

Économétrie des Séries Temporelles

Fiche TD R #2

Processus ARMA stationnaires

Packages

```
library(readr)
library(zoo)
library(astsa)
library(forecast)
library(stats)
library(tseries)
library(aTSA)
#install.packages("aTSA")
```

Données (identiques au TP1)

Nice : https://github.com/bilelsanhaji/EdSTM1/blob/main/Data/SH_MIN006088001.csv

Paris : https://github.com/bilelsanhaji/EdSTM1/blob/main/Data/SH_MIN175114001.csv

```
urlSHnice <- "https://raw.githubusercontent.com/bilelsanhaji/EdSTM1/refs/heads/main/Data/SH_MIN006088001.csv"
InsoNice <- read_delim(urlSHnice,
                       delim = ";",
                       escape_double = FALSE,
                       col_types = cols(YYYYMM = col_date(format = "%Y%m")),
                       comment = "#", trim_ws = TRUE)

urlSHparis <- "https://raw.githubusercontent.com/bilelsanhaji/EdSTM1/refs/heads/main/Data/SH_MIN175114001.csv"
InsoParis <- read_delim(urlSHparis,
                        delim = ";",
                        escape_double = FALSE,
                        col_types = cols(YYYYMM = col_date(format = "%Y%m")),
                        comment = "#", trim_ws = TRUE)

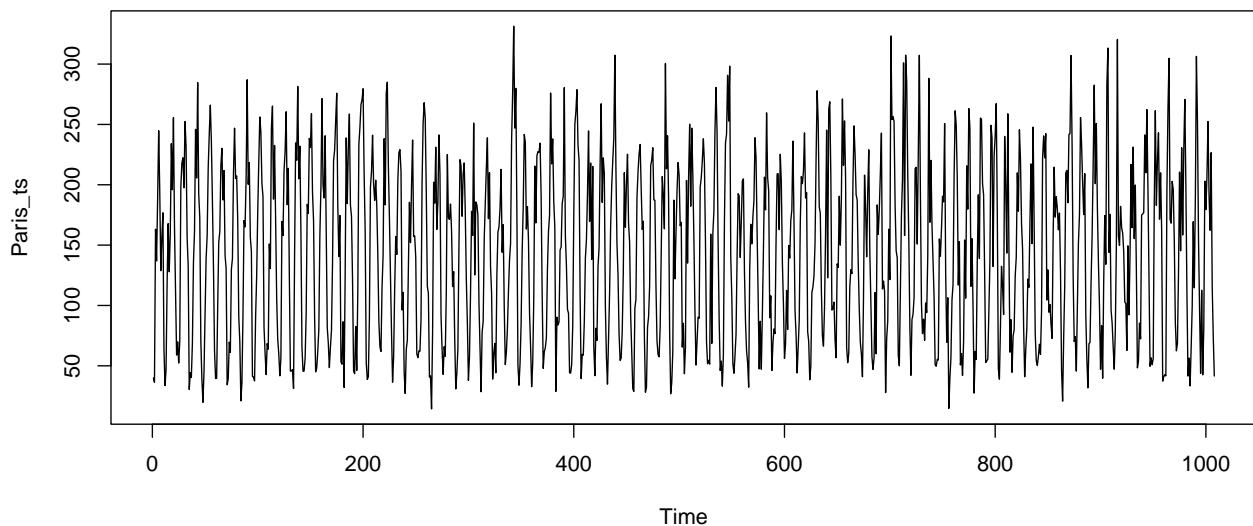
Nice_ts = ts(InsoNice$VALEUR)
Paris_ts = ts(InsoParis$VALEUR)
```

Exercice 1

À partir des données d'insolation de Nice et Paris, utilisez les séries pour

- donner une représentation graphique et tester statistiquement :

```
plot(Nice_ts)
plot(Paris_ts)
```



1. la stationnarité #KPSS

```
# tseries::kpss.test()
#x = rnorm(1000) # is level stationary
#tseries::kpss.test(x)
tseries::kpss.test(Nice_ts)

##
## KPSS Test for Level Stationarity
##
## data: Nice_ts
## KPSS Level = 0.058965, Truncation lag parameter = 7, p-value = 0.1
aTSA::kpss.test(Nice_ts)

## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##   lag stat p.value
##     7  5.8    0.01
## -----
## Type 2: with drift no trend
##   lag   stat p.value
##     7 0.0145    0.1
## -----
## Type 1: with drift and trend
##   lag   stat p.value
##     7 0.0151    0.1
## -----
## Note: p.value = 0.01 means p.value <= 0.01
##       : p.value = 0.10 means p.value >= 0.10

#ts.plot(x)
#y = cumsum(x) # has unit root
#tseries::kpss.test(y)
#ts.plot(y)
# aTSA::kpss.test()
```

```

#Dickey-Fuller (ADF)
tseries::adf.test(Nice_ts)

##
##  Augmented Dickey-Fuller Test
##
## data: Nice_ts
## Dickey-Fuller = -9.551, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
aTSA::adf.test(Nice_ts)

## Augmented Dickey-Fuller Test
## alternative: stationary
##
## Type 1: no drift no trend
##      lag   ADF p.value
## [1,] 0 -3.71 0.0100
## [2,] 1 -4.18 0.0100
## [3,] 2 -4.79 0.0100
## [4,] 3 -4.52 0.0100
## [5,] 4 -3.62 0.0100
## [6,] 5 -2.72 0.0100
## [7,] 6 -2.04 0.0418
## Type 2: with drift no trend
##      lag   ADF p.value
## [1,] 0 -12.6 0.01
## [2,] 1 -15.9 0.01
## [3,] 2 -21.9 0.01
## [4,] 3 -26.1 0.01
## [5,] 4 -27.4 0.01
## [6,] 5 -27.1 0.01
## [7,] 6 -25.8 0.01
## Type 3: with drift and trend
##      lag   ADF p.value
## [1,] 0 -12.6 0.01
## [2,] 1 -15.9 0.01
## [3,] 2 -21.9 0.01
## [4,] 3 -26.1 0.01
## [5,] 4 -27.4 0.01
## [6,] 5 -27.2 0.01
## [7,] 6 -25.9 0.01
## ----
## Note: in fact, p.value = 0.01 means p.value <= 0.01

#Philips Perron (PP)
tseries::pp.test(Nice_ts)

##
##  Phillips-Perron Unit Root Test
##
## data: Nice_ts
## Dickey-Fuller Z(alpha) = -217.81, Truncation lag parameter = 7, p-value
## = 0.01
## alternative hypothesis: stationary

