Analyse des séries temporelles avec R

ENSAE-ENSAI
Formation continue
(Cepe)

6 - Compléments

Alain Quartier-la-Tente

Objectifs de cette séquence

- Présenter quelques compléments sans exercice associé

Sommaire

Régresseurs externes classiques

Dans certaines méthodes (régression linéaire, ARIMA, etc. mais pas ETS) permettent de rajouter des régresseurs externes qui peuvent aider à l'analyse/prévision

- polynômes sur les dates (e.g. tendance linéaire) (on peut s'aider de `forecast::tslm()`)
- indicatrices sur la périodicité (avec variable de contraste) :
  - Sur les jours de la semaine
  - Sur les mois/trimestres
- Régresseurs JO :
  - On compte le nombre de lundis, mardis, ... dans le mois et on construit des variables contraste (en faisant des éventuels regroupement)
  - Régresseurs sur les jours fériés (éventuellement regroupés avec dimanches) + éventuels effets graduels (notamment fêtes mobiles)

Régresseurs de Fourier

Lorsque la périodicité est trop élevée ou lorsqu'il y plusieurs saisonnalités, ajouter des indicatrices peut être trop coûteux.

Solution : ajouter des variables sinusoïdales aux fréquences étudiées !

$$\cos\left(\frac{2k\pi}{m}\right) \quad \sin\left(\frac{2k\pi}{m}\right) \qquad \text{avec } 0 < k < m$$

Généralement $k \ll m$ lorsque m est grand

- Pour séries mensuelles : $m = 12$
- Pour les séries hebdomadaires $m = 365.25/7 \simeq 52$
- Pour les séries journalières $m = 365.25$ pour saisonnalité annuelle, $m = 365.25/12 \simeq 30$ pour saisonnalité mensuelle.

TBATS (1)

Une transformation de Box-Cox est utilisée :

$$y_t^{(\lambda)} = \begin{cases} \frac{y_t^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \log(y_t) & \text{if } \lambda = 0 \end{cases}$$

Ensuite un modèle avec *Trigonometric seasonality, ARMA errors, Trend and Seasonal components* est calculé : c'est tout ce que l'on a vu dans les précédents cours.

Voir `?forecast::tbats()`

TBATS (2)

$$\begin{cases} y_t^{(\lambda)} = l_{t-1} + \phi b_{t-1} + \sum_{i=1}^{T} s_{t-m_i}^{(i)} + d_t \text{ and } d_t \sim ARMA(p,q) \\ l_t = l_{t-1} + \phi b_{t-1} + \alpha d_t \\ b_t = \phi b_{t-1} + \beta d_t \end{cases}$$

$$\begin{cases} s_t^{(i)} = \sum_{j=1}^{k_i} s_{j,t}^{(i)} \\ s_{j,t}^{(i)} = s_{j,t-1}^{(i)} \cos \omega_j + s_{j,t-1}^{*(i)} \sin \omega_j + \gamma_1^{(i)} d_t \\ s_{j,t-1}^{*(i)} = s_{j,t-1}^{(i)} \sin \omega_j + s_{j,t-1}^{*(i)} \cos \omega_j + \gamma_2^{(i)} d_t \end{cases} \qquad \text{and } \omega_j = \frac{2\pi j}{m_i}$$

Notation : $TBATS(omega, p, q, phi, < m1, k1 >, ..., < mJ, kJ >)$ avec

- $omega$ = paramètre de Box-Cox
- (p, q) = ARMA(p,q)
- phi = paramètre d'amortissement
- $m_1, ..., m_J$ les périodicités et $k_1, ..., k_J$ le nombre de termes de fourrier

TBATS (3)

```
library(forecast)
tbats(USAccDeaths)
```

```
TBATS(1, {0,0}, -, {<12,5>})

Call: tbats(y = USAccDeaths)

Parameters
  Alpha: 0.5950012
  Gamma-1 Values: -0.01207202
  Gamma-2 Values: 0.01159708

Seed States:
            [,1]
 [1,] 9357.62824
 [2,] -999.76784
 [3,]  279.84741
 [4,] -193.88870
```

