Time-Series Modelling

Ralf Becker

April 2021

```
library(tidyverse)
```

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

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

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

```
library(ggplot2)
library(pdfetch)
library(xts)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
library(AER)          # access to HS robust standard errors
```

```
## Loading required package: car
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

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

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
## Loading required package: lmtest
## Loading required package: sandwich
## Loading required package: survival
```

```
library(stargazer)
```

```
##
## Please cite as:
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
source("stargazer_HC.r")  # includes the robust regression display
source("stargazer_HAC.r")  # includes the Newey-West standard errors
library(gridExtra)  # required for the combination of ggplots
```

```
##
## Attaching package: 'gridExtra'
## The following object is masked from 'package:dplyr':
##
##     combine
```

Import data and look at ACF

When you get data from the ONS you need to know what the series id is and in which ONS dataset you can find these data. The way to find out what that info is it is best to go to (the ONS time series tool)[https://www.ons.gov.uk/timeseriestool].

For instance we can get real GDP data (id: AMBI) from the UKEA databank.

```
rGDP <- pdfetch_ONS("ABMI","UKEA")
periodicity(rGDP)   # check data availability
```

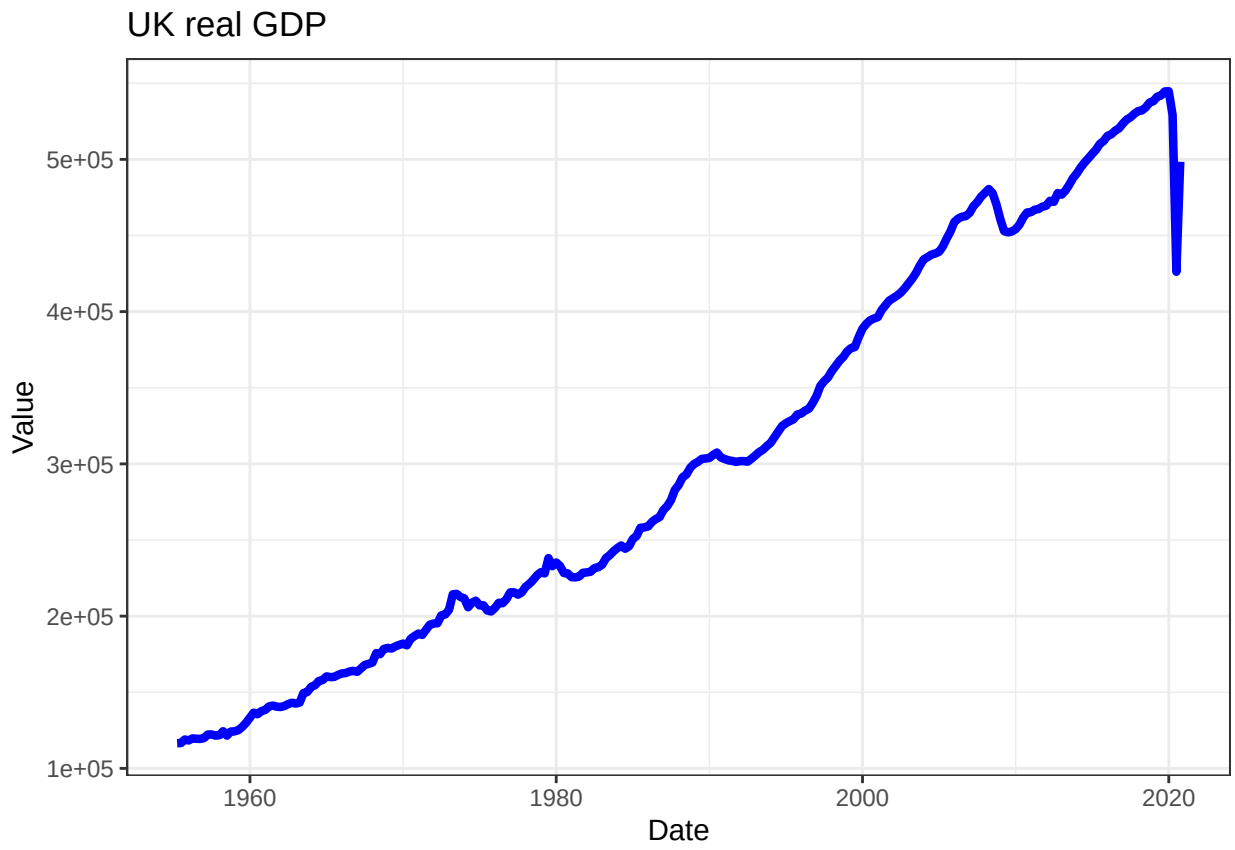
```
## Quarterly periodicity from 1955-03-31 to 2020-12-31
```

```
names(rGDP) <- "real GDP" # give a sensible name

# keep all the data including 2020-Q3
# this was the last observation available at the time this was written
# remove this line if you want to use updated data
rGDP <- rGDP["/2020-9"]

# we prepare the data for being kept in long format
# that is useful for plotting in ggplot
rGDP_l <- data.frame(index(rGDP),stack(as.data.frame(coredata(rGDP))))
# Give sensible names to columns
names(rGDP_l)[1] <- "Date"   # first col will have date
names(rGDP_l)[2] <- "Value"  # second col will have value
names(rGDP_l)[3] <- "id"     # third col will have series name
```

