Applied Time Series Econometrics

Oxana Malakhovskaya, NRU HSE

September 17, 2019

Working with time series

The algorighm of working with times series models.

1. Importing data that we will do with `readr` or `readxl` packages.
2. Data visualisation that we will do with `ggplot2` package (if necessary).
3. Data manipulation that we will do with `dplyr` package treating series as data frames.
4. Transformation data frames into special time series (ts) format if necessary (some packages work only with ts data).
5. Estimation and evaluation of models with special packages developed specially for a certain class of models.

Times Series Formats

- Data frame is a principal object in R to hold data. However, to use special packages for times series models we have to transform our dataset to a `ts` format.

Three times series formats (among others) are best-known:

1. ts: a data format that allows the user to work with regular times series.
2. zoo: a data format that allows the user to work also with irregular times series, and a package that introduces this data format.
3. xts: a data format that that allows the user to work also with irregular times series with numerous observations for a time point, also a package that introduces this data format.

`zoo` extends `ts`. `xts` extends `zoo`.

TS class

- To create a time series object of the `ts` class we use the `ts()` function. The arguments of the `ts()` function include:

data: a vector or a matrix of the observed time serries.

start: the time of the first observation

end: the time of the last observation (may be dropped if `start` and `freq` are indicated)

frequency: the number of observation per unit of time

Converting data frames into a ts object

- If we need to convert a data frame into a `ts` object we have to estract the data matrix first.

```
# read_csv function is from readr package (tidyverse set)
rus_data <- read_csv("rus_data.csv")
class(rus_data)
rus_data <- select(rus_data, -time)
rus_data_ts <- ts(rus_data, start = c(1995, 1), freq = 12)
class(rus_data_ts)
```

Stationary univariate processes

Stationarity:

A stochastic process y_t is called (weakly) *stationary* if it has time-invariant first and second moments:

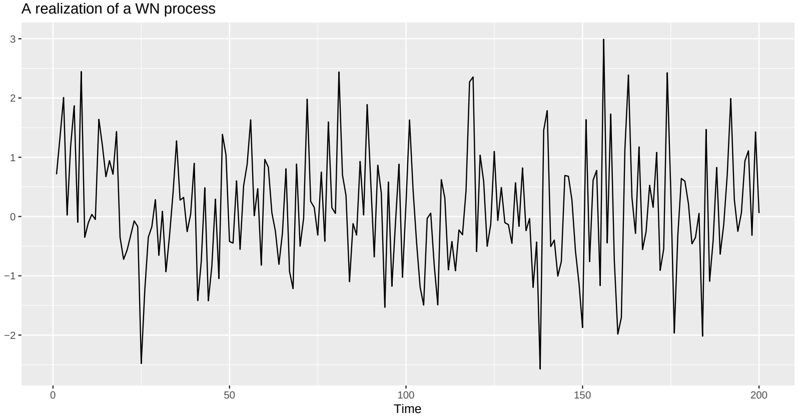
$$\mathbb{E}(y_t) = \mu_y \forall t \in T$$
$$cov(y_t, y_{t-h}) = \gamma_h$$
$$\forall t \in T \text{ and } \forall h \text{ such as } t-h \in T$$

The simplest stationary process is white noise (WN):

Stochastic process y_t is a WN process if:

1. $\mathbb{E}(y_t) = 0$
2. $var(y_t)$ is const
3. $cov(y_t, y_{t-h}) = 0 \quad \forall t \in T$ and $\forall h$ such as $t-h \in T$

WN graph



Wold's Representation Theorem

First of all, we are going to work with ARMA models. Why?

Wold's Representation Theorem

If y_t is a weakly stationary process, then it can be represented as:

$$y_t = \mu_t + \sum_{j=0}^{\infty} \psi_j \nu_{t-j}$$

where μ_t is deterministic time series (possibly a constant), ν_t is white noise and $\sum_{j=0}^{\infty} \psi_j^2 < \infty$ The infinite sum is understood as the mean square limit of $\sum_{j=0}^{n} \psi_j \nu_{t-j}$ as $n \to \infty.\}$

We need a model with a finite number of parameters that can be equivalent to the weighted infinite sum of WN process.