Analyse des données haute fréquence

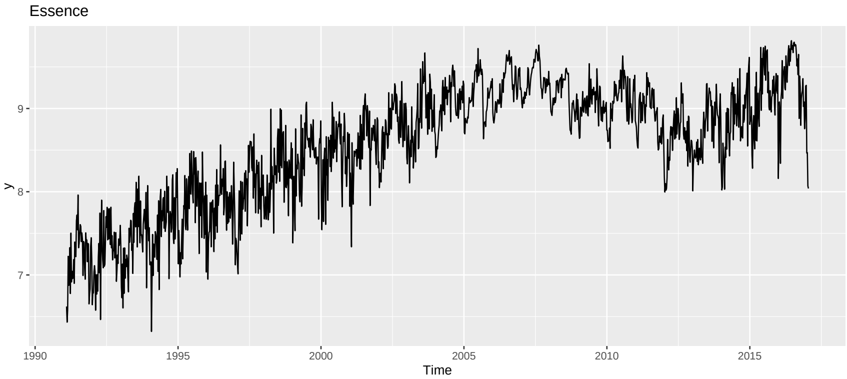
Pour les séries à haute fréquence (hebdomadaires, journalières, horaires, etc.)

- Les effets calendaires peuvent être relativement importants (notamment les jours feriés)
- On peut utiliser des modèles avec des régresseurs externes (e.g. fourrier)
- TBATS
- On peut combiner des modèles STL + ETS ou ARIMA sur série désaisonnalisée

Voir https://otexts.com/fpp2/weekly.html et https://otexts.com/fpp3/weekly.html pour des exemples

Exemples (1)

```r
library(forecast);library(ggplot2);library(patchwork)
y <- fpp2::gasoline
autoplot(y, main = "Essence")
```

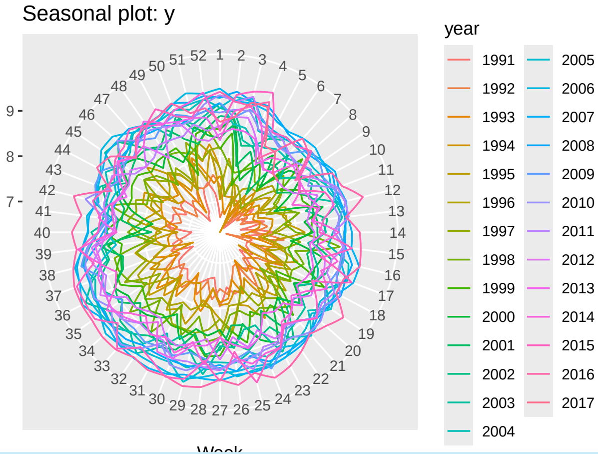


Exemples (2)

```r
frequency(y)
```

```
[1] 52.17857
```

```r
ggseasonplot(y, polar = TRUE)
```



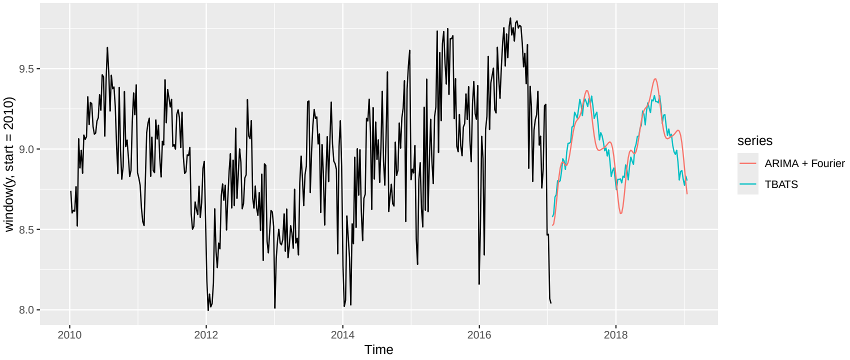
Exemples (3)

```
# Saisonnalité annuelle et mensuelle :
tbats <- tbats(y, seasonal.periods = c(365.25/7, 365.25/12/7))
arima_fourier <- auto.arima(y, seasonal = FALSE, xreg = fourier(y,K=5))
tbats
```