Let's look at the series

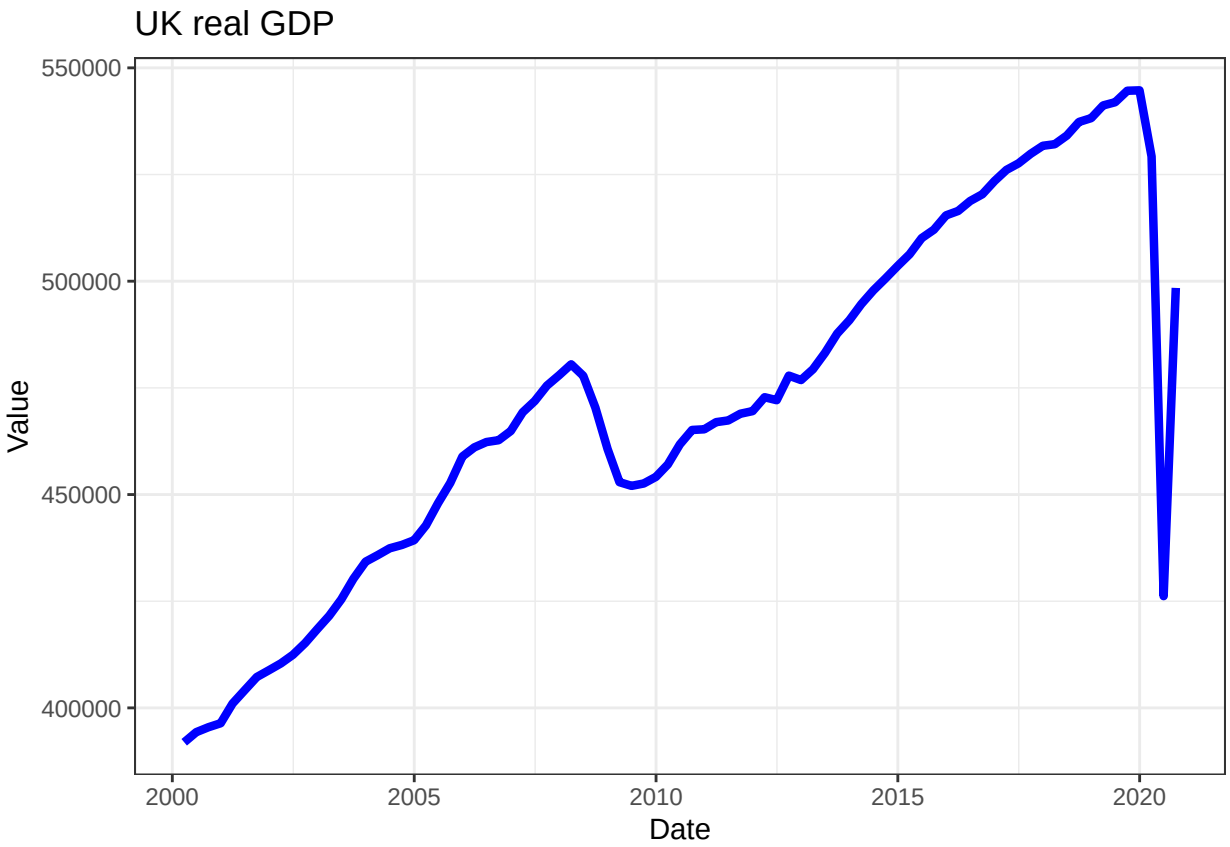
```
ggplot(rGDP_l,aes(x =Date, y=Value)) +
  geom_line(colour = "blue",size = 1.5) +
```

```
  ggtitle("UK real GDP") +
  theme_bw()
```



If you want to plot a subsample of the data only, the easiest way to do that is to use the `subset` function as you call the data into the `ggplot` function. Compare what has changed in the code and figure out what the `subset` function did.

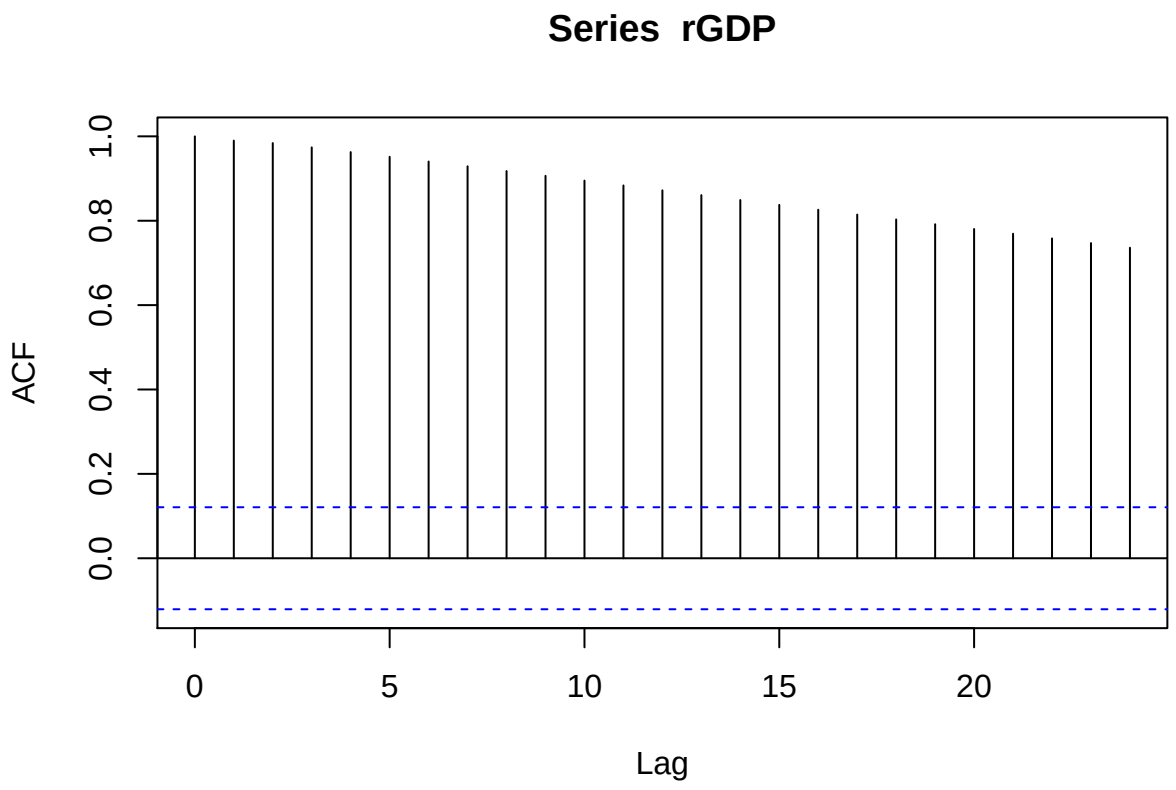
```
ggplot(subset(rGDP_l, Date >"2000-01-01"),aes(x =Date, y=Value)) +
  geom_line(colour = "blue",size = 1.5) +
  ggtitle("UK real GDP") +
  theme_bw()
```



Can you change the above code such that only the GDP from 2005 onwards is shown?

Let's calculate the autocorrelation function:

```
temp_acf <- acf(rGDP)
```



Now we download female unemployment rates and monthly inflation rates from the ONS database. On both occasions we put the

```
# Download: Female unemployment rate (YCPL in database LMS)
ur_female <- pdfetch_ONS("YCPL","LMS")
names(ur_female) <- "Unemp Rate (female)"
periodicity(ur_female)
```

```
## Monthly periodicity from 1992-04-30 to 2020-12-31
```

```
# keep all the data including 2020-Dec
# this was the last observation available at the time this was written
# remove this line if you want to use updated data
ur_female <- ur_female["/2020-12"]

ur_female_l <- data.frame(index(ur_female),stack(as.data.frame(coredata(ur_female))))
names(ur_female_l)[1] <- "Date"
names(ur_female_l)[2] <- "Value"
names(ur_female_l)[3] <- "id"

# Download: Inflation rate (D7OE in database MM23)
infl <- pdfetch_ONS("D7OE","MM23")
names(infl) <- "CPI Inflation"
periodicity(infl)
```

```
## Monthly periodicity from 1988-02-29 to 2021-02-28
```

