Time series forecasting

ARIMA models

Julien JACQUES

Université Lumière Lyon 2

Trend and seasonal pattern estimation

ARMA models

Non-seasonal ARIMA models

Seasonal ARIMA models

Heteroscedastic series

Trend and seasonal pattern estimation

Removing trend + seasonal pattern

In order to modelize the stochastic part of the times series, we have to **remove the deterministic part** (trend + seasonal pattern)

We will see two methods:

- Estimation by moving average
- Removing by differencing

Time series components

We assume that the time series can be decomposed into:

$$x_t = T_t + S_t + \epsilon_t$$

where :

- T_t is the trend,
- S_t is the seasonal pattern (of period T)
- ϵ_t is the residual part

Rk: if x_t admits a multiplicative decomposition, $\log x_t$ admits an additive decomposition.

Moving average

A moving average estimation of the trend T_t of order m (m-MA) is:

$$\hat{T}_t = \frac{1}{m} \sum_{j=-k}^{k} x_{t+j}$$

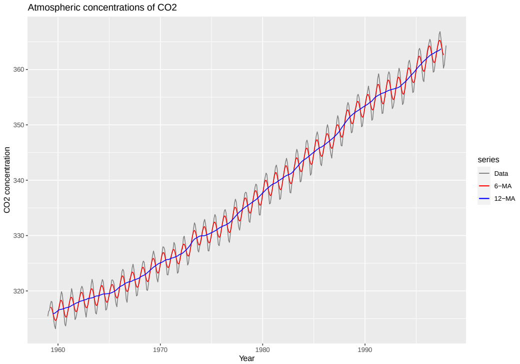
where $m = 2k + 1$.

$\hat{T}_t$ is the average of the m values nearby time t.

- greater is m, greater is the smoothing
- for series with seasonnal pattern of period T, we generally choose $m \geq T$.

Moving average

```r
autoplot(co2, series="Data") +
  autolayer(ma(co2,6), series="6-MA") +
  autolayer(ma(co2,12), series="12-MA") +
  xlab("Year") + ylab("CO2 concentration") +
  ggtitle("Atmospheric concentrations of CO2 ") +
  scale_colour_manual(
    values=c("Data"="grey50","6-MA"="red","12-MA"="blue"),
    breaks=c("Data","6-MA","12-MA"))
```



Moving average

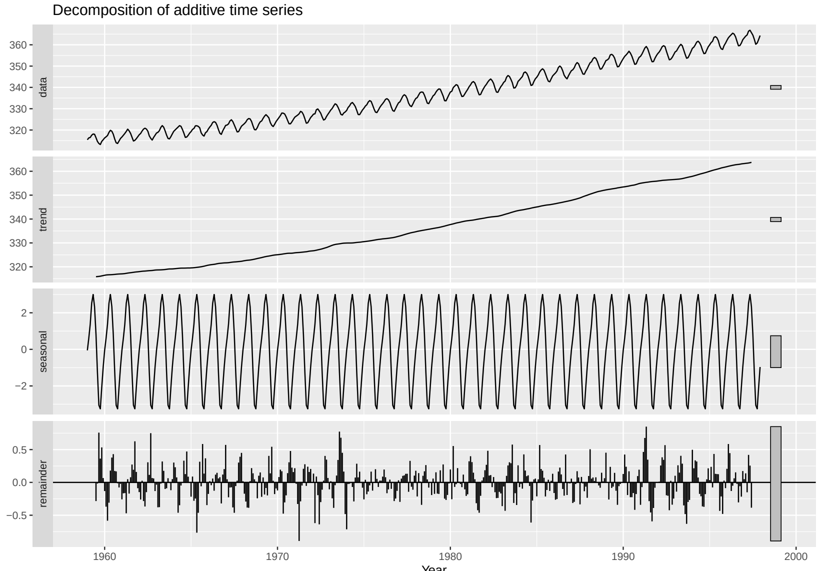
Once the trend T_t has been estimated, we remove it from the series:

$$\tilde{x}_t = x_t - \hat{T}_t$$

Estimation of the **seasonal pattern** is obtained by simply **averaging the values of $\tilde{x}_t$ on each season**.

Moving average

```
autoplot(decompose(co2,type="additive"))+
  xlab('Year')
```



Moving average

Advantage:

- quickly gives an overview of the components of the series

Disadvantage:

- no forecast is possible with such non parametric estimation

Differencing

Let Δ_T be the operator of *lag* T which maps x_t to $x_t - x_{t-T}$:

$$\Delta_T x_t = x_t - x_{t-T}.$$

Differencing

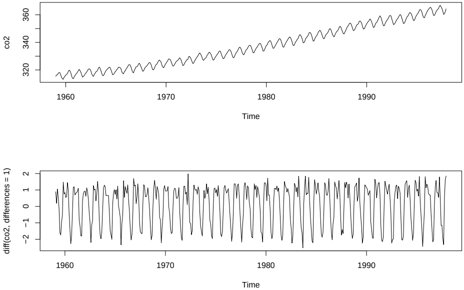
Let x_t be a time series with a polynomial trend of order k :

$$x_t = \sum_{j=0}^{k} a_j t^j + \epsilon_t.$$

Then $\Delta_T x_t$ **admits a polynomial trend of order** $k-1$.

Applying Δ_T reduces by 1 the degree of the polynomial trend.

```
par(mfrow=c(2,1))
plot(co2)
plot(diff(co2,differences=1))
```

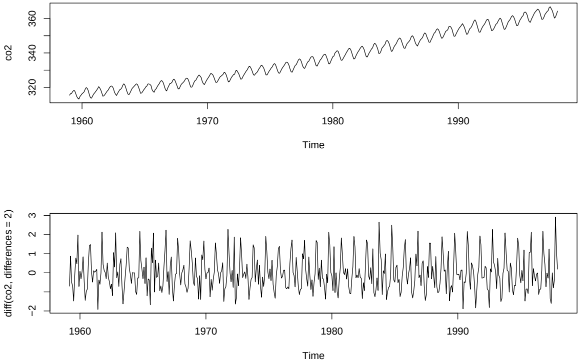


Differencing

Applying Δ_T k times reduces by k the degree of the polynomial trend.

$$\Delta_T^k = \underbrace{\Delta_T \circ \ldots \circ \Delta_T}_{k \text{ times}}$$

```r
par(mfrow=c(2,1))
plot(co2)
plot(diff(co2,differences=2))
```