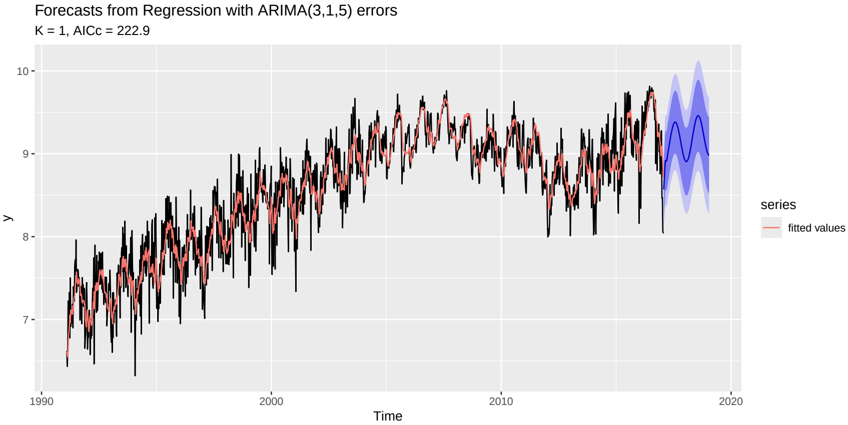
```
TBATS(1, {0,0}, 0.8, {<4.35,6>, <52.18,1>})

Call: tbats(y = y, seasonal.periods = c(365.25/7, 365.25/12/7))

Parameters
  Alpha: 0.2350472
  Beta: -0.03474354
  Damping Parameter: 0.800008
  Gamma-1 Values: -0.0002755258 6.336552e-05
  Gamma-2 Values: 5.566591e-05 5.769457e-05

Seed States:
              [,1]
 [1,]  6.949478933
 [2,]  0.021926676
 [3,] -0.030850684
 [4,]  0.006212825
 [5,] -0.009773578
```

Exemples (4)

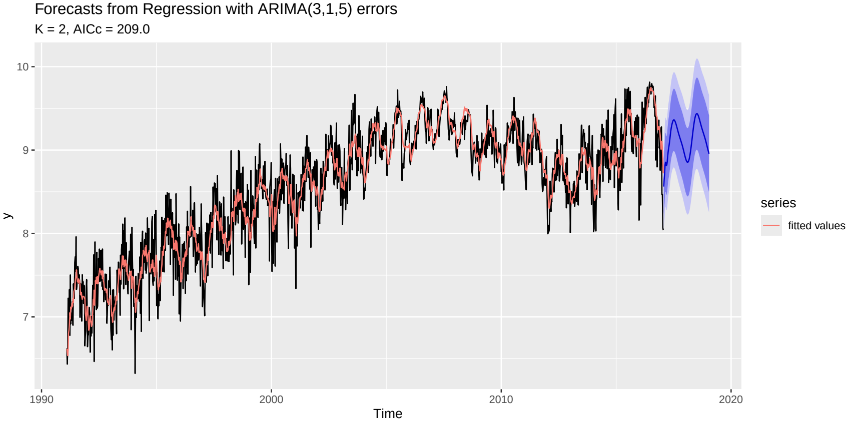
```
autoplot(window(y, start = 2010)) +
    autolayer(forecast(tbats, h = 52*2)$mean, series="TBATS")+
    autolayer(forecast(arima_fourier, h = 52*2, xreg = fourier(y, K=5, h = 52*2)),
              series = "ARIMA + Fourier")
```