ARMA models

Let y_t be a time series variable. Consider a model so that y_t is fully determined by its own dynamics:

$$y_t = \mu + \sum_{i=1}^{p} \phi_i y_{t-i} + \nu_t + \sum_{i=1}^{q} \psi_i \nu_{t-i}$$

This model represents the univariate class of linear autoregressive moving average models with p AR lags and q MA lags, ARMA(p,q).

Some special cases of ARMA(p,q) model are:

$$\begin{array}{ll}
ARMA(0,0) = \text{WN with a constant:} & y_t = \mu + \nu_t \\
ARMA(1,0) = AR(1): & y_t = \mu + \phi_1 y_{t-1} + \nu_t \\
ARMA(2,0) = AR(2): & y_t = \mu + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \nu_t \\
ARMA(0,1) = MA(1): & y_t = \mu + \nu_t + \psi_1 \nu_{t-1} \\
ARMA(0,2) = MA(2): & y_t = \mu + \nu_t + \psi_1 \nu_{t-1} + \psi_2 \nu_{t-2} \\
ARMA(1,1) & y_t = \mu + \phi_1 y_{t-1} + \nu_t + \psi_1 \nu_{t-1},
\end{array}$$

where ν_t is an iid process.

Lag operators

The ARMA(p,q) model can be rewritten in terms of lag polynomials.

$$
\begin{aligned}
y_{t-1} &= Ly_t \\
y_{t-2} &= L(Ly_t) = L^2 y_t \\
&\ldots\ldots \\
y_{t-k} &= L(L^{k-1} y_t) = L^k y_t
\end{aligned}
$$

$$\text{Then:} y_t = \mu + \sum_{i=1}^{p} \phi_i y_{t-i} + \nu_t + \sum_{i=1}^{q} \psi_i v_{t-i} \Leftrightarrow$$

Lag operators

The ARMA(p,q) model can be rewritten in terms of lag polynomials.

$$
\begin{aligned}
y_{t-1} &= Ly_t \\
y_{t-2} &= L(Ly_t) = L^2 y_t \\
&\ldots\ldots \\
y_{t-k} &= L(L^{k-1}y_t) = L^k y_t
\end{aligned}
$$

$$\text{Then:} y_t = \mu + \sum_{i=1}^{p} \phi_i y_{t-i} + \nu_t + \sum_{i=1}^{q} \psi_i v_{t-i} \Leftrightarrow$$

$$y_t = \mu + \phi_1 L y_t + \ldots \phi_p L^p y_t + \nu_t + \psi_1 L \nu_t + \ldots + \psi_q L^q \nu_t$$

$$(1 - \phi_1 L - \ldots - \phi_p L^p) y_t = (1 + \psi_1 L + \ldots + \psi_q L^q) \nu_t$$

Lag operators

The ARMA(p,q) model can be rewritten in terms of lag polynomials.

$$
\begin{aligned}
y_{t-1} &= Ly_t \\
y_{t-2} &= L(Ly_t) = L^2 y_t \\
&\ldots\ldots \\
y_{t-k} &= L(L^{k-1} y_t) = L^k y_t
\end{aligned}
$$

$$\text{Then:} y_t = \mu + \sum_{i=1}^{p} \phi_i y_{t-i} + \nu_t + \sum_{i=1}^{q} \psi_i v_{t-i} \Leftrightarrow$$

$$y_t = \mu + \phi_1 L y_t + \ldots \phi_p L^p y_t + \nu_t + \psi_1 L \nu_t + \ldots + \psi_q L^q \nu_t$$

$$(1 - \phi_1 L - \ldots - \phi_p L^p) y_t = (1 + \psi_1 L + \ldots + \psi_q L^q) \nu_t$$

$$\phi_p(L) y_t = \mu + \psi_q(L) \nu_t, \qquad \nu_t \sim iidN(0, \sigma_\nu^2),$$

$$\text{where: } \phi_p(L) = 1 - \phi_1 L - \ldots - \phi_p L^p$$

$$\psi_q(L) = 1 + \psi_1 L + \ldots + \psi_q L^q$$

Stability

The process given by the equation

$$y_t = \mu + \sum_{i=1}^{p} \phi_i y_{t-i} + \nu_t + \sum_{i=1}^{q} \psi_i \nu_{t-i}$$

is stable provided that the roots of the polynomial $\phi_p(z) = 0$ lie outside the unit circle, i.e.if :

$$\phi(z) \neq 0 \quad \text{for } |z| \leq 1$$

If the process is stable, it has a pure (possibly infinite order) MA representation.