Differencing

Let x_t be a time series with a ternd T_t and a season pattern S_t of period T:

$$x_t = T_t + S_t + \epsilon_t.$$

Then,

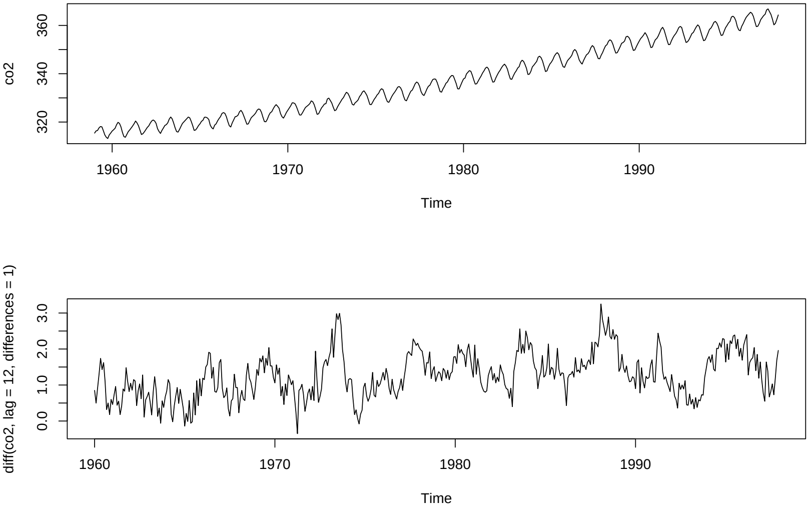
$$\Delta_T x_t = (T_t - T_{t-T}) + (\epsilon_t - \epsilon_{t-T})$$

does not admit any more seasonal pattern.

Applying Δ_T^k remove a seasonal pattern of period T and a polynomial trend of order k

Differencing

```
par(mfrow=c(2,1))
plot(co2)
plot(diff(co2,lag=12,differences=1))
```



Differencing

Advantage:

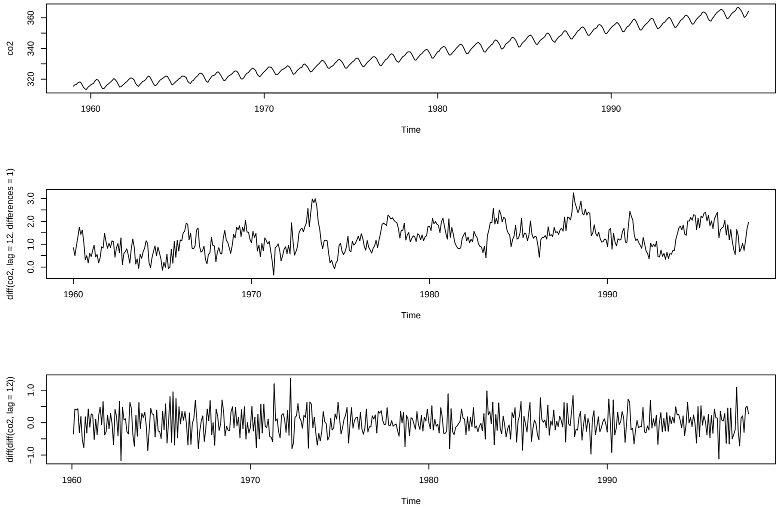
- easy to understand
- allows forecast since we can forecast $\Delta_T x_t$ and then go back to x_t

In practice :

- we start by removing the season by applying Δ_T
- then, if it visually does not seem stationary, we apply again Δ_1
- eventually we apply again Δ_1, but we will try to keep small value for the number k of differencing.

Differencing

```r
par(mfrow=c(3,1))
plot(co2)
plot(diff(co2,lag=12,differences=1))
plot(diff(diff(co2,lag=12)))
```



Stationary series

x_t is a **stationary time series** if, for all s, the distribution of $(x_t, \ldots, x_{t+s})$ does not depend on t.

Consequently, a stationary time serie is one whose properties do not depend on the time at which the series is observed.

In particular, a stationary time serie has:

- no trend
- no season pattern

(A stationary time serie can have a cyclic pattern since its period is not constant.)

ARMA models, one of the main objects of this course, are models for stationary time serie.

White noise

A **white noise** is an independent and identically distributed series with zero mean.

A Gaussian white noise ϵ_t are i.i.d. observations from $\mathcal{N}(0, \sigma^2)$

In such series, there is nothing to forecast. Or more precisely, the best forecast for such series is its means: 0.

White noise

After having differecing our time series for removing trend + seasonal pattern, we have to **check that the residual series is not a white noise**.

In the countrary case, our work is finished: there is nothing else to forecast than trend and seasonal pattern, thus let use exponential smoothing.

```
Box.test(diff(co2,lag=12,differences=1),lag=10,type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  diff(co2, lag = 12, differences = 1)
## X-squared = 1415.4, df = 10, p-value < 2.2e-16
```

Here the p-value is very low, we reject that `diff(co2,lag=12,differences=1)` can be assimilted to a white noise

Exercice

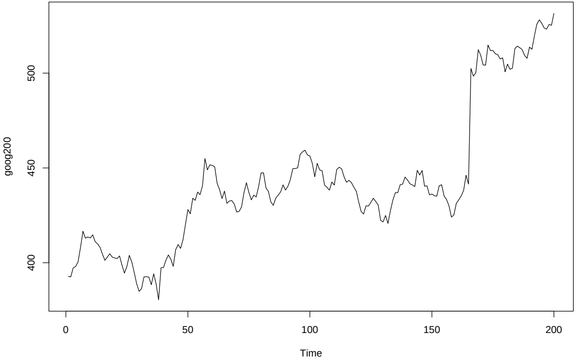
We study the number of passengers per month (in thousands) in air transport, from 1949 to 1960. This time series is available on R (`AirPassengers`).

- Plot this time series graphically. Do you think this process is stationary? Does it show trends and seasonality?
- Apply the differencing method to remove trend and seasonal pattern. Specify the period of the seasonal pattern, the degree of the polynomial trend.
- Does the differenciated series seems stationary?
- Is it a white noise?

Exercice

Same exercice with the Google stock price:

```r
library(fpp2)
plot(goog200)
```



ARMA models

Autoregressive models AR_p

An autoregressive model (x_t) of order p (AR_p) can be written:

$$x_t = c + \epsilon_t + \sum_{j=1}^{p} a_j x_{t-j}, \tag{1}$$

where ϵ_t is a white noise of variance σ^2.

An AR_p model is the sum of:

- a random chock ϵ_t, independent from previous observation
- a linear regression of the previous obseration $\sum_{j=1}^{p} a_j X_{t-j}$

Rk: we restrict AR_p models to stationary models, which implies some restrictions on the value of the coefficients a_j.