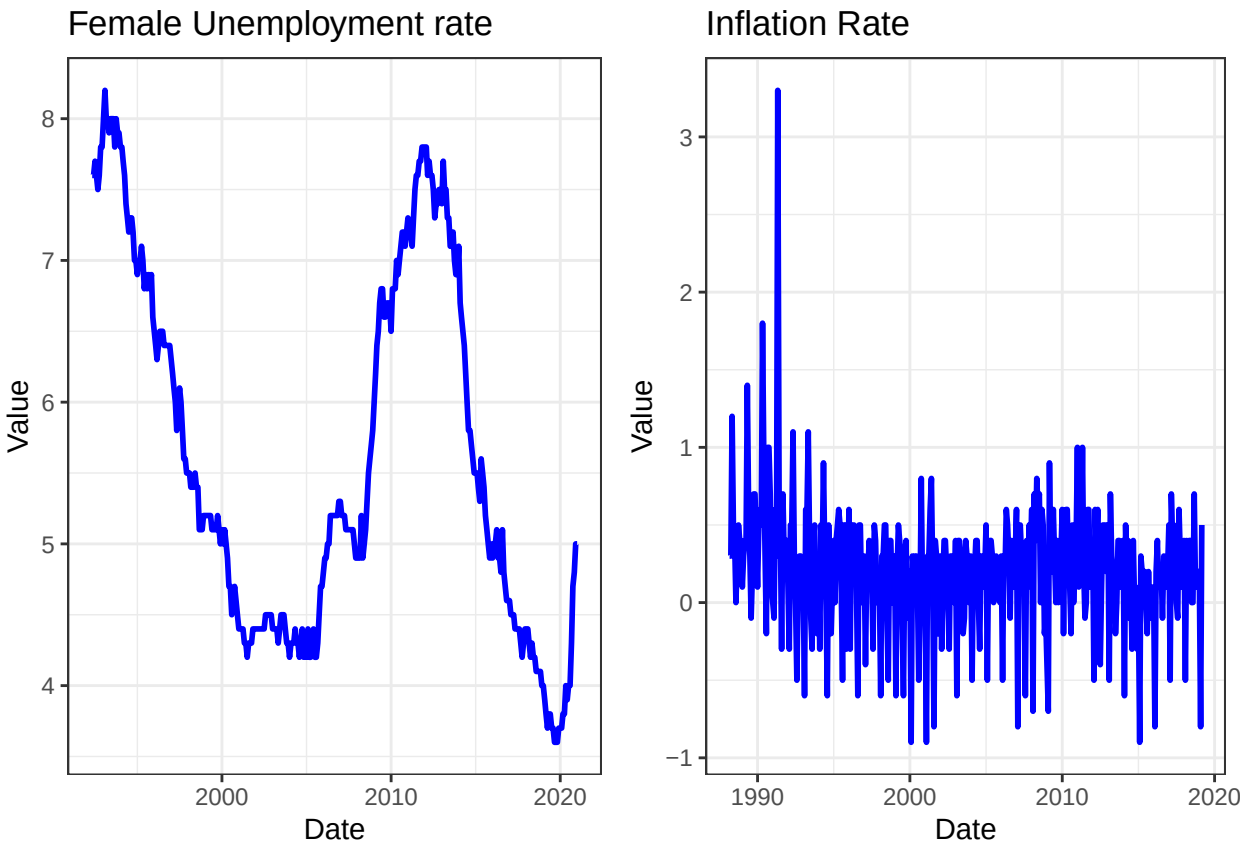
```
# keep all the data including 2019-Feb
# this was the last observation available at the time this was written
# remove this line if you want to use updated data
infl <- infl["/2019-2"]

infl_l <- data.frame(index(infl),stack(as.data.frame(coredata(infl))))
names(infl_l)[1] <- "Date"
names(infl_l)[2] <- "Value"
names(infl_l)[3] <- "id"
```

Now we put both series into one dataframe by attaching the individual dataframes. We can do this as we gave identical names to the three columns in both dataframes.

```
data_l <- rbind(rGDP_l,ur_female_l)
data_l <- rbind(data_l,infl_l)
```

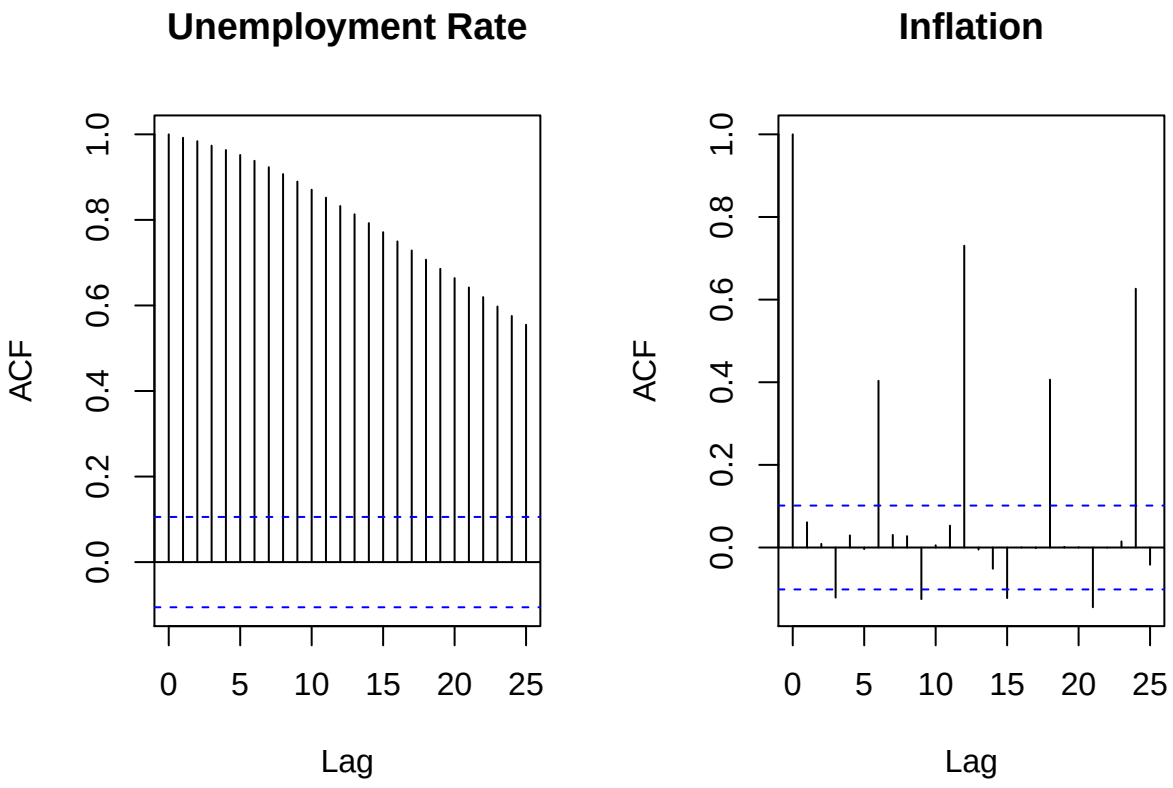
Now we produce some time series plots and ACF functions

```
p1 <- ggplot(data_l[data_l$id == "Unemp Rate (female)",],aes(x =Date, y=Value)) +
  geom_line(colour = "blue",size = 1.0) +
  ggtitle("Female Unemployment rate") +
  theme_bw()

p2 <- ggplot(data_l[data_l$id == "CPI Inflation",],aes(x =Date, y=Value)) +
  geom_line(colour = "blue",size = 1.0) +
  ggtitle("Inflation Rate") +
  theme_bw()

grid.arrange(p1, p2, nrow=1, ncol=2)
```