Example for AR(1)

$$y_t = \mu + \phi_1 y_{t-1} + \nu_t$$
$$y_t - \phi_1 y_{t-1} = \mu + \nu_t$$
$$y_t - \phi_1 L y_t = \mu + \nu_t$$
$$(1 - \phi_1 L) y_t = \mu + \nu_t$$

The relevant polynomial $\phi_p(L) = 1 - \phi_1 L$ is

$$1 - \phi_1 z = 0$$

The only root is given by $z_1 = \phi_1^{-1}$. So if $|\phi_1| < 1$ than y_t is *stable*. We will take the condition into account while simulating models.

ARMA models: simulation

Simulation of a model from the ARMA family is done with the `arima.sim()`function from preloaded package `stats`.

```
#Simulating AR(1) model
ar1 <- arima.sim(model = list(ar = 0.8,
                 order = c(1,0,0)), n = 200)
#Simulating ARMA(1,1) model
arma11 <- arima.sim(model = list(ar = 0.4, ma = 0.5,
                 order = c(1,0,1)), n = 200)
#Simulating MA(2) model
ma2 <- arima.sim(model = list(ma = c(0.5, -0.3),
                 order = c(0,0,2)), n = 200)
```

- `i` in arima.sim means the order of integration. The order of integration is zero for stationary series, this explains why the second element in all order vectors is equal to zero: c(0,0,2).

ARMA models: packages

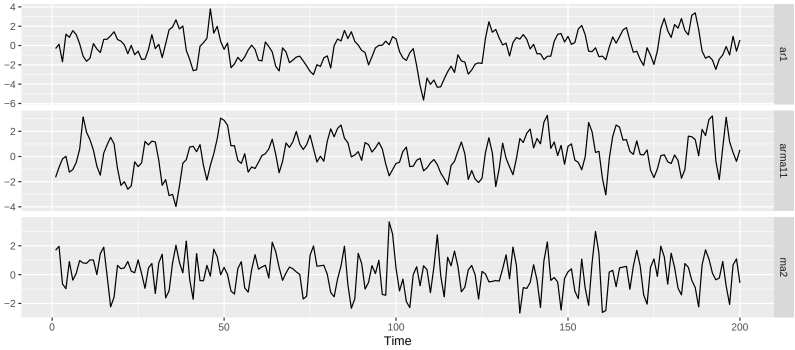
- Though there exist default functions for manipulating time series in preloaded package `stat`, we will use a special package for forecasting time series.

```
install.packages("forecast")
library(forecast)
```

Time series visualisation

How the series we simulated look like? Use `autoplot()` function from the `ggplot2`package.

```
autoplot(cbind(ar1, arma11,ma2), facets = TRUE) + ylab('')
```



ACF and PACF: interpretation

- To understand the dynamic properties of the series we compute the autocorrelation function (ACF) and partial autocorrelation function (PACF) of the variables.
- The ACF and PACF are the parameter estimates on the explanatory variable y_{t-k} in each of the following regressions:

ACF: Regress y_t on $\{const, y_{t-k}\}$

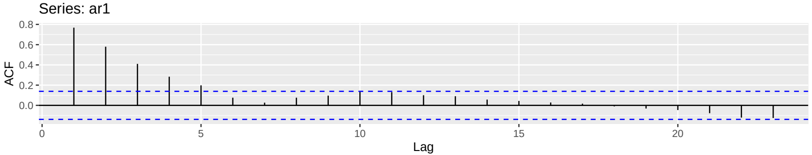
PACF: Regress y_t on $\{const, y_{t-1}, \cdots\ y_{t-k}\}$

- ACF is also a sample $\widehat{corr}(y_t, y_{t-k})$. PACF is a sample $\widehat{corr}(u_{1,t}, u_{2,t})$, where $u_{1,t}$ are residuals in a regression of y_t on $y_{t-1}, \ldots y_{t-k+1}$ and $u_{2,t}$ are residuals in a regression of y_{t-k} on $y_{t-1}, \ldots y_{t-k+1}$