AR_p properties

- autocorrelation $\rho(h)$ exponentialy decreases to 0 when $h \to \infty$
- partial autocorrelation $r(h)$ is null for all $h > p$, and is equal to a_p at order p :

$$\begin{aligned} r(h) &= 0 \qquad \forall h > p, \\ r(p) &= a_p. \end{aligned}$$

Example of AR_1

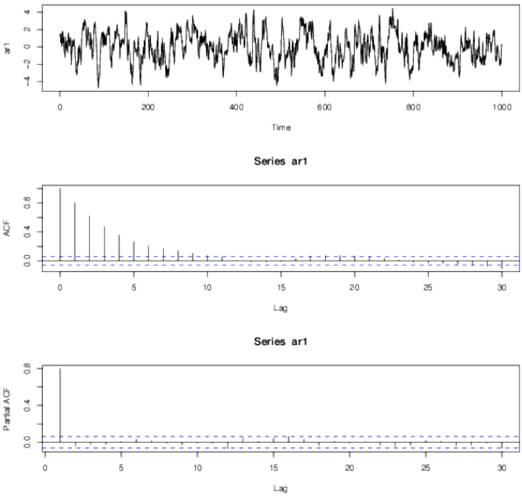


Figure 1: $AR1$ ($x_t = 0.8x_{t-1} + \epsilon_t$), autocorrelation et partial autocorrelation

Example of AR_1

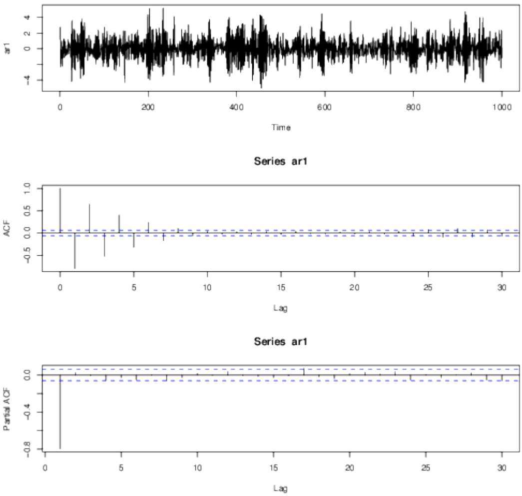


Figure 2: $AR1$ ($x_t = -0.8x_{t-1} + \epsilon_t$), autocorrelation et partial autocorrelation

Example of AR_2

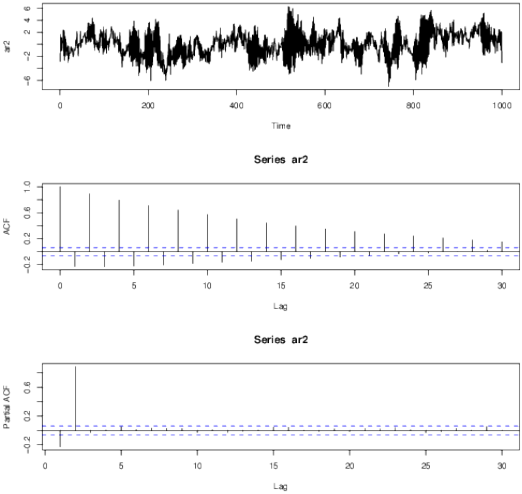


Figure 3: AR_2 ($x_t = 0.9x_{t-2} + \epsilon_t$), autocorrelation et partial autocorrelation

Example of AR_2

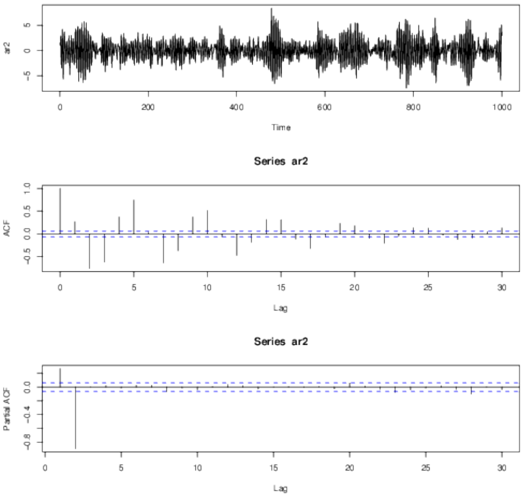


Figure 4: AR_2 ($x_t = -0.5x_{t-1} - 0.9x_{t-2} + \epsilon_t$), autocorrelation et partial autocorrelation

It's your turn!

Function `arima.sim` allows to simulate an AR_p.

Do it several times and observe the auto-correlations (partial or not)

```
par(mfrow=c(3,1))
modele<-list(ar=c(0.8))
ar1<-arima.sim(modele,1000)
plot.ts(ar1)
acf(ar1)
pacf(ar1)
```

Moving average models MA_q

A moving average model (x_t) of order q (MA_q) can be written:

$$X_t = c + \epsilon_t + b_1\epsilon_{t-1} + \ldots + b_q\epsilon_{t-q},$$

where ϵ_j for $t - q \leq j \leq t$ are white noises of variance σ^2.

Warning: Moving average models should not be confused with moving average smoothing...

MA_q properties

- autocorrelation $\rho(h)$ is null for all $h > q$:

$$\sigma(h) = \begin{cases} \sigma^2 \sum_{k=0}^{q-h} b_k b_{k+h} & \forall h \leq q \\ 0 & \forall h > q \end{cases} \quad \text{où } b_0 = 1$$

- partial autocorrelation exponentialy decreases to 0 when $h \to \infty$
- any AR_p can be seen as an MA_∞
- under some conditions on the b_j, an MA_q can be seen as an AR_∞

Example of MA_1

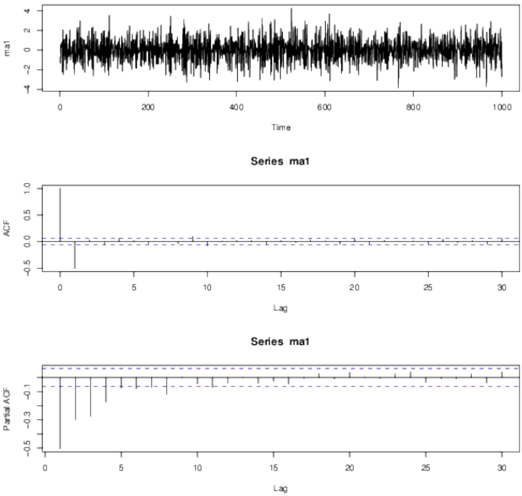


Figure 5: MA_1 ($x_t = \epsilon_t - 0.8\epsilon_{t-1}$), autocorrelation et partial autocorrelation