The `grid.arrange(p1, p2, nrow=1, ncol=2)` line in the code helped us to set the two images next to each other. That is very nice and often useful when you present data. This sort of effect can be achieved in different ways. It is important to realise that usually you can do the same think in R in different ways. So let us present such a different way. Below it is the `par(mfrow=c(1,2))` line which tells R that the next plots should be presented in a 1 by 2 frame.

```
par(mfrow=c(1,2))

acf(ur_female,main = "Unemployment Rate")
acf(infl, main = "Inflation")
```



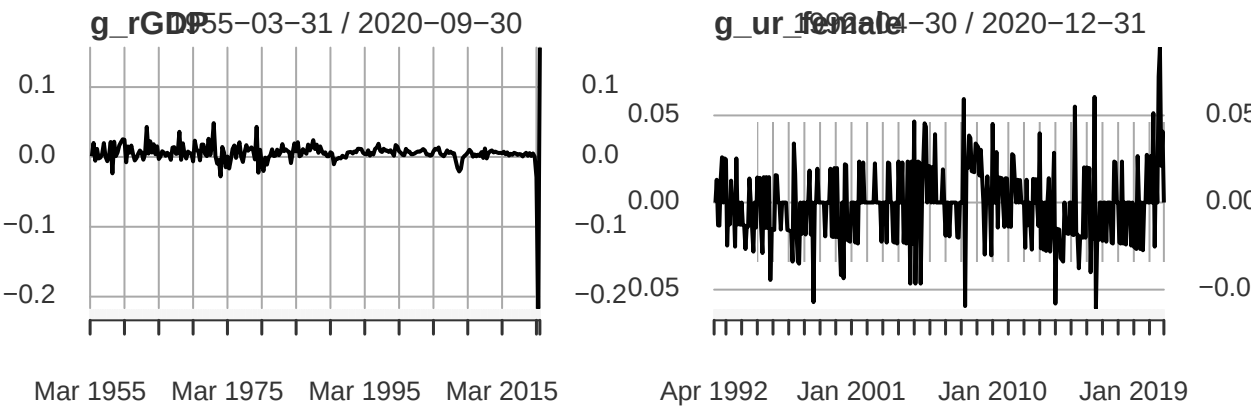
It will be interestimg to see what happens when we difference data series (or often their logs).

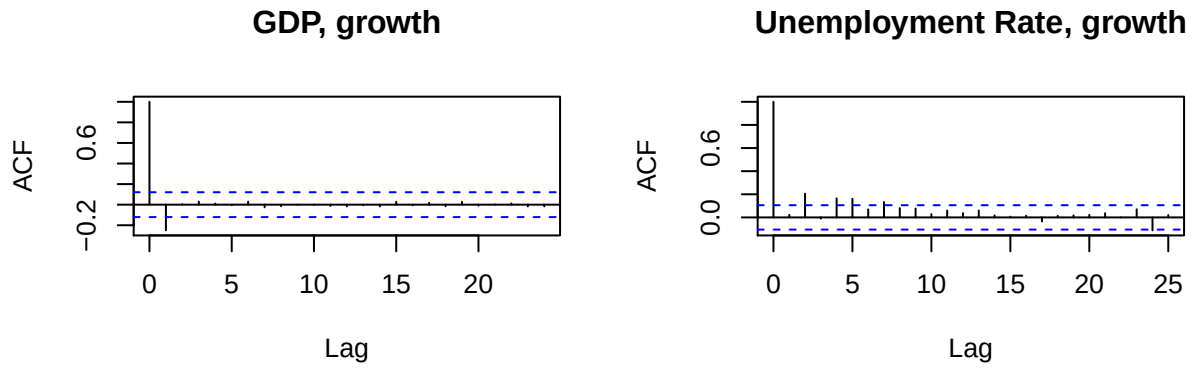
```
g_rGDP <-diff(log(rGDP))
names(g_rGDP) <- "GDP, quarterly growth rate"
g_ur_female <-diff(log(ur_female))
names(g_ur_female) <- "Growth in Unemp Rate (female)"
```

```
par(mfrow=c(1,2))

acf(g_rGDP,main = "GDP, growth")
acf(g_ur_female, main = "Unemployment Rate, growth")
```

When you do this you will get an error message as the differencing created a missing value in the `g_rGDP` and `g_ur_female` series and the `acf` function really dislikes missing values. Hence we need to tell it to ignore these.

```
par(mfrow=c(2,2))
plot(g_rGDP)
plot(g_ur_female)
acf(g_rGDP,main = "GDP, growth", na.action = na.pass)
acf(g_ur_female, main = "Unemployment Rate, growth", na.action = na.pass)
```





Run a simple regression model

Let's run a simple regression model with time series data.

$$ur_t = \alpha + \beta\ rGDP_t + u_t$$

As we have quarterly GDP series we will want to reduce the frequency of the monthly unemployment data to quarterly. the `xts` package which we have been using to deal with the dating aspect of our data has a handy little function to achieve this. `to.period()`.

```
ur_female_q <- to.period(ur_female,period="quarters")
```

As a result we get four values for each quarter (start, end, high and low). We shall associate the last monthly unemployment rate with a particular quarter.

```
ur_female_q <- ur_female_q$ur_female.Close
```

We now have two quarterly series `rGDP` and `ur_female_q`. We shall merge them into the same dataframe.

```
reg_data <- merge(rGDP, ur_female_q)
tail(reg_data,10)
```

```
##            real.GDP ur_female.Close
## 2018-09-30   537326             4.1
## 2018-12-31   538222             4.0
## 2019-03-31   541195             3.7
```

```
## 2019-06-30   541944             3.7
## 2019-09-30   544639             3.6
## 2019-12-31   544733             3.7
## 2020-03-31   529223             3.8
## 2020-06-30   426197             4.0
## 2020-09-30   498429             4.7
## 2020-12-31       NA             5.0
```

By looking at the last 10 observations we can see that automatically the dates have been matched. This is super convenient.

We can now feed these data into the `lm` function.