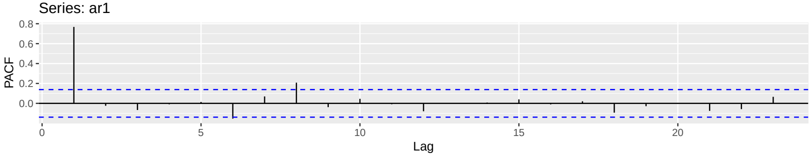
- To compute and plot ACF and PACF functions, we use the `ggAcf()` and `ggPacf()` functions.

ACF and PACF for AR(1) model

```r
ggAcf(ar1)
```



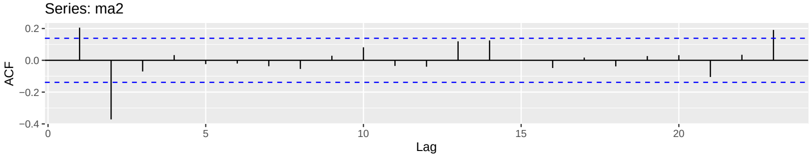

```r
ggPacf(ar1)
```



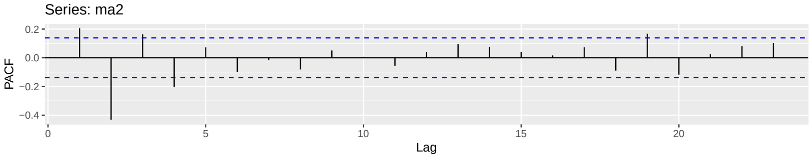
ACF exhibits damped oscillatory behavior, while PACF shows one spike for the first lag.

ACF and PACF for MA(2)

```r
ggAcf(ma2)
```



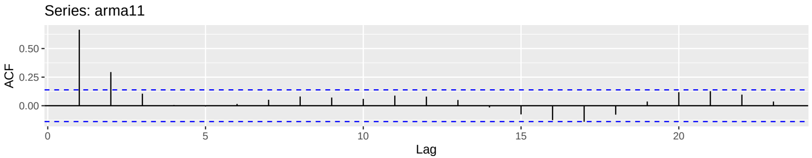

```r
ggPacf(ma2)
```



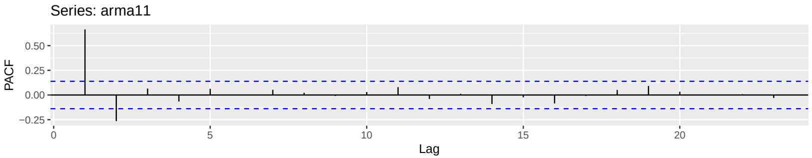
ACF shows two spikes for the first two lags and, while PACF exhibits damped oscillatory behavior.

ACF and PACF for ARMA(1,1)

```
ggAcf(arma11)
```




```
ggPacf(arma11)
```



The graph shows both a damped ACF and PACF.

Confidence intervals

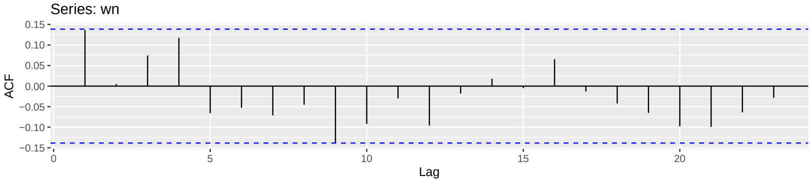
Sample autocorrelation is equal to $\hat{\rho}_j = \frac{\widehat{cov}(y_t, y_{t-j})}{\widehat{var}(y_t)}$.

If y_t is iid, $\sqrt{T}\hat{\rho}_j \to^{\text{d}} N(0,1) \Rightarrow \left(-\frac{1.96}{\sqrt{T}}, \frac{1.96}{\sqrt{T}}\right)$ is a 95-% confidence interval.