Example of MA_1

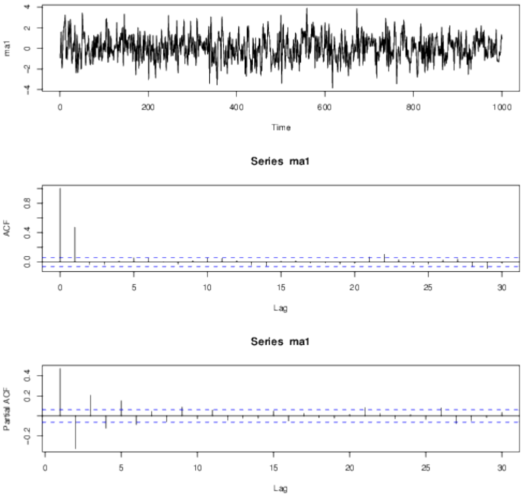


Figure 6: MA_1 ($x_t = \epsilon_t + 0.8\epsilon_{t-1}$), autocorrelation et partial autocorrelation

Example of MA_3

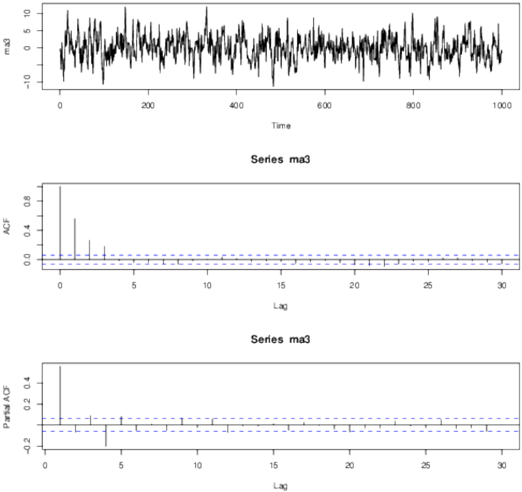


Figure 7: MA_3, autocorrelation et partial autocorrelation

It's your turn!

Function `arima.sim` allows to simulate an MA_q.

Do it several times and observe the auto-correlations (partial or not)

```
modele<-list(ma=c(0.8))
ma1<-arima.sim(modele,1000)
plot.ts(ma1)
acf(ma1)
pacf(ma1)
```

Autoregressive moving average model $ARMA_{pq}$

An autoregressive moving average model $ARMA_{pq}$ can be written:

$$x_t = c + \sum_{k=1}^{p} a_k x_{t-k} + \sum_{j=0}^{q} b_j \epsilon_{t-j}.$$

where ϵ_j for $t - q \leq j \leq t$ are white noise of variance σ^2.

Properties

- autocorrelation of an $ARMA_{p,q}$ exponentially descreases to 0 when $h \to \infty$, *from order* $q + 1$.

Example of $ARMA_{2,2}$

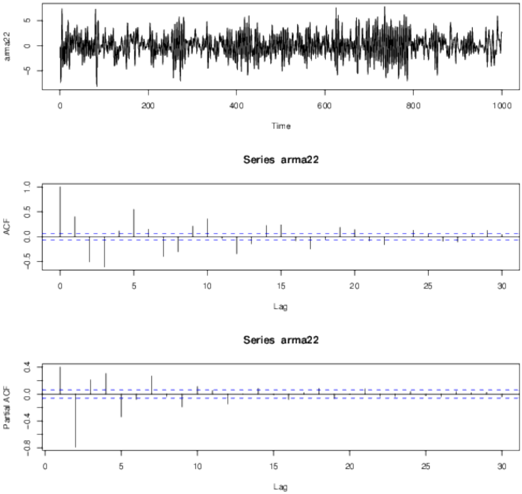


Figure 8: $ARMA_{2,2}$, autocorrelation et partial autocorrelation

Properties of MA_q, AR_p and $ARMA_{p,q}$

	MA_q	AR_p	$ARMA_{p,q}$
ACF	$\rho(h) = 0 \; \forall h > q$	$\lim_{h \to \infty} \rho(h) = 0$	$\forall h > q, \; \lim_{h \to \infty} \rho(h) = 0$
PACF	$\lim_{h \to \infty} r(h) = 0$	$r(h) = 0 \; \forall h > p$	
		et $r(p) = a_p$	

These properties *may* help to identify the order of a MA_q or an AR_p...

Non-seasonal ARIMA models

Non-seasonal ARIMA models

x_t **is an** $ARIMA_{p,d,q}$ **model if** $\Delta^d x_t$ **is an** $ARMA_{p,q}$ **model**
($\Delta^d x_t$ is x_t differenced d times)

ARIMA means *Auto Regressive Integrated Moving Average*

Selecting the orders p, d and q can be difficult.

Understanding ARIMA models

The intercept c of the model and the differencing order d have an important **effect on the long-term forecasts**:

- $c = 0$ and $d = 0 \Rightarrow$ long-term forcasts go to 0
- $c = 0$ and $d = 1 \Rightarrow$ long-term forcasts go to constant $\neq 0$
- $c = 0$ and $d = 2 \Rightarrow$ long-term forcasts will follow a straight line
- $c \neq 0$ and $d = 0 \Rightarrow$ long-term forcasts go to the mean of the data
- $c \neq 0$ and $d = 1 \Rightarrow$ long-term forcasts will follow a straight line
- $c \neq 0$ and $d = 2 \Rightarrow$ long-term forcasts will follow a quadratic trend

Some particular ARIMA models

- $ARIMA_{(0,1,0)}$ = random walk
- $ARIMA_{(0,1,1)}$ without constant = simple exponential smoothing
- $ARIMA_{(0,2,1)}$ without constant = linear exponential smoothing
- $ARIMA_{(1,1,2)}$ with constant = damped-trend linear exponential smoothing

Estimation

Once orders (p, d, q) are selected, **maximum likelihood estimation** (MLE) through optimization algorithms is used to estimate model parameters $\theta = (c, a_1, \ldots, a_p, b_1, \ldots, b_q)$