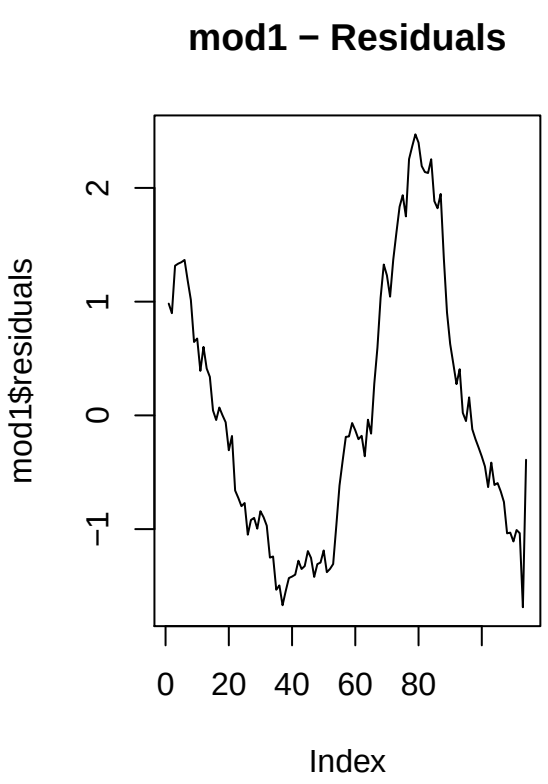
```
mod1 <- lm(ur_female.Close~real.GDP,data = reg_data)
stargazer_HAC(mod1)
```

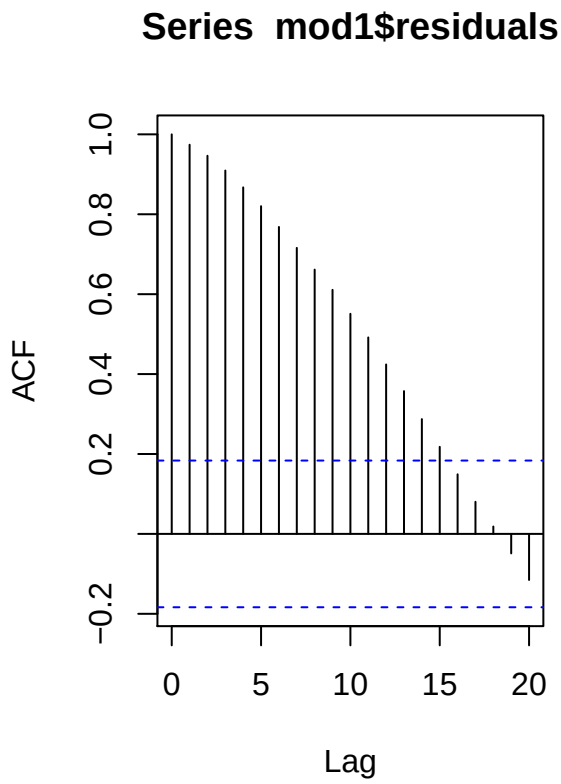
```
##
## ===============================================================
##                                 Dependent variable:
##                     ------------------------------------------
##                                   ur_female.Close
## ---------------------------------------------------------------
## real.GDP                            -0.00001***
##                                      (0.00000)
##
## Constant                             9.208***
##                                       (0.685)
##
## ---------------------------------------------------------------
## Observations                           114
## R2                                    0.202
## Adjusted R2                           0.195
## Residual Std. Error              1.158 (df = 112)
## F Statistic                  28.307*** (df = 1; 112)
## ===============================================================
## Note:                           *p<0.1; **p<0.05; ***p<0.01
##                       Robust standard errors in parenthesis
```

This seems to suggest that higher GDP, significantly, reduces the unemployment rate.

Let's have a look at the residuals.

```
par(mfrow=c(1,2))
plot(mod1$residuals, type = "l", main = "mod1 - Residuals")
acf(mod1$residuals)
```





We can see that there is a significant amount of autocorrelation in the residuals. We can apply the Breusch-Godfrey hypothesis test (`bgtest`). The null hypothesis is that there is no autocorrelation.

```
bgtest(mod1,order=4)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 4
##
## data:  mod1
## LM test = 108.83, df = 4, p-value < 2.2e-16
```

The p-value is virtually 0 suggesting that there is stistically significant evidence that we should reject the null hypothesis of no autocorrelation. What is the consequence?

Spurious correlations and regressions

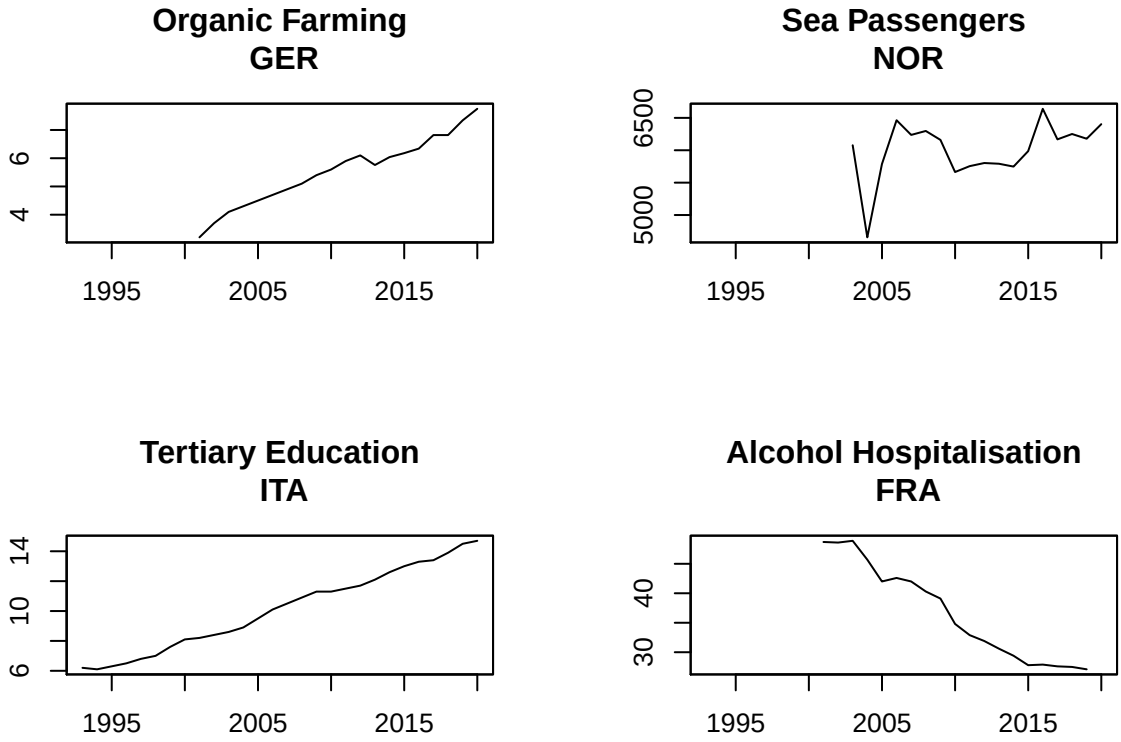
Let's get some datasets fron EUROSTAT.

```
# % of agricultural area Total fully converted and under conversion
# to organic farming in Germany
ser1 <- pdfetch_EUROSTAT("sdg_02_40", GEO=c("DE"))
names(ser1) <- "Organic.Farming.GE"
ser1 <- ser1["/2019"]  # keep data up until 2019 ***

# Thousands of passengers travelling to and from Norway by boat
ser2 <- pdfetch_EUROSTAT("mar_pa_aa", FREQ = "A", DIRECT = "TOTAL", UNIT = "THS_PAS", REP_MAR = "NO")
```

```
names(ser2) <- "Boat.Passengers.NO"
ser2 <- ser2["/2019"]  # keep data up until 2019 ***

# Population with tertiary education (%)
ser3 <- pdfetch_EUROSTAT("edat_lfse_03", FREQ = "A", SEX = "M", AGE = "Y15-64", UNIT = "PC",
                         ISCED11 = "ED5-8", GEO = "IT")
names(ser3) <- "Tert.Educ.IT"
ser3 <- ser3["/2019"]  # keep data up until 2019 ***

# Hospital Discharges, Alcoholic liver disease
ser4 <- pdfetch_EUROSTAT("hlth_co_disch2", FREQ = "A", AGE = "TOTAL", UNIT = "P_HTHAB", SEX = "T",
                         ICD10 = "K70", GEO = "FR")
names(ser4) <- "Alc.Disease.FR"
ser4 <- ser4["/2019"]  # keep data up until 2019 ***

# *** these lines merely make sure to keep the data which were available
#     at the time this file was produced. Remove if you want to work with
#     the latest data
```