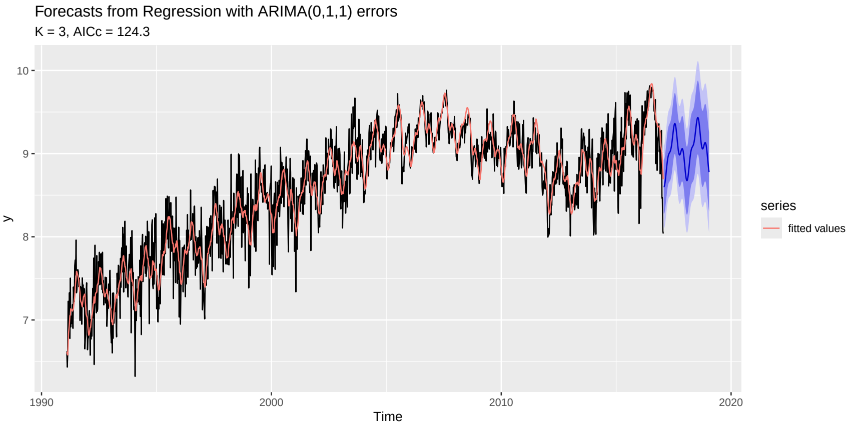
Exemples analyse de K (1)



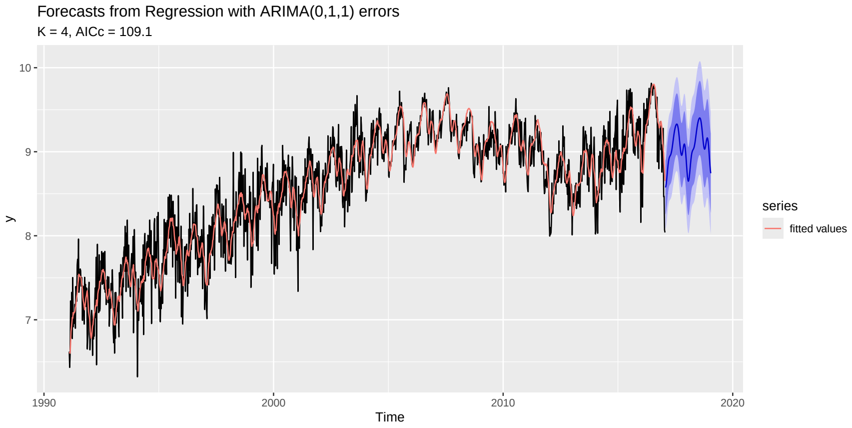
Exemples analyse de K (2)



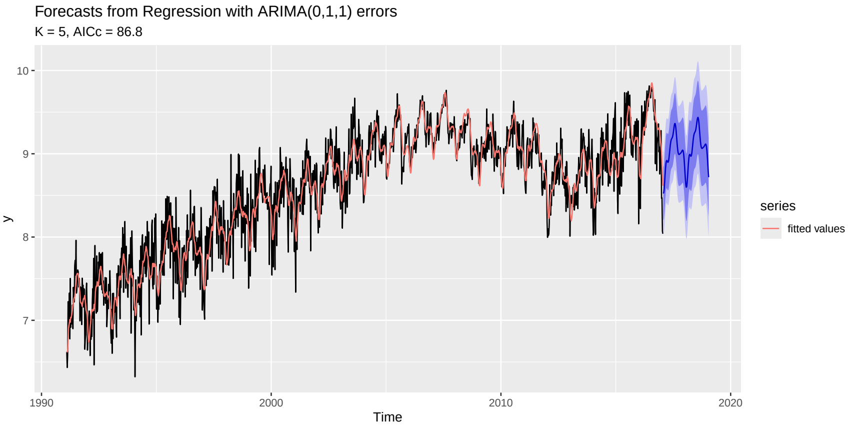
Exemples analyse de K (3)



