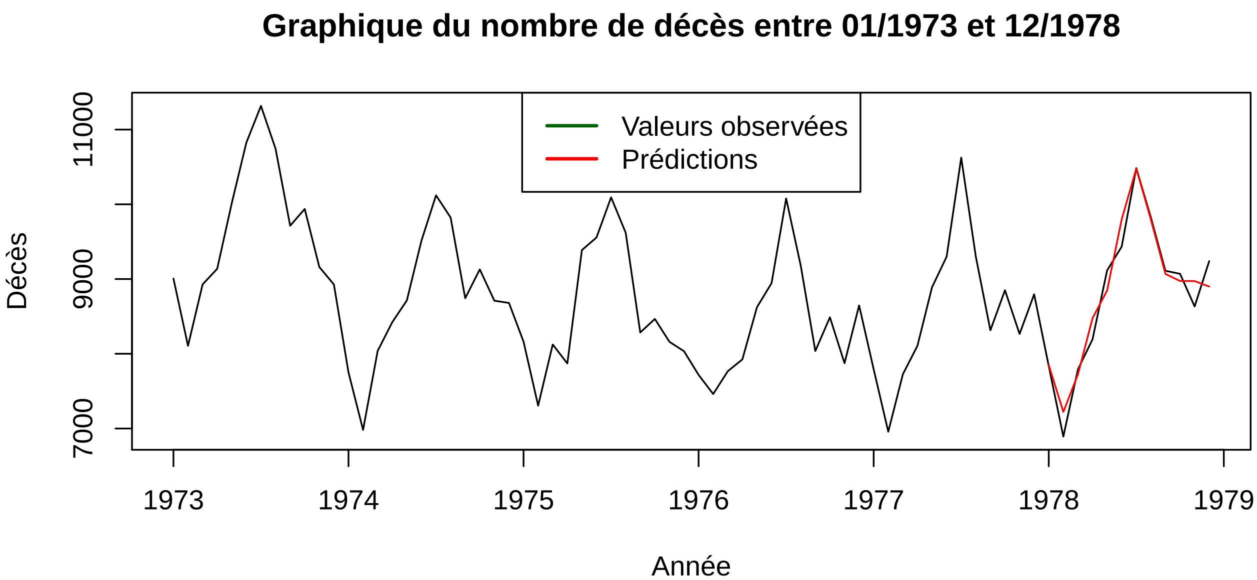
Nous allons utiliser un modèle polynomial et trigonométrique afin de prédire la série.

```
T <- seq(1973,1977+11/12,by=1/12)
times=seq(1978,1978+11/12,by=1/12)  #temps prédits

modTRIGO=lm(y~x+I(x^2)+I(x^3)+sin(2*pi*x)+cos(2*pi*x)+sin(4*pi*x)+cos(4*pi*x)+sin(6*pi*x)+cos(6*pi*x)+s
predTRIGO<-predict(modTRIGO,data.frame(x=times))
```

```
## Warning in predict.lm(modTRIGO, data.frame(x = times)): prediction from a rank-
## deficient fit may be misleading
```

```
plot(USAccDeaths, main="Graphique du nombre de décès entre 01/1973 et 12/1978",
    xlab="Année",
    ylab="Décès")
lines(times,predTRIGO,col="red")
legend('top',c("Valeurs observées","Prédictions"),col=c("darkgreen","red"),lty=rep(1,2),lwd = rep(2,2))
```



Grâce au graphique, nous observons que le modèla à plutôt bien prédit la série, cependant elle présente une erreur de prédiction à la fin où elle n'arrive pas à prévoir une chut dans le nombre de décès accidentels fin 1978.