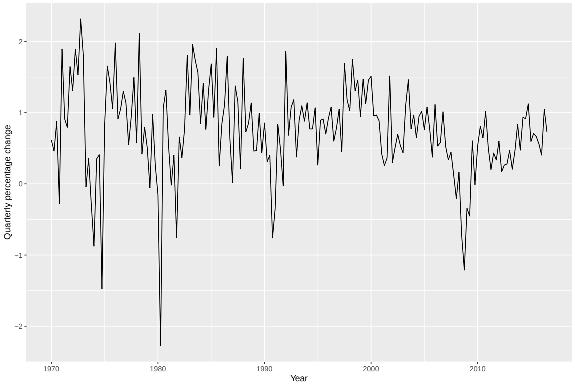
Model selection

- MLE can not be used to choose orders (p, d, q):
  higher are $(p, d, q) \Rightarrow$ higher is the number of parameters $\Rightarrow$ higher is the flexibility of the model $\Rightarrow$ higher is the likelihood
- MLE should be penalized by the complexity of the model ($\simeq$ number of parameters $\nu = p + q + 2$):
  - $AIC = -2 \log L(\hat{\theta}) + 2\nu$
  - $BIC = -2 \log L(\hat{\theta}) + \ln(n)\nu$
  - or for small sample size $AICc = AIC + \frac{2\nu(\nu+1)}{n-\nu-1}$
- or directly compute RMSE on test data

Example: US consumption expenditure

The following data contains quarterly percentage changes in US consumption expenditure

```
library(fpp2)
autoplot(uschange[,"Consumption"]) +
  xlab("Year") + ylab("Quarterly percentage change")
```



Example: US consumption expenditure

```r
Arima(uschange[,"Consumption"],order=c(2,0,2))
```

```
## Series: uschange[, "Consumption"]
## ARIMA(2,0,2) with non-zero mean
##
## Coefficients:
##          ar1      ar2      ma1     ma2    mean
##       1.3908  -0.5813  -1.1800  0.5584  0.7463
## s.e.  0.2553   0.2078   0.2381  0.1403  0.0845
##
## sigma^2 = 0.3511:  log likelihood = -165.14
## AIC=342.28   AICc=342.75   BIC=361.67
```

Warning: *the* `ar1` *parameter 1.3908 is the effect of* $(x_{t-1} - c)$ *on* x_t, *where* c *is the intercept of the model (*`mean`*).*

How to choose order (p, d, q) in practice

In practice, you have two choices, depending on your goal:

- to obtain quickly a good forecast, convenient if you have a lot of series to predict
  - let's use automatic function

```r
auto.arima(uschange[,"Consumption"])
```

```
## Series: uschange[, "Consumption"]
## ARIMA(1,0,3)(1,0,1)[4] with non-zero mean
##
## Coefficients:
##           ar1     ma1     ma2     ma3     sar1    sma1
##       -0.3548  0.5958  0.3437  0.4111  -0.1376  0.3834
## s.e.   0.1592  0.1496  0.0960  0.0825   0.2117  0.1780
##
## sigma^2 = 0.3481:  log likelihood = -163.34
## AIC=342.67   AICc=343.48   BIC=368.52
```

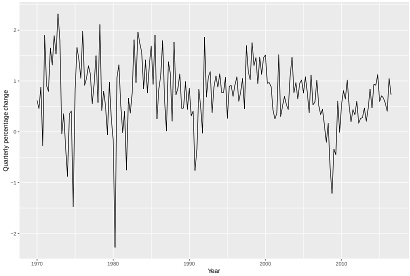
How to choose order (p, d, q) in practice

In practice, you have two choices, depending on your goal:

- to obtain a good forecast and an understanding of the model
  - let's start by differencing the series if needed, in order to obtain something visually stationary
  - look at the ACF and PACF plot ot identify possible models
  - take eventually into account knowledge on the series (knwon autocorrelation…)
  - estimate models and select the best one by AICc / AIC / BIC

Example: US consumption expenditure

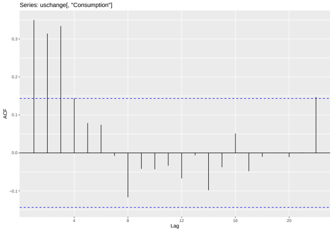
```
autoplot(uschange[,"Consumption"]) +
  xlab("Year") + ylab("Quarterly percentage change")
```



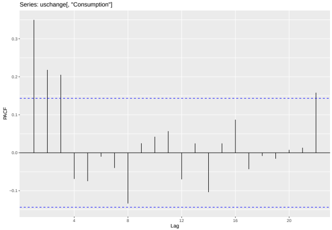
The series seems approximatively stationary. . .

Example: US consumption expenditure

```r
ggAcf(uschange[,"Consumption"])
```




```r
ggPacf(uschange[,"Consumption"])
```



May be an AR_3 or an MA_3

Example: US consumption expenditure

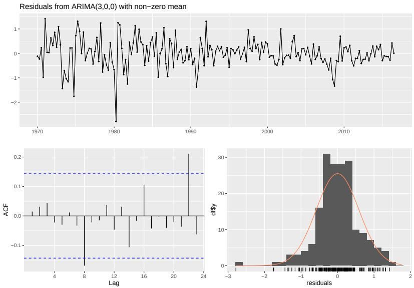
```r
Arima(uschange[,"Consumption"],order=c(3,0,0))
```

```
## Series: uschange[, "Consumption"]
## ARIMA(3,0,0) with non-zero mean
##
## Coefficients:
##          ar1     ar2     ar3    mean
##       0.2274  0.1604  0.2027  0.7449
## s.e.  0.0713  0.0723  0.0712  0.1029
##
## sigma^2 = 0.3494:  log likelihood = -165.17
## AIC=340.34   AICc=340.67   BIC=356.5
```

Example: US consumption expenditure

We check that residuals are un-correlated (LB test pvalue >0.05)

```
model=Arima(uschange[,"Consumption"],order=c(3,0,0))
checkresiduals(model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,0,0) with non-zero mean
## Q* = 6.7407, df = 5, p-value = 0.2407
##
```

Example: US consumption expenditure

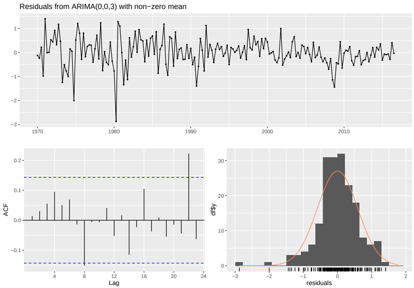
```r
Arima(uschange[,"Consumption"],order=c(0,0,3))
```

```
## Series: uschange[, "Consumption"]
## ARIMA(0,0,3) with non-zero mean
##
## Coefficients:
##          ma1     ma2     ma3    mean
##       0.2403  0.2187  0.2665  0.7473
## s.e.  0.0717  0.0719  0.0635  0.0739
##
## sigma^2 = 0.354:  log likelihood = -166.38
## AIC=342.76   AICc=343.09   BIC=358.91
```

Example: US consumption expenditure

The residuals are also uncorrelated

```
model=Arima(uschange[,"Consumption"],order=c(0,0,3))
checkresiduals(model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,0,3) with non-zero mean
## Q* = 8.5791, df = 5, p-value = 0.1271
##
```

Example: US consumption expenditure

- AIC criterion slightly better for AR_3 (340.34) than for MA_3 (342.76)
- Note that AICc for AR_3 is better than for the model chosen by `auto.arima`! That is because all the possible models are not tested, but a stepwise search is used (see Hyndman, p245)

Forecasting

Once the model is selected, it will be use to forecast the future of the series.