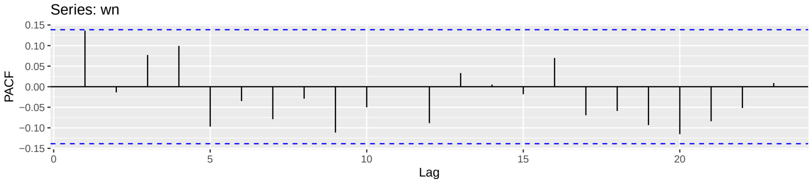
- If all the ACF and PACF values are inside of the confidence interval, then the series can be considered as WN process.
- Particular case of white noise: $y_t = \mu + \nu_t$, where ν_t is $iid(0, \sigma^2)$

White noise

```r
wn <- arima.sim(model = list(order = c(0,0,0)), n = 200)
ggAcf(wn)
```




```r
ggPacf(wn)
```



Ljung-Box test

To test if a series can be considered as WN, the Ljung-Box test is used. The test considers k first autocorrelation together.

H_0: All $\rho_1 = \rho_2 = \cdots = \rho_k = 0$

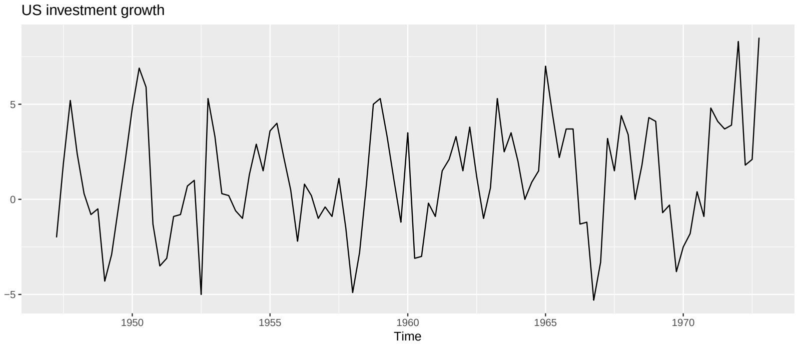
H_1: otherwise

To do the Ljung-Box test we use the `Box.test()` function

US investment growth: importing and plotting data

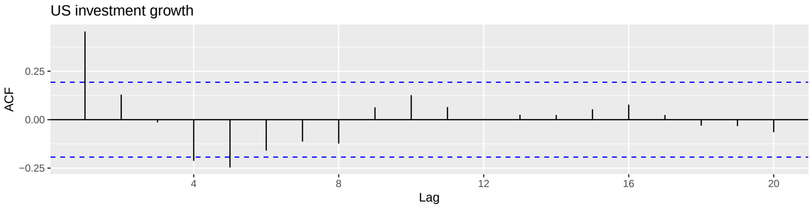
```
US_inv <- read_csv("US_investment.dat")
US_inv_ts <- ts(US_inv, start = c(1947, 2), freq = 4)
autoplot(US_inv_ts) + ylab("") +
        ggtitle("US investment growth")
```

US investment growth (quarterly SA data)

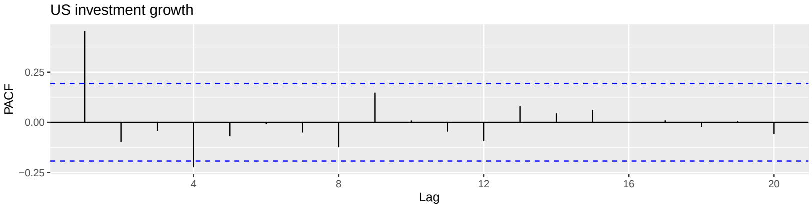


ACF and PACF for US investment growth

```r
ggAcf(US_inv_ts) + ggtitle("US investment growth")
```




```r
ggPacf(US_inv_ts) + ggtitle("US investment growth")
```



Ljung - Box test for US investment growth series

```r
Box.test(US_inv_ts, lag = 24, fitdf = 0, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  US_inv_ts
## X-squared = 49.776, df = 24, p-value = 0.001512
```

Estimaton(1)

- The parameters of the ARMA models are estimated by maximum likelihood methods.
- In a special case when the moving average terms are absent, the maximum likelihood estimates are obtained by OLS.
- To estimate a certain ARMA model in R we use the `Arima()` function.

```
Arima(US_inv_ts, order = c(2,0,3),
      include.constant = TRUE)
```