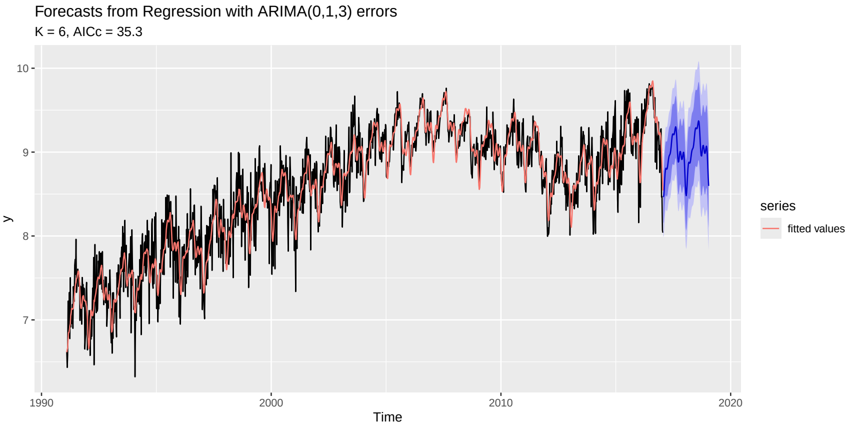
Exemples analyse de K (4)



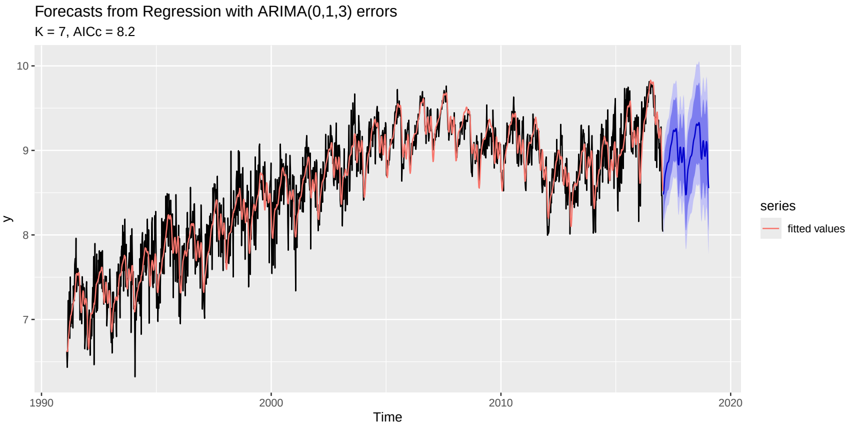
Exemples analyse de K (5)



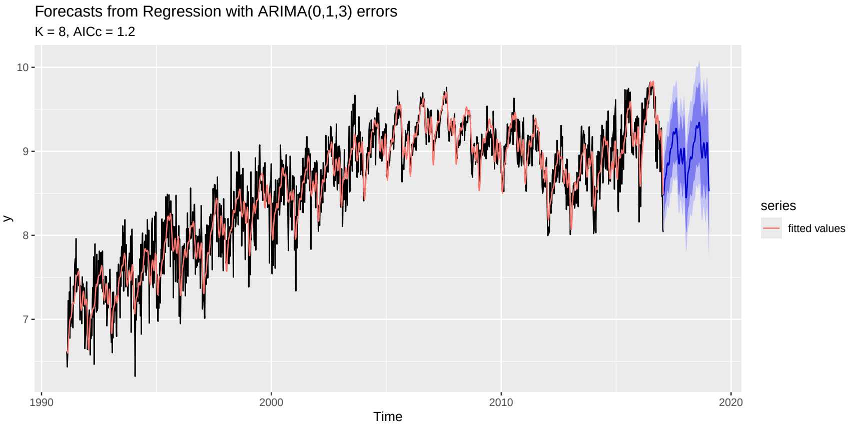
Exemples analyse de K (6)