For an AR_p:

- forecasting at horizon $h = 1$:

  $$\hat{x}_{n+1} = \hat{c} + \hat{a}_1 x_n + \ldots + \hat{a}_p x_{n+1-p}$$

  95% prediction interval can be obtained by: $\pm 1.96 \hat{x}_{n+1}$

- forceasting at horizon $h = 2$:

  $$\hat{x}_{n+2} = \hat{c} + \hat{a}_1 \hat{x}_{n+1} + \hat{a}_2 x_n + \ldots + \hat{a}_p x_{n+2-p}$$

- and so on...

Forecasting

Once the model is selected, it will be use to forecast the future of the series.

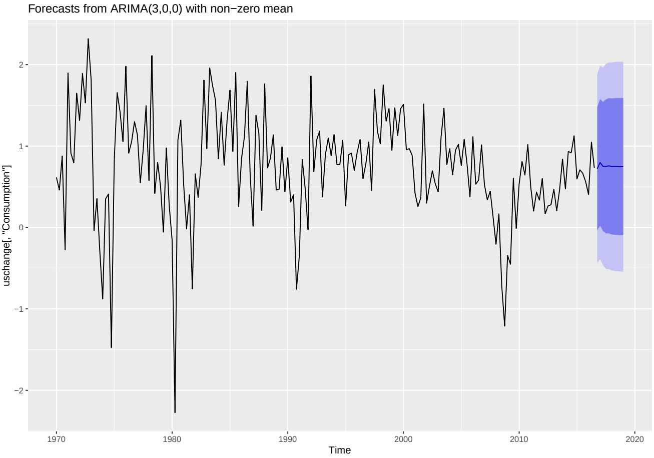
For an MA_q:

$$\hat{x}_{n+1} = \hat{c} + \hat{b}_1 \hat{\epsilon}_n + \ldots + \hat{b}_q \hat{\epsilon}_{n+1-q}$$

where $\hat{\epsilon}_n = x_n - \hat{x}_n$ and $\hat{\epsilon}_{n+1-q} = x_{n+1-q} - \hat{x}_{n+1-q}$

Example: US consumption expenditure

```r
fit=Arima(uschange[,"Consumption"],order=c(3,0,0))
autoplot(forecast(fit,h=10))
```



Exercice: uschange

The following time series contain percentage changes in personal disposable income and unemployment rate for the US, from 1960 to 2016.

```
autoplot(uschange[,c("Income","Unemployment")])
```

Choose an ARIMA model and forecast the income and unemployment rate for 2017 to 2020.

Seasonal ARIMA models

Backshift notation

A convenient notation for ARIMA models is **backshift notation**:

$$\begin{aligned} Bx_t &= x_{t-1} \\ B(Bx_t) &= B^2 x_t = x_{t-2} \end{aligned}$$

With this notation:

$$\begin{aligned} \Delta x_t &= (1-B)x_t = x_t - x_{t-1} \\ \Delta_T x_t &= (1-B^T)x_t = x_t - x_{t-T} \\ \Delta^d x_t &= (1-B)^d x_t \\ \Delta_T^d x_t &= (1-B^T)^d x_t \end{aligned}$$

Backshift notation

The backshift notation of an $ARIMA_{p,d,q}$ model is:

$$\underbrace{(1 - a_1 B - \ldots - a_p B^p)}_{AR_p} \underbrace{(1 - B)^d x_t}_{d \text{ differences}} = c + \underbrace{(1 + b_1 B - \ldots + b_q B^q)}_{MA_q} \epsilon_t$$

For instance, an $ARIMA_{1,1,1}$ without constant model is:

$$(1 - a_1 B)(1 - B) x_t = (1 + b_1 B) \epsilon_t$$

Rk: R uses a slightly different parametrization (see Hyndman p237)

Seasonal ARIMA models

A seasonnal ARIMA (SARIMA) model is formed by including additional seasonal terms in an ARIMA:

$$ARIMA \quad \underbrace{(p,d,q)}_{\text{non-seasonnal part}} \quad \underbrace{(P,D,Q)_T}_{\text{seasonnal part}}$$

where T is the period of the seasonal part.

Corresponding backshift notations is, for an $SARIMA_{(1,1,1)(1,1,1)_{12}}$ without constant model is:

$$(1-a_1B)(1-a_2B^{12})(1-B)(1-B^{12})x_t = (1+b_1B)(1+b_2B^{12})\epsilon_t$$

SARIMA properties

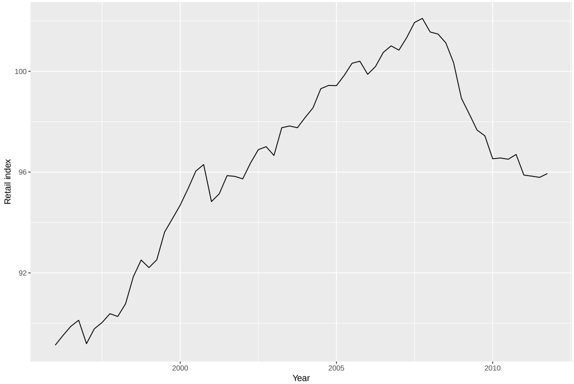
The seasonal part of an AR or MA model can be seen in the seasonal lags of the PACF and ACF.