Estimaton(2)

```
## Series: US_inv_ts
## ARIMA(2,0,3) with non-zero mean
##
## Coefficients:
##          ar1      ar2     ma1     ma2     ma3    mean
##       0.1533  -0.7661  0.3603  0.9236  0.5100  1.1460
## s.e.  0.1062   0.0733  0.1155  0.0606  0.1048  0.4123
##
## sigma^2 estimated as 6.253:  log likelihood=-239.63
## AIC=493.27   AICc=494.45   BIC=511.71
```

Estimaton(3)

- All the parameter estimate appear to be significant. Do we need more lags?

```
Arima(US_inv_ts, order = c(4,0,4),
      include.constant = TRUE)
```

```
Arima(US_inv_ts, order = c(5,0,5),
      include.constant = TRUE)
```

Estimaton(4)

```
## Series: US_inv_ts
## ARIMA(4,0,4) with non-zero mean
##
## Coefficients:
##          ar1      ar2     ar3      ar4      ma1     ma2
##       0.7719  -0.7940  0.9219  -0.4783  -0.2579  0.6688
## s.e.  0.1857   0.1375  0.1455   0.1541   0.2115  0.1708
##         mean
##       1.0837
## s.e.  0.2648
##
## sigma^2 estimated as 5.994:  log likelihood=-236.45
## AIC=492.91   AICc=495.3   BIC=519.25
```

Estimaton(5)

```
## Series: US_inv_ts
## ARIMA(5,0,5) with non-zero mean
##
## Coefficients:
##           ar1      ar2     ar3     ar4      ar5     ma1
##       -0.1497  -0.0537  0.1554  0.3808  -0.4799  0.6746
## s.e.   0.2060   0.1820  0.1720  0.1533   0.1439  0.2362
##           ma4      ma5    mean
##       -0.7427  -0.0405  1.0806
## s.e.   0.2422   0.2174  0.2628
##
## sigma^2 estimated as 5.994:  log likelihood=-235.94
## AIC=495.88   AICc=499.35   BIC=527.5
```

Lag order choice

- We do not know a priori how many lags to choose.
- A common data-driven way to select lags is to use information criteria (IC).
- `Arima()` provides three IC: AIC (Akaike), AICc (corrected AIC), and BIC (Bayes).

```
model1 <- Arima(US_inv_ts, order = c(5,0,5),
      include.constant = TRUE)
model1$aic
```

```
## [1] 495.8797
```

```
model1$aicc
```

```
## [1] 499.3464
```

```
model1$bic
```

```
## [1] 527.4965
```

Lag order choice(2)

- If the disturbance term is assumed to be normally distributed, these IC are computed as folows:

$$AIC = -2\ln\ell + 2k \tag{1}$$

$$AICc = AIC + \frac{2k(k+1)}{T-s-k-1} \tag{2}$$

$$BIC = -2\ln\ell + kln(T-s), \tag{3}$$

where k is a number of model parameters, $s = max\{p_{max}, q_{max}\}$, and $\ln\ell$ is log-likelihood.

Algorithm of choosing number of lags

1. Choose a maximum number of lags for the AR and MA parts of the model. The choice of p_{max} and q_{max} is governed by ACFs and PACFs, the data frequency and the sample size.
2. To estimate the models with all possible combinations of $p = 0, \ldots, p_{max}$ and $q = 0, \ldots, q_{max}$.
3. To choose a model that corresponds to the minimum values of IC. In case of disagreement between different IC, the final decision is matter of judgement.

Automatic ARIMA

Q: Can we speed up the lag order selection?

A: Yes, we can make usage of the auto.arima() function

```
auto.arima(US_inv_ts, d = 0, seasonal = FALSE)
```

```
## Series: US_inv_ts
## ARIMA(2,0,1) with non-zero mean
##
## Coefficients:
##          ar1      ar2      ma1    mean
##       1.2991  -0.5034  -0.7913  1.0871
## s.e.  0.1550   0.0954   0.1566  0.2630
##
## sigma^2 estimated as 6.636:  log likelihood=-241.79
## AIC=493.59   AICc=494.21   BIC=506.76
```