I told you earlier that there are different ways to arrange several plots in a grid. Below I present yet another (`layout` - this is possibly the most powerful of the three as you can determine how big each graph should be)! Don't say that this is annoying! I am merely ramming home the message that the same thing can be done in different ways and anyways, you will not be remembering how to do all these things. When you want to do it you will be searching for "R arrange plots in grid" or something similar. Then you will hit on any of these methods and that is what you will adjust.

```
data_sr <- merge(ser1,ser2)
data_sr <- merge(data_sr,ser3)
data_sr <- merge(data_sr,ser4)

layout(matrix(c(1,2,3,4), 2, 2, byrow = TRUE),
   widths=c(1,1), heights=c(1,1))
plot.zoo(data_sr$Organic.Farming.GE, ylab="", xlab = "", main = "Organic Farming\n GER")
plot.zoo(data_sr$Boat.Passengers.NO, ylab="", xlab = "", main = "Sea Passengers\n NOR")
plot.zoo(data_sr$Tert.Educ.IT, ylab="", xlab = "", main = "Tertiary Education\n ITA")
plot.zoo(data_sr$Alc.Disease.FR, ylab="", xlab = "", main = "Alcohol Hospitalisation\n FRA")
```



In the above graphing commands I used `plot.zoo` and not `ggplot` or the standard `plot`. You could have used `ggplot` after bringing the data into long format. You could have used the standard `plot` function. It actually does recognise that the data are time-series data and then automatically adds all sorts of formatting to the graphs (try by replacing `plot.zoo` with `plot` in the above) which is rather annoying. `plot.zoo` is a specialised plotting function for time-series data which allows some manual finessing.

```
mod_sr <- lm(Alc.Disease.FR~Organic.Farming.GE, data = data_sr)
stargazer_HAC(mod_sr)
```

```
##
## ===============================================================
##                               Dependent variable:
##                           -----------------------------------
##                                   Alc.Disease.FR
## ---------------------------------------------------------------
## Organic.Farming.GE                  -6.878***
##                                      (0.483)
##
## Constant                            73.812***
##                                      (2.667)
##
## ---------------------------------------------------------------
## Observations                           19
## R2                                    0.923
## Adjusted R2                           0.918
## Residual Std. Error              2.310 (df = 17)
## F Statistic                  202.752*** (df = 1; 17)
```

```
## ============================================================
## Note:                          *p<0.1; **p<0.05; ***p<0.01
##                       Robust standard errors in parenthesis
```

One way how you can unmask the spuriousness, if both series are trending is to include a time trend

```
mod_sr2 <- lm(Alc.Disease.FR~Organic.Farming.GE+index(data_sr), data = data_sr)
```

Now we can see that the time trend is the variable which becomes significant and the coefficient to the organic farming variable becomes insignificant (as it should be).

We may also want to look at running a model in the differences of variables rather than the levels.

```
mod_sr3 <- lm(diff(Alc.Disease.FR)~diff(Organic.Farming.GE), data = data_sr)
stargazer_HAC(mod_sr,mod_sr2,mod_sr3,type_out = "text", omit.stat = "f")
```

```
##
## ================================================================================
##                                          Dependent variable:
##                             ---------------------------------------------------
##                                    Alc.Disease.FR          diff(Alc.Disease.FR)
##                                (1)             (2)                 (3)
## --------------------------------------------------------------------------------
## Organic.Farming.GE          -6.878***         0.486
##                              (0.427)         (0.748)
##
## index(data_sr)                               -0.004***
##                                              (0.0003)
##
## diff(Organic.Farming.GE)                                          1.329
##                                                                  (0.946)
##
## Constant                     73.812***       93.946***         -1.506***
##                              (2.258)         (2.619)           (0.456)
##
## --------------------------------------------------------------------------------
## Observations                   19              19                  18
## R2                            0.923           0.958              0.036
## Adjusted R2                   0.918           0.952              -0.024
## Residual Std. Error      2.310 (df = 17) 1.760 (df = 16)   1.393 (df = 16)
## ================================================================================
## Note:                                              *p<0.1; **p<0.05; ***p<0.01
##                                    Newey-West standard errors in parenthesis
```

Run a simple regression model - but better

Let's run a simple regression model with time series data.

$$\Delta ur_t = \alpha + \beta\ \Delta rGDP_t + u_t$$

As we have quarterly GDP series we will want to reduce the frequency of the monthly unemployment data to quarterly. the `xts` package which we have been using to deal with the dating aspect of our data has a handy little function to achieve this. `to.period()`.

```
ur_female_q <- to.period(ur_female,period="quarters")
```

As a result we get four values for each quarter (start, end, high and low). We shall associate the last monthly unemployment rate with a particular quarter.

```
ur_female_q <- ur_female_q$ur_female.Close
```

We now have two quarterly series `rGDP` and `ur_female_q`. We shall merge them into the same dataframe.

```
reg_data <- merge(rGDP, ur_female_q)
tail(reg_data,10)
```

```
##            real.GDP ur_female.Close
## 2018-09-30   537326             4.1
## 2018-12-31   538222             4.0
## 2019-03-31   541195             3.7
## 2019-06-30   541944             3.7
## 2019-09-30   544639             3.6
## 2019-12-31   544733             3.7
## 2020-03-31   529223             3.8
## 2020-06-30   426197             4.0
## 2020-09-30   498429             4.7
## 2020-12-31       NA             5.0
```

By looking at the last 10 observations we can see that automatically the dates have been matched. This is super convenient.