For instance:

- an $SARIMA_{(0,0,0)(0,0,1)_{12}}$ will show:
  - a spike at lag 12 in the ACF, and no other significant spikes
  - exponential decay in the seasonal lags of the PACF (i.e. at lag 12, 24, 36. . .)
- an $SARIMA_{(0,0,0)(1,0,0)_{12}}$ will show:
  - a spike at lag 12 in the PACF, and no other significant spikes
  - expoenntial decay in the seasonal lags of the ACF

Example: European quaterly retail trade

```r
autoplot(euretail) + ylab("Retail index") + xlab("Year")
```



This time series is clearly non stationary: trend an probably seasonal pattern of period 4 (*quaterly retrail trade...*)

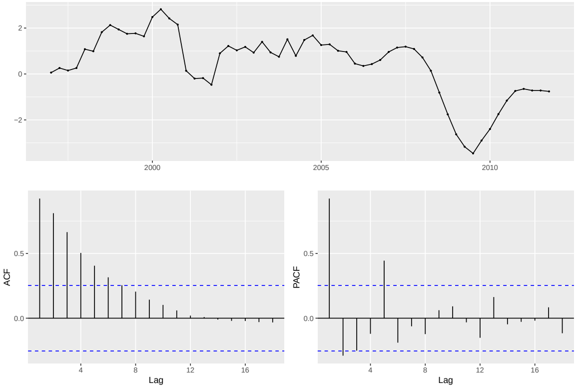
Example: European quaterly retail trade

Let's differenciate

```
ggtsdisplay(diff(euretail,lag=4))
```

or equivalently

```
euretail %>% diff(lag=4) %>% ggtsdisplay()
```

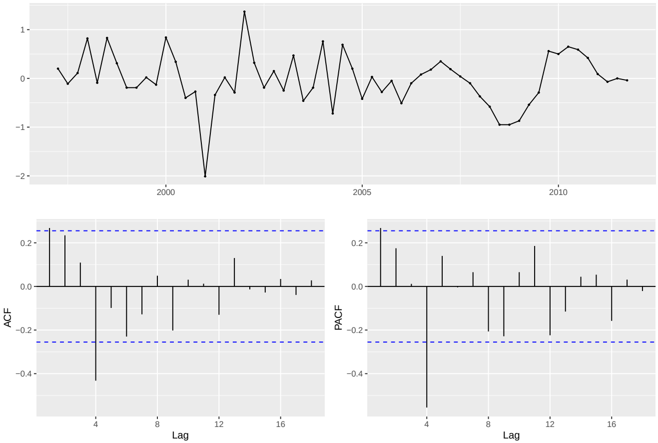


The linear decay of the ACF suggests that there is still a trend

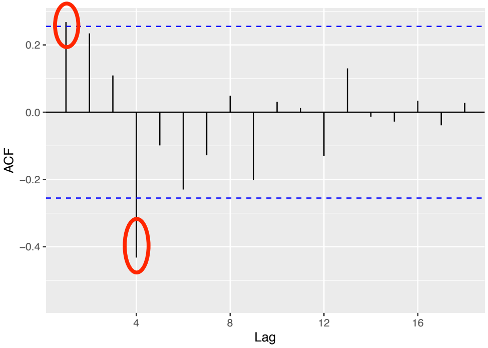
Example: European quaterly retail trade

Let's differenciate again

```
euretail %>% diff(lag=4) %>% diff()  %>% ggtsdisplay()
```



Example: European quaterly retail trade



- the slightly significant ACF at lag 1 suggests a non-seasonnal MA_1
- the significant ACF at lag 4 (the size of the period) suggests a seasonnal MA_1

Consequently we can try an $SARIMA_{(0,1,1)(0,1,1)_4}$.

Rk: similar reasoning with PACF suggests $SARIMA_{(1,1,0)(1,1,0)_4}$

Example: European quaterly retail trade

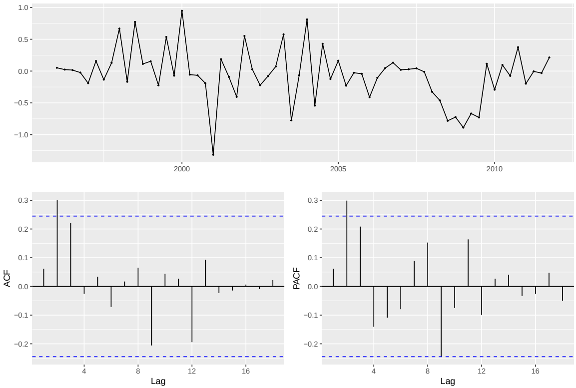
Let's estimate an $SARIMA_{(0,1,1)(0,1,1)_4}$

```
fit=Arima(euretail, order=c(0,1,1), seasonal=c(0,1,1))
```

Example: European quaterly retail trade

Let's have a look to the residual

```
fit %>% residuals() %>% ggtsdisplay()
```

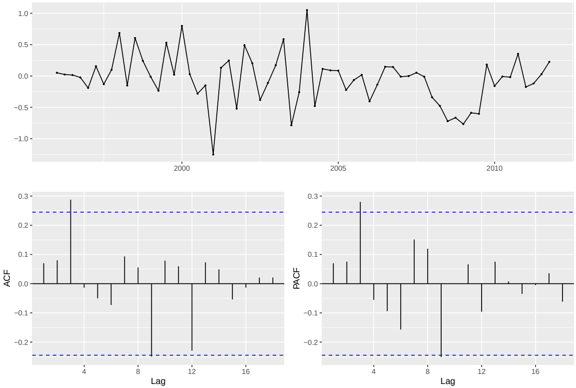


There is still significant ACF and PACF at lag 2. We can add some additional non-seasonal terms (for instance with $SARIMA_{(0,1,2)(0,1,1)_4}$)

Example: European quaterly retail trade

Let's estimate an $SARIMA_{(0,1,2)(0,1,1)_4}$

```
euretail %>%
  Arima(order=c(0,1,2), seasonal=c(0,1,1)) %>%
  residuals() %>% ggtsdisplay()
```

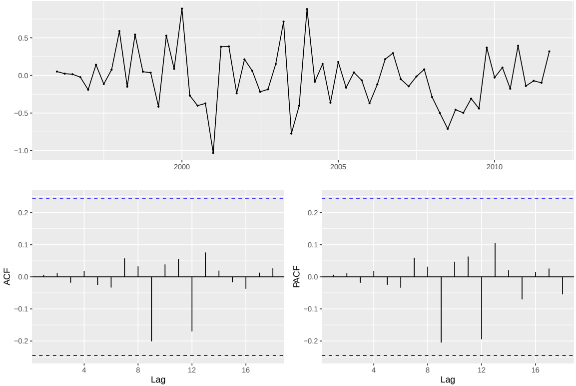


There is still significant ACF and PACF at lag 3.

Example: European quaterly retail trade

Let's estimate an $SARIMA_{(0,1,3)(0,1,1)_4}$

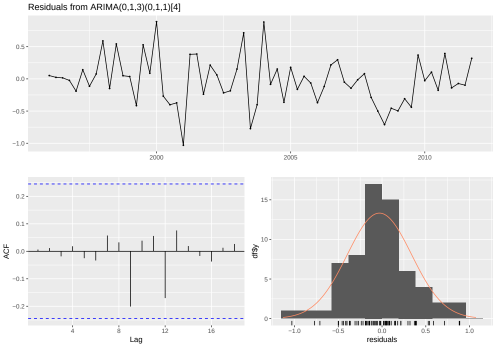
```
fit=Arima(euretail, order=c(0,1,3), seasonal=c(0,1,1))
fit %>% residuals() %>% ggtsdisplay()
```



Now the model seems to have capture all auto-correlations.