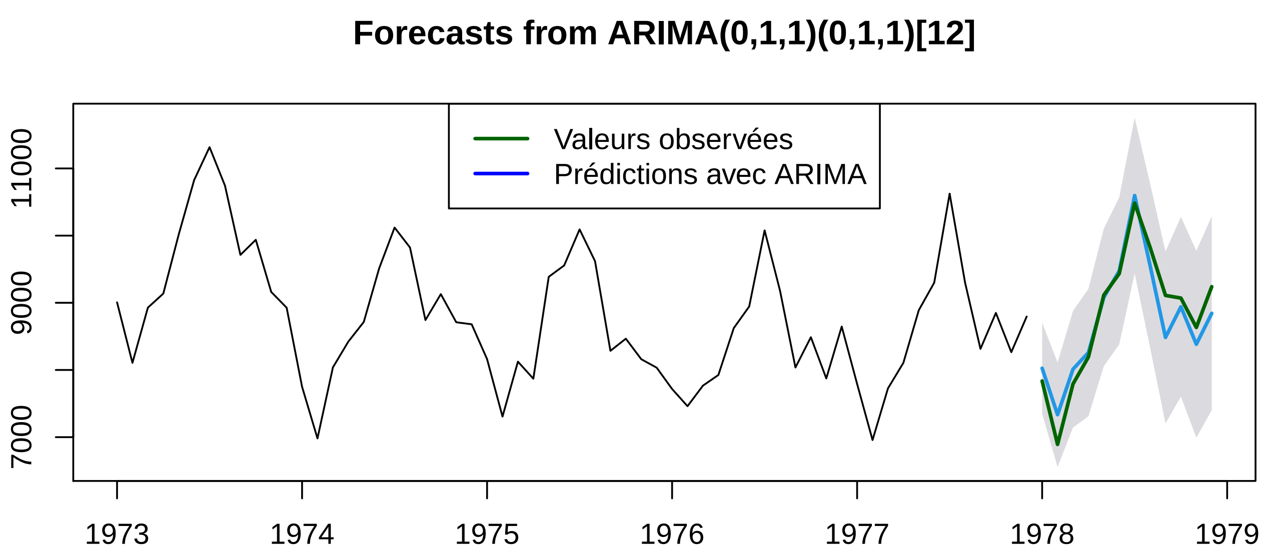
b) Modèle ARIMA

Nous allons determiner le meilleur modèle ARIMA à l'aide de la fonction "auto.arima". Cette fonction cherche le meilleur modèle suivant un des 3 critère d'information : AIC, AICc ou BIC.

```r
modAUTO = auto.arima(USA_train)
summary(modAUTO)
```

```
## Series: USA_train
## ARIMA(0,1,1)(0,1,1)[12]
##
## Coefficients:
##           ma1     sma1
##       -0.4316  -0.4506
## s.e.   0.1366   0.1823
##
## sigma^2 estimated as 119029:  log likelihood=-341.77
## AIC=689.54   AICc=690.1   BIC=695.09
##
## Training set error measures:
##                    ME    RMSE      MAE       MPE    MAPE     MASE        ACF1
## Training set 57.39284 298.784 207.6925 0.6697495 2.43218 0.431177 -0.03157018
```

```r
predAUTO <- forecast(modAUTO, level = 0.95, h=1*12)
plot(predAUTO)
points(USA_test,type='l',col='darkgreen',lwd=2)
legend('top',c("Valeurs observées","Prédictions avec ARIMA"),col=c("darkgreen","blue"),lty=rep(1,2),lwd
```



Le meilleur modèle determiné par cette méthode est le modèle ARIMA(0,1,1)(0,1,1)[12], il possède un AIC de 689,54. Grâce au graphique, nous observons que le modèle est très efficace car il prédit très bien la série réelle.

Les deux prochaines méthodes que nous allons utiliser pour prédire la série temporelle font parties de la famille des méthodes dites de "lissage exponentiel". Ces méthodes permettent de prédire une série temporelle sans prendre en compte des variables indépendantes. Leurs prévisions sont des moyennes pondérées d'observations du passé de sorte que les poids décroissent de manière exponentielle au fur et à mesure que les observations vieillissent, ie : plus l'observation est récente plus elle sera "importante".

Il existe plusieurs méthodes :

- Lissage exponentiel simple : les prédictions toutes ont la même valeur, ie : c'est une moyenne pondérée des données passées, avec des poids décroissant vers les valeurs passées
- Lissage exponentiel double : on y ajoute une tendance linéaire à la prédiction.
- Holt-Winters : semblable au lissage exponentiel double mais on y ajoute un paramètre afin de rendre cette méthode plus flexible.