As we will be modelling the differenced logs of the GDP and unemployment rate it is most convenient to create these variables explicitely in the data frame, otherwise we will have to deal with very long variable names.

```
# we multiply by 100 to express in percentage points, i.e. 0.5 is 0.5% or 0.005
reg_data$d_lgdp <- 100*diff(log(reg_data$real.GDP))
reg_data$d_lur <- 100*diff(log(reg_data$ur_female.Close))
```

We can now feed these data into the `lm` function.

```
mod4 <- lm(d_lur~d_lgdp,data = reg_data)
stargazer_HAC(mod4)
```

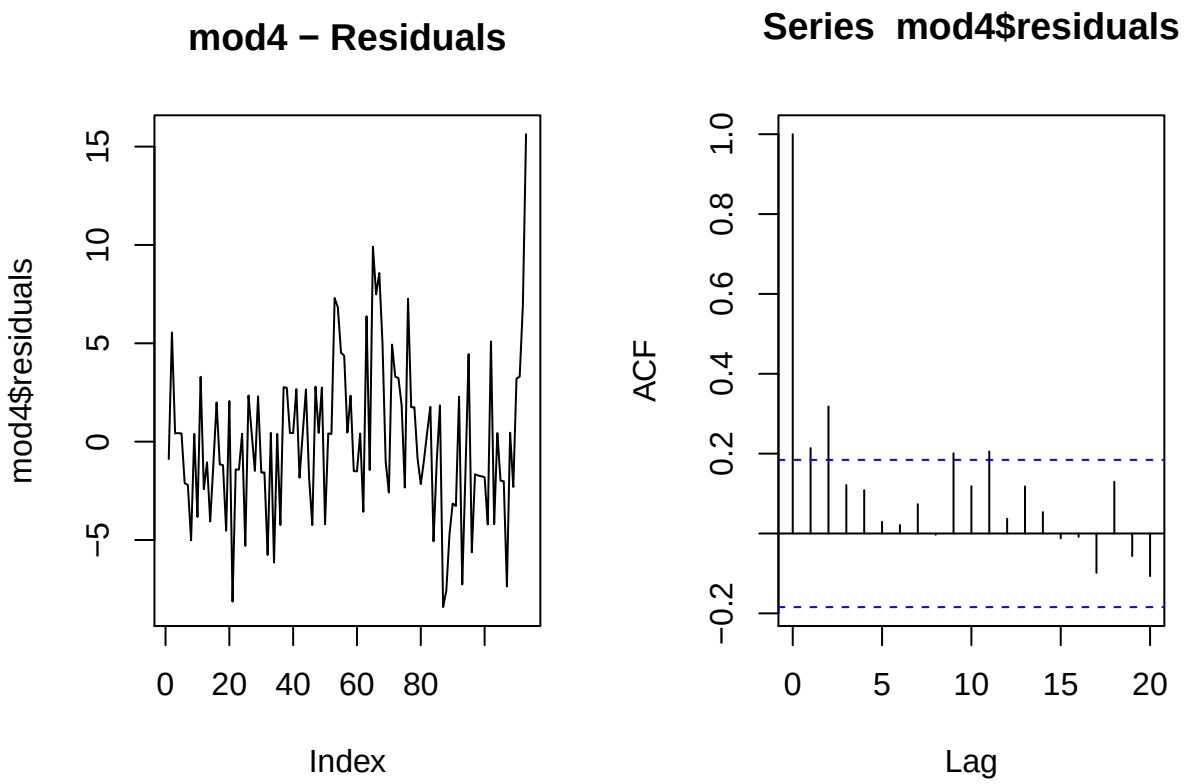
```
##
## ===============================================================
##                               Dependent variable:
##                           -----------------------------------
##                                         d_lur
## ---------------------------------------------------------------
## d_lgdp                                   0.061
##                                         (0.146)
##
## Constant                                -0.464
##                                         (0.384)
##
## ---------------------------------------------------------------
## Observations                              113
## R2                                       0.002
## Adjusted R2                             -0.007
## Residual Std. Error                4.026 (df = 111)
## F Statistic                        0.178 (df = 1; 111)
```

```
## ============================================================
## Note:                            *p<0.1; **p<0.05; ***p<0.01
##                       Robust standard errors in parenthesis
```

This seems to suggest that higher GDP, significantly, reduces the unemployment rate.

Let's have a look at the residuals.

```
par(mfrow=c(1,2))
plot(mod4$residuals, type = "l", main = "mod4 - Residuals")
acf(mod4$residuals)
```



We can see that, at lag 2, there is a small amount of autocorrelation in the residuals. We can again apply the hypothesis test the Breusch-Godfrey test (`bgtest`). The null hypothesis is that there is no autocorrelation.

```
bgtest(mod4,order=4)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 4
##
## data:  mod4
## LM test = 17.134, df = 4, p-value = 0.00182
```

The p-value of 0.0605 suggests that there is a little evidence that we should reject the null hypothesis of no autocorrelation. What is the consequence? Fortunately, here, there is little evidence. We already calculated HAC standard errors.