Example: European quaterly retail trade

```
checkresiduals(fit)
```

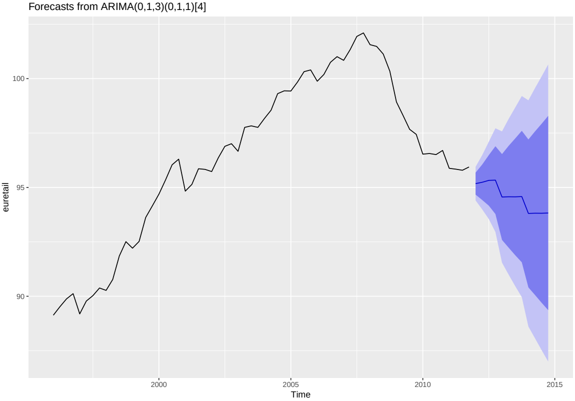



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,3)(0,1,1)[4]
## Q* = 0.51128, df = 4, p-value = 0.9724
##
## Model df: 4.   Total lags used: 8
```

Example: European quaterly retail trade

The model passes all checks: it is ready for forecasting

```
fit %>% forecast(h=12) %>% autoplot()
```



Exercice: San Francisco precipitation

San Fransisco precipitation from 1932, January to 1966, December are available here:
http://eric.univ-lyon2.fr/jjacques/Download/DataSet/sanfran.csv

- Try to improve your forecast obtained with exponential smoothing

Exercice: Varicella dataset

- Try to improve your forecast obtained with exponential smoothing

Heteroscedastic series

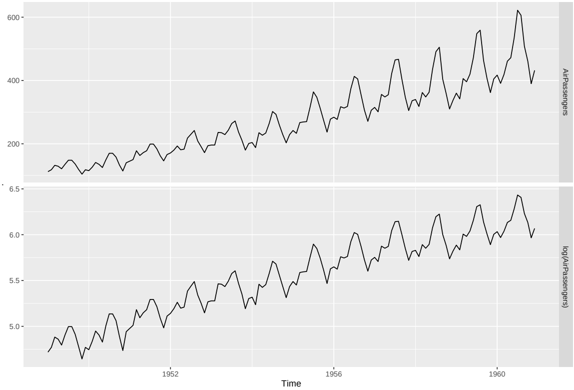
Stabilizing the variance

Previous models assume that the variance is stable in time.

For some series variance can decrease or increase.

Taking the log can help to stabilize it.

```
cbind(AirPassengers,log(AirPassengers)) %>%
autoplot(facets=TRUE)
```



Stabilizing the variance

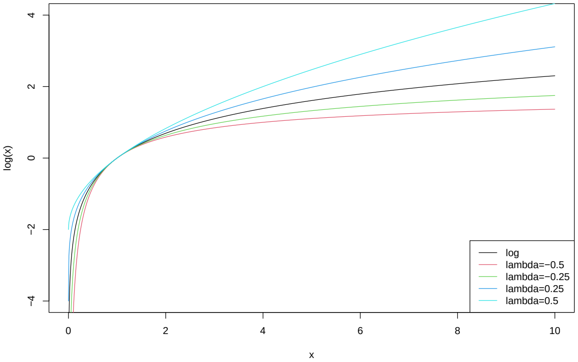
Rather than log transformation we can also use power transformation (square roots...).

A more general method for stabilizing the variance is to use Box-Cox transformation:

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

Box-Cox transformation

```
x=seq(0,10,0.01)
plot(x,log(x),type='l',ylim=c(-4,4))
lambda=-0.5;lines(x,(x^lambda-1)/lambda,col=2)
lambda=-0.25;lines(x,(x^lambda-1)/lambda,col=3)
lambda=0.25;lines(x,(x^lambda-1)/lambda,col=4)
lambda=0.5;lines(x,(x^lambda-1)/lambda,col=5)
legend('bottomright',col=1:5,lty=1,legend=c('log','lambda=-0.5',
```



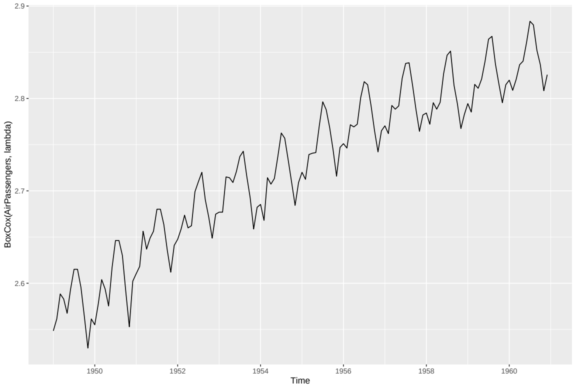
Stabilizing the variance

The `BoxCox.lambda()` function will choose a value of λ for you

```
(lambda=BoxCox.lambda(AirPassengers))
```

```
## [1] -0.2947156
```

```
autoplot(BoxCox(AirPassengers,lambda))
```



Stabilizing the variance

The BoxCox transformation is available as an option in the `hw` or `auto.arima` functions.

Automatic choice of λ is obtained by selecting: `lambda="auto"`.

ARCH and GARCH models

Such techniques allows to stabilize a variance which monotically increases or decreases.

For more complexe variations of the variance, as it can be in financial series, specific models for non constant variance exist:

- **ARCH: autoregressive conditional heteroscedasticity**
- and their generalization **GARCH**

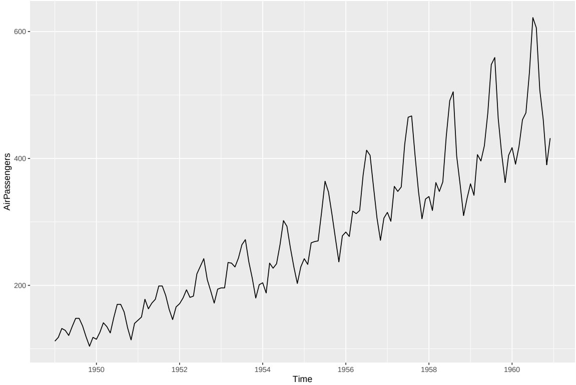
For more details refer to:

Brockwell P.J. et Davis R.A. Introduction to Time Series and Forecasting, Springer, 2001.

AirPassengers

Try to obtain the best model (exponential smoothing, SARIMA) for the AirPassengers data.

```r
autoplot(AirPassengers)
```



The models will be evaluated on a test set made up of the last two years.