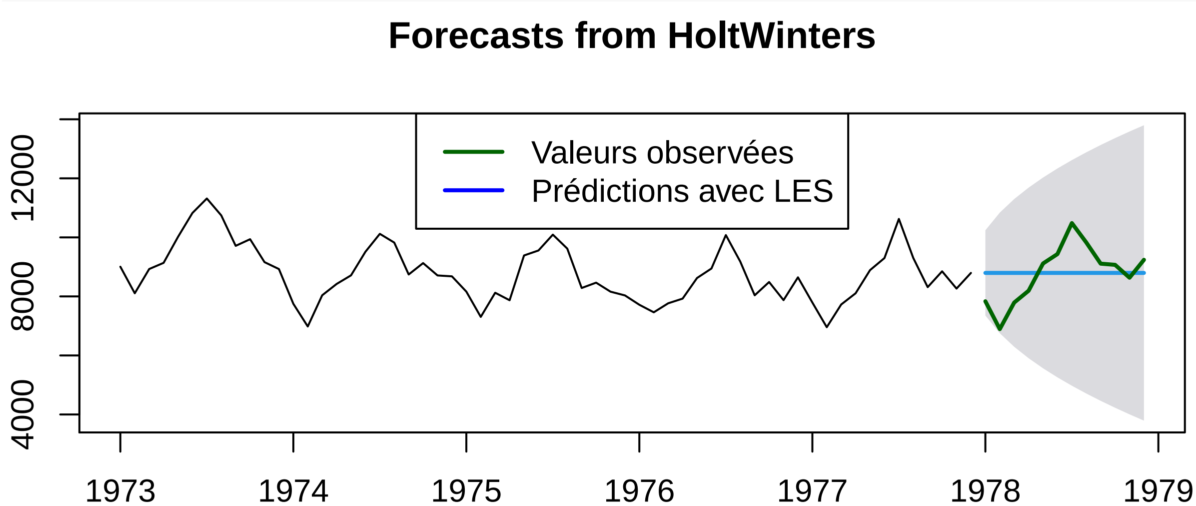
Remarque : La méthode de Holt-Winters présente plusieurs méthodes, plus haut nous avons présentés celle que nous allons utiliser, à savoir la méthode dite "linéaire".

c) Lissage exponentiel simple

```r
modLES <- HoltWinters(USA_train, gamma = FALSE, beta = FALSE)
summary(modLES)
```

```
##              Length Class  Mode
## fitted       118    mts    numeric
## x             60    ts     numeric
## alpha          1    -none- numeric
## beta           1    -none- logical
## gamma          1    -none- logical
## coefficients   1    -none- numeric
## seasonal       1    -none- character
## SSE            1    -none- numeric
## call           4    -none- call
```

```r
predLES <- forecast(modLES, level = 0.95, h=1*12)
plot(predLES)
points(USA_test,type='l',col='darkgreen',lwd=2)
legend('top',c("Valeurs observées","Prédictions avec LES"),col=c("darkgreen","blue"),lty=rep(1,2),lwd =
```