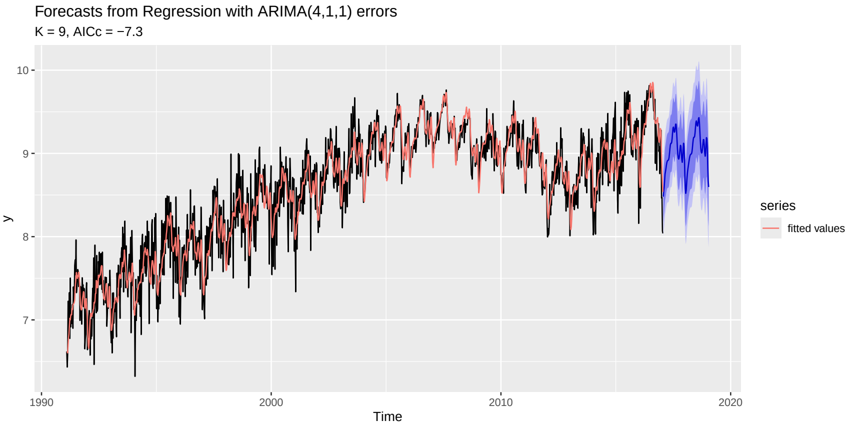
Exemples analyse de K (7)



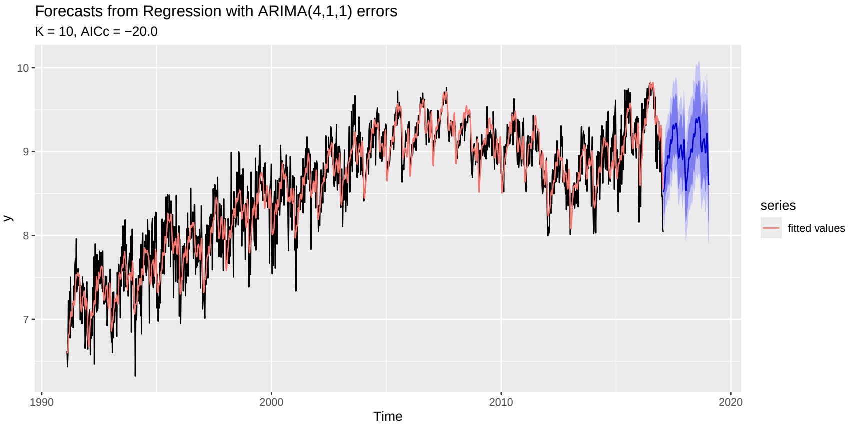
Exemples analyse de K (8)



Exemples analyse de K (9)



Exemples analyse de K (10)



Sommaire

Modèles ECM

Les modèles à correction d'erreur (ECM) permettent de mettre en relation deux variables x_t, y_t non-stationnaires qui partagent la même tendance stochastique. Modèle suivant est utilisé :

$$\Delta y_t = \underbrace{\gamma + \sum_{i=1}^{p} \Delta y_{t-i} + \sum_{i=1}^{p} \Delta x_{t-i}}_{\text{court terme}} + \alpha \underbrace{(y_{t-1} - \beta_0 - \beta_1 x_{t-1})}_{\text{long terme}} + \varepsilon_t$$

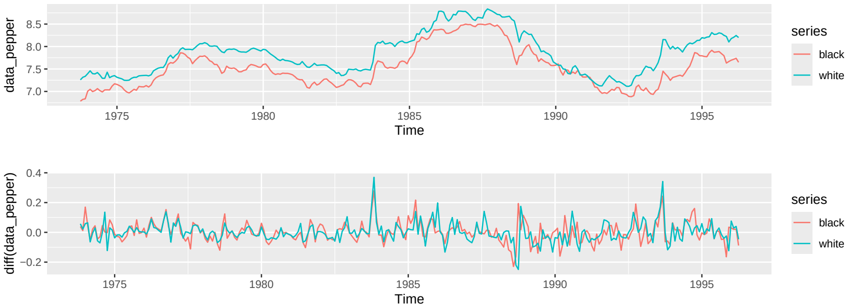
Peut s'estimer par double MCO : long terme puis court terme sur les résidus. On peut s'aider de `dynlm::dynlm()` ou utiliser le package `ecm`.