```

```

aTSA::pp.test(Nice_ts)

## Phillips-Perron Unit Root Test
## alternative: stationary
##
## Type 1: no drift no trend
##   lag Z_rho p.value
##     7    -20    0.01
## -----
## Type 2: with drift no trend
##   lag Z_rho p.value
##     7   -218    0.01
## -----
## Type 3: with drift and trend
##   lag Z_rho p.value
##     7   -218    0.01
## -----
## Note: p-value = 0.01 means p.value <= 0.01

```

2. l'autocorrélation

3. la normalité

```

# dickey-fuller
# dickey-fuller augmenté (adf)
#tseries::adf.test()
tseries::adf.test(Nice_ts)

```

```

##
## Augmented Dickey-Fuller Test
##
## data: Nice_ts
## Dickey-Fuller = -9.551, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
#atSA::adf.test()
aTSA::adf.test(Nice_ts)

```

```

## Augmented Dickey-Fuller Test
## alternative: stationary
##
## Type 1: no drift no trend
##   lag ADF p.value
## [1,] 0 -3.71 0.0100
## [2,] 1 -4.18 0.0100
## [3,] 2 -4.79 0.0100
## [4,] 3 -4.52 0.0100
## [5,] 4 -3.62 0.0100
## [6,] 5 -2.72 0.0100
## [7,] 6 -2.04 0.0418
## Type 2: with drift no trend
##   lag ADF p.value
## [1,] 0 -12.6 0.01
## [2,] 1 -15.9 0.01
## [3,] 2 -21.9 0.01
## [4,] 3 -26.1 0.01

```

```

## [5,] 4 -27.4 0.01
## [6,] 5 -27.1 0.01
## [7,] 6 -25.8 0.01
## Type 3: with drift and trend
##      lag ADF p.value
## [1,] 0 -12.6 0.01
## [2,] 1 -15.9 0.01
## [3,] 2 -21.9 0.01
## [4,] 3 -26.1 0.01
## [5,] 4 -27.4 0.01
## [6,] 5 -27.2 0.01
## [7,] 6 -25.9 0.01
## -----
## Note: in fact, p.value = 0.01 means p.value <= 0.01
#ur.df()
# philips-perron (pp)
#tseries::pp.test()
tseries::pp.test(Nice_ts)

##
## Phillips-Perron Unit Root Test
##
## data: Nice_ts
## Dickey-Fuller Z(alpha) = -217.81, Truncation lag parameter = 7, p-value
## = 0.01
## alternative hypothesis: stationary
#aTSA::pp.test()
aTSA::pp.test(Nice_ts)

## Phillips-Perron Unit Root Test
## alternative: stationary
##
## Type 1: no drift no trend
##      lag Z_rho p.value
##      7 -20 0.01
## -----
## Type 2: with drift no trend
##      lag Z_rho p.value
##      7 -218 0.01
## -----
## Type 3: with drift and trend
##      lag Z_rho p.value
##      7 -218 0.01
## -----
## Note: p-value = 0.01 means p.value <= 0.01
#pp.test()
#####
#exemple de la R Documentation pp.test {tseries}
# x = rnorm(1000) # no unit-root
# ts.plot(x)
# tseries::pp.test(x)
# y = cumsum(x) # has unit root
# ts.plot(y)

```

```

# tseries::pp.test(y)

#kpss
tseries::kpss.test(Nice_ts)

##
## KPSS Test for Level Stationarity
##
## data: Nice_ts
## KPSS Level = 0.058965, Truncation lag parameter = 7, p-value = 0.1
aTSA::kpss.test(Nice_ts)

## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##   lag stat p.value
##     7  5.8    0.01
## -----
## Type 2: with drift no trend
##   lag stat p.value
##     7 0.0145    0.1
## -----
## Type 1: with drift and trend
##   lag stat p.value
##     7 0.0151    0.1
## -----
## Note: p.value = 0.01 means p.value <= 0.01
##       : p.value = 0.10 means p.value >= 0.10

```

2. l'autocorrélation

```

# durbin-watson
# ljung-box
# box-pierce

```

3. la normalité

```

# jarque-bera
# shapiro-wilk

```

(b) estimer et interpréter un AR(1) pour chaque série, puis, sur les résidus :

1. tester autocorrelation, normalité et hétéroscédasticité
2. interprétez tous les résultats obtenus
3. discutez la différence qu'il y a entre les séries

Exercice 2

Simulez un processus AR(1) stationnaire avec 50 observations. Puis

(a) “testez” graphiquement et testez statistiquement :

1. la stationnarité
2. l'autocorrélation
3. la normalité

- (b) estimatez la série simulée et discutez les résultats
- (c) reproduire les étapes (a) et (b) avec 5000 observations
 - 1. la stationnarité
 - 2. l'autocorrélation
 - 3. la normalité
- (b) estimatez la série simulée et discutez les résultats