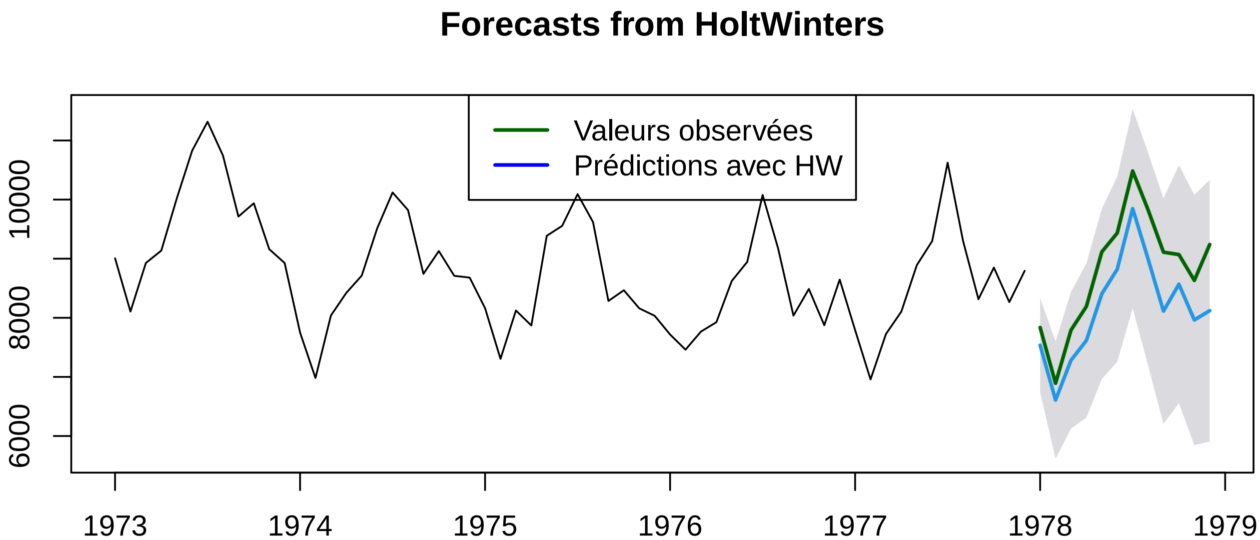


Ici, nous remarquons que modèle n'est pas assez efficace pour prédire notre série. Cela s'explique en grande partie par le fait que se modèle convient aux séries sans tendance claire, ni tendance saisonnière.

d) HoltWinters

```
modHW <- HoltWinters(USA_train)

predHW <- forecast(modHW, level = 0.95, h=1*12)
plot(predHW)
points(USA_test,type='l',col='darkgreen',lwd=2)
legend('top',c("Valeurs observées","Prédictions avec HW"),col=c("darkgreen","blue"),lty=rep(1,2),lwd = :
```



Grâce au graphique, nous observons que le modèle est très efficace car il prédit très bien la série réelle, cependant il n'est pas aussi performant que le modèle ARIMA présenté plus haut.

Nous avons vu que seul le modèle par lissage exponentiel simple ne réalisait pas de bonnes prédictions, et que le modèle ARIMA(0,1,1)(0,1,1)[12] via la méthode "auto_arima" était le meilleur.

5. Proposer différents modèles ARMA et sélectionner celui qui vous semble le plus pertinent en le justifiant. Comparer la prédiction obtenue avec ce modèle avec celle des 3 méthodes descriptives.

Dans cette partie, nous allons proposer différents modèles ARMA et selectionner celui qui nous parait le plus pertinent pour notre série. Nous allons tout d'abord utiliser le critère d'AIC et ensuite utiliser le critère BIC afin de pouvoir choisir le meilleur modèle.

a) Selection selon le critère AIC

```
print("Modèle ARIMA(1,1,0) avec différentes saisonnalité")
```

```
## [1] "Modèle ARIMA(1,1,0) avec différentes saisonnalité"
```

```
AIC(arima(USA_train,order=c(1,1,0), seasonal=c(1,1,0)))
```

```
## [1] 692.0307
```

```
AIC(arima(USA_train,order=c(1,1,0), seasonal=c(0,1,1)))
```

```
## [1] 691.0756
```

```
AIC(arima(USA_train,order=c(1,1,0), seasonal=c(1,1,1)))
```

```
## [1] 693.0569
```

```
print("Modèle ARIMA(0,1,1) avec différentes saisonnalité")
```

```
## [1] "Modèle ARIMA(0,1,1) avec différentes saisonnalité"
```

```
AIC(arima(USA_train,order=c(0,1,1), seasonal=c(1,1,0)))
```

```
## [1] 689.9487
```

```
AIC(arima(USA_train,order=c(0,1,1), seasonal=c(0,1,1)))
```

```
## [1] 689.5419
```

```
AIC(arima(USA_train,order=c(0,1,1), seasonal=c(1,1,1)))
```

```
## [1] 691.534
```

```
print("Modèle ARIMA(1,1,1) avec différentes saisonnalité")
```

```
## [1] "Modèle ARIMA(1,1,1) avec différentes saisonnalité"
```

```
AIC(arima(USA_train,order=c(1,1,1), seasonal=c(1,1,0)))
```

```
## [1] 691.9348
```

```
AIC(arima(USA_train,order=c(1,1,1), seasonal=c(0,1,1)))
```

```
## [1] 691.5315
```

```
AIC(arima(USA_train,order=c(1,1,1), seasonal=c(1,1,1)))
```

```
## [1] 693.5237
```

Parmis les 9 modèles choisis nous remarquons que les 3 meilleurs sont les modèles ARIMA(0,1,1) dont deux qui possèdent une AIC très proche (environ 689,94 et 689,54), nous rejetons les 7 autres, et allons désormais utiliser le critère BIC afin de les départager.

b) Selection selon le critère BIC

```
BIC(arima(USA_train,order=c(0,1,1), seasonal=c(1,1,0)))
```

```
## [1] 695.4991
```

```
BIC(arima(USA_train,order=c(0,1,1), seasonal=c(0,1,1)))
```