Pour que le modèle soit valide il faut que $y_{t-1} - \beta_0 - \beta_1 x_{t-1}$ soit stationnaire : on peut faire un test de racine unité sur les résidus ou appliquer le test de Johansen (`urca::ca.jo`).

Généralement $\alpha < 0$: s'interprète comme une force de rappel.

Exemple (1)

```r
# install.packages("PepperPrice")
library(urca);library(dynlm);library(forecast);library(ggplot2)
data("PepperPrice", package = "AER")
# On passe au log pour analyser les différences comme des évolutions
data_pepper <- log(PepperPrice)
autoplot(data_pepper) / autoplot(diff(data_pepper))
```



Exemple (2)

```
# Séries sont dites I(1) :
# Elles ne sont pas stationnaires
tseries::kpss.test(data_pepper[,"black"])
```

```

	KPSS Test for Level Stationarity

data:  data_pepper[, "black"]
KPSS Level = 0.60007, Truncation lag parameter = 5, p-value = 0.02263
```

```
tseries::kpss.test(data_pepper[,"white"])
```

```

	KPSS Test for Level Stationarity

data:  data_pepper[, "white"]
KPSS Level = 0.61733, Truncation lag parameter = 5, p-value = 0.02106
```

```
# Mais les séries différenciées le sont
tseries::kpss.test(diff(data_pepper[,"black"], 1))
```

Exemple (3)

```
    KPSS Test for Level Stationarity

data:  diff(data_pepper[, "black"], 1)
KPSS Level = 0.15877, Truncation lag parameter = 5, p-value = 0.1
```

```r
tseries::kpss.test(diff(data_pepper[,"white"], 1))
```

```
    KPSS Test for Level Stationarity

data:  diff(data_pepper[, "white"], 1)
KPSS Level = 0.13062, Truncation lag parameter = 5, p-value = 0.1
```

```r
# Le test de Johansen doit se lire de manière croissante avec r
# r=0 signifie qu'il n'y a pas de relation de co-intégration
# si on le rejette (test > valeurs critiques), on regarde le test suivant
# Dans notre cas il n'y a que deux tests car on a que deux variables
# Le test est plus général pour les cas où l'on fait des VECM
# (potentiellement plusieurs relations de cointegration)
# Ici on conclut qu'il y a bien relation de cointegration
summary(ca.jo(data_pepper))
```

Exemple (4)

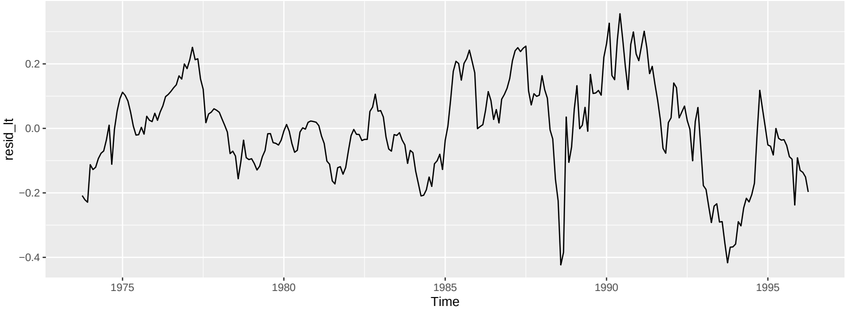
```
######################
# Johansen-Procedure #
######################

Test type: maximal eigenvalue statistic (lambda max) , with linear trend

Eigenvalues (lambda):
[1] 0.04923322 0.01262841

Values of teststatistic and critical values of test:

          test 10pct  5pct  1pct
r <= 1 |  3.42  6.50  8.18 11.65
r = 0  | 13.58 12.91 14.90 19.19

Eigenvectors, normalised to first column:
(These are the cointegration relations)

           black.l2  white.l2
black.l2  1.0000000  1.000000
```