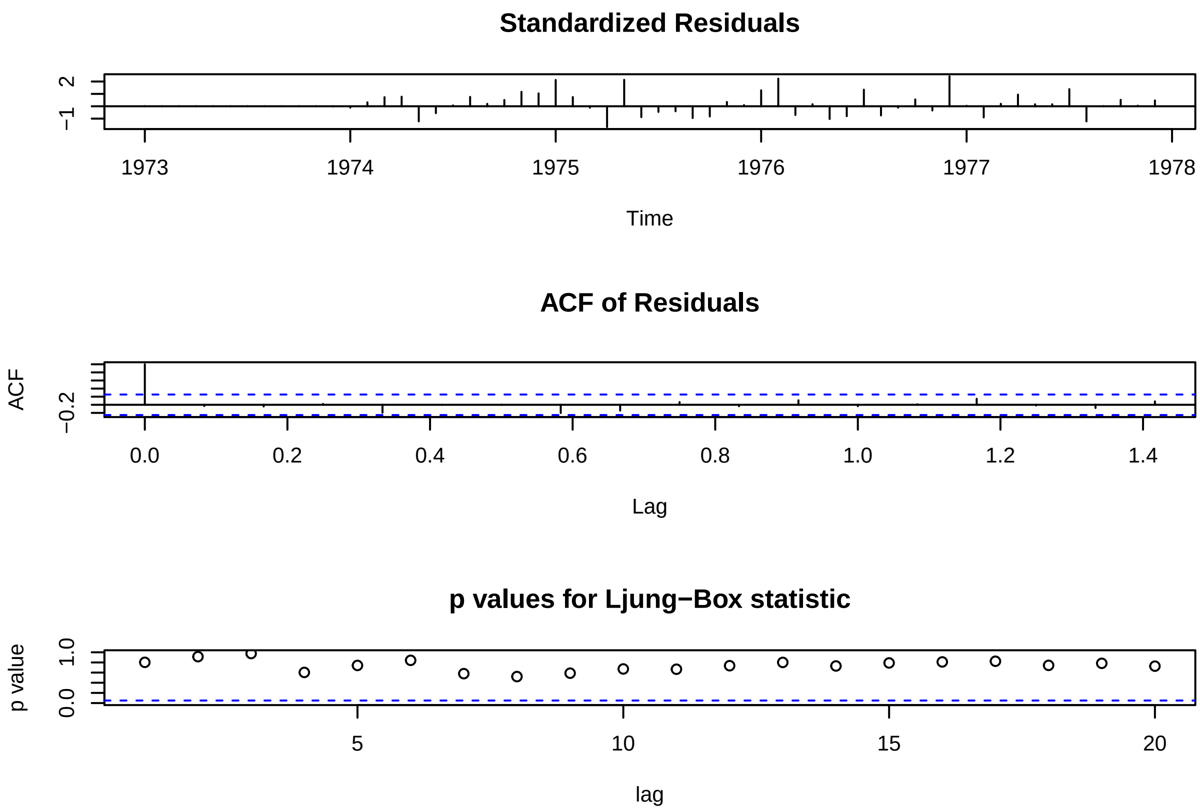
```
## [1] 695.0923
```

En utilisant le critère BIC, nous remarquons que ces deux modèles ont encore des valeurs très proches, cependant le modèle ARIMA(0,1,1)(0,1,1) possède un léger avantage qui fait que nous allons le selectionner aux dépens de l'autre. Nous notons qu'il s'agit du modèle obtenus grâce à la méthode "auto.arima" dans la partie 4.

6. Etude les résidus du modèle ARMA sélectionné.

Comme indiqué plus haut, nous avons séléctionnés le modèle ARIMA(0,1,1)(0,1,1). Nous allons nous intéresser à l'étude des résidus de ce modèle.

```
tsdiag(modAUTO, gof.lag=20)
```



Nous allons nous interesser au second graphique (qui correspond au graphique de l'ACF) afin de conclure sur la nature des résidus du modèle associé.

En effet, nous observons dans le graphique de l'ACF que les résidus semblent être centrés en 0, et dont la grande majorité des valeurs sont situées entre les deux lignes horizontales bleues, ce qui nous permet de conclure que les résidus sont indépendants et suivent une loi normale centrée. Par conséquent les résidus que l'on a dans ce modèle représentent un bruit blanc, et de ce fait, nous n'avons plus d'éléments à étudier dans les résidus de ce modèle.