Exemple (5)

```
white.l2 -0.8904272 -6.177004

Weights W:
(This is the loading matrix)

           black.l2    white.l2
black.d -0.07423986 0.001970073
white.d  0.02088163 0.002811481
```

```r
# On estime la relation de long-terme
lm_lt <- lm(black ~ white, data = data_pepper)
resid_lt <- ts(residuals(lm_lt), start = start(data_pepper),
               frequency = frequency(data_pepper))
autoplot(resid_lt)
```

Exemple (6)



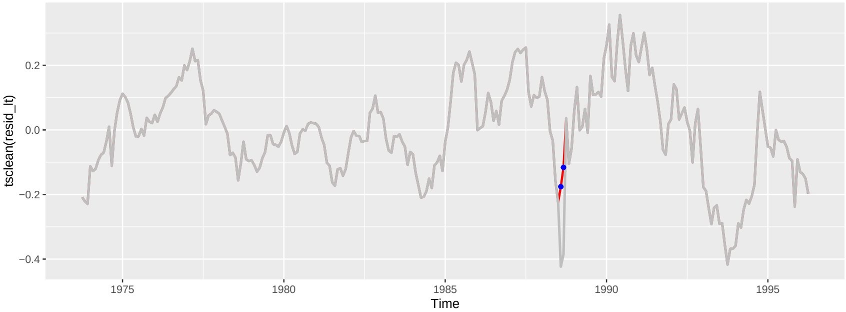

```
# La série est bien stationnaire
tseries::kpss.test(resid_lt)
```

```

    KPSS Test for Level Stationarity

data:  resid_lt
KPSS Level = 0.23504, Truncation lag parameter = 5, p-value = 0.1
```

Exemple (7)

```r
# Rmq il y a quelques points atypiques que l'on pourrait corriger
# en ajoutant par exemple des indicatrices
# On peut aussi utiliser la fonction forecast::tsoutliers() pour les repérer
# En reprenant le code disponible ici
# https://robjhyndman.com/hyndsight/tsoutliers/ :
autoplot(tsclean(resid_lt), series="clean", color='red', lwd=0.9) +
    autolayer(resid_lt, series="original", color='gray', lwd=1) +
    geom_point(data = tsoutliers(resid_lt) %>% as.data.frame(),
               aes(x=time(resid_lt)[index], y=replacements), col='blue')
```

Exemple (8)




```
data <- ts.union(data_pepper, resid_lt)
colnames(data) <- c(colnames(data_pepper), "long_term")
# On a bien une force de rappel négative
summary(
    dynlm(diff(black, 1) ~ lag(diff(black, 1),-1) +
            lag(diff(white, 1),-1) +
            lag(long_term, -1), data = data)
)
```

Exemple (9)

```
Time series regression with "ts" data:
Start = 1973(12), End = 1996(4)

Call:
dynlm(formula = diff(black, 1) ~ lag(diff(black, 1), -1) + lag(diff(white,
    1), -1) + lag(long_term, -1), data = data)

Residuals:
      Min        1Q    Median        3Q       Max
-0.200930 -0.035547 -0.005632  0.028757  0.228076

Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.001830   0.003876   0.472  0.63724
lag(diff(black, 1), -1)  0.327341   0.069431   4.715 3.92e-06 ***
lag(diff(white, 1), -1)  0.051700   0.069819   0.740  0.45966
lag(long_term, -1)      -0.072551   0.027006  -2.686  0.00768 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Exemple (10)

```
Residual standard error: 0.06345 on 265 degrees of freedom
Multiple R-squared:  0.144, Adjusted R-squared:  0.1343
F-statistic: 14.86 on 3 and 265 DF,  p-value: 5.698e-09
```