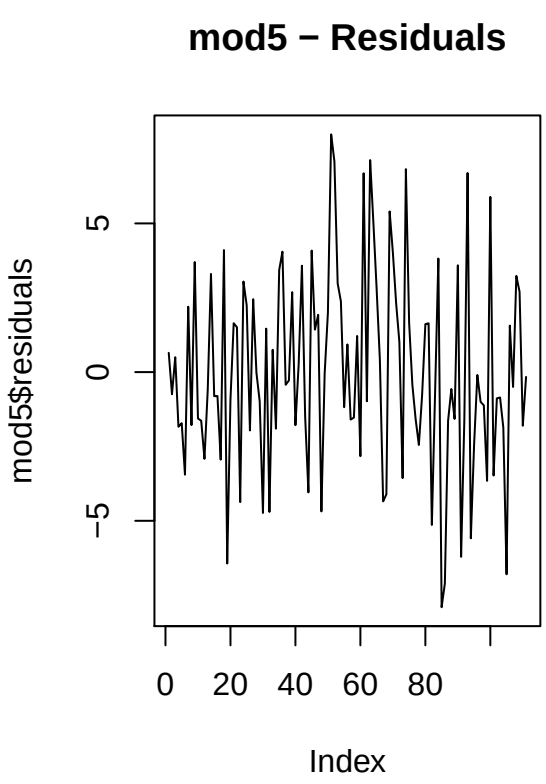
ADL Models

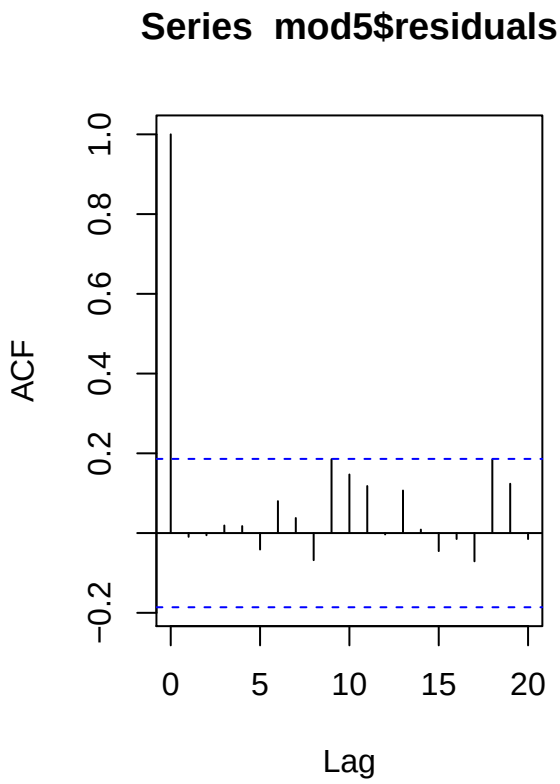
What happens if we include a lag dependent variable. Let us create the lagged variable.

```
mod5 <- lm(d_lur~lag(d_lur,1)+lag(d_lur,2)+d_lgdp+lag(d_lgdp,1)+lag(d_lgdp,2),data = reg_data)
stargazer_HAC(mod5)
```

```
##
## ===========================================================
##                               Dependent variable:
##                           -------------------------------
##                                        d_lur
## -----------------------------------------------------------
## lag(d_lur, 1)                           0.067
##                                        (0.094)
##
## lag(d_lur, 2)                         0.268***
##                                        (0.094)
##
## d_lgdp                                 -0.166
##                                        (0.136)
##
## lag(d_lgdp, 1)                        -0.804***
##                                        (0.207)
##
## lag(d_lgdp, 2)                         -0.107
##                                        (0.695)
##
## Constant                                0.104
##                                        (0.444)
##
## -----------------------------------------------------------
## Observations                             111
## R2                                      0.307
## Adjusted R2                             0.274
## Residual Std. Error                3.418 (df = 105)
## F Statistic                    9.306*** (df = 5; 105)
## ===========================================================
## Note:                           *p<0.1; **p<0.05; ***p<0.01
##                         Robust standard errors in parenthesis
```

```
par(mfrow=c(1,2))
plot(mod5$residuals, type = "l",main = "mod5 - Residuals")
acf(mod5$residuals)
```





Now the coefficient to the real GDP growth rate at time t remains statistically insignificant, but only marginally. Most important appears to be the unemployment rate from two quarters prior (t-2), `lag(d_lur,2)`.

```
bgtest(mod5,order=4)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 4
##
## data:  mod5
## LM test = 0.14449, df = 4, p-value = 0.9975
```

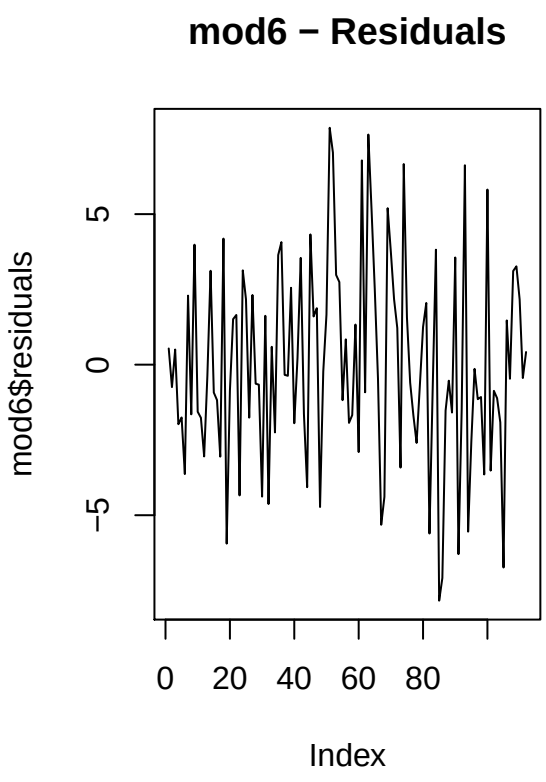
Now we remove the contemporaneous GDP growth rate.

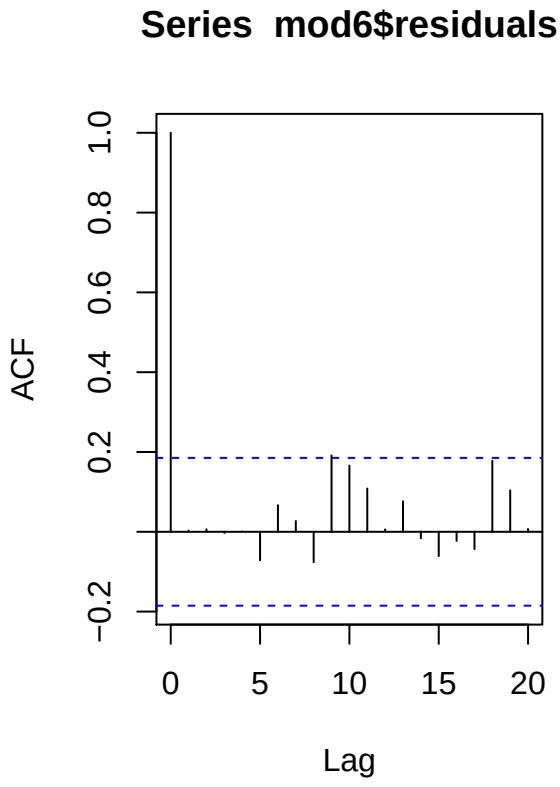
```
mod6 <- lm(d_lur~lag(d_lur,1)+lag(d_lur,2)+lag(d_lgdp,1)+lag(d_lgdp,2),data = reg_data)
stargazer_HAC(mod6)
```

```
##
## ===================================================
##                             Dependent variable:
##                         -----------------------
##                                    d_lur
## ---------------------------------------------------
## lag(d_lur, 1)                       0.076
##                                    (0.092)
##
## lag(d_lur, 2)                     0.268***
##                                    (0.092)
##
```

```
## lag(d_lgdp, 1)                    -0.625***
##                                    (0.136)
##
## lag(d_lgdp, 2)                    -0.588***
##                                    (0.186)
##
## Constant                           0.229
##                                   (0.336)
##
## ------------------------------------------------------
## Observations                         112
## R2                                  0.312
## Adjusted R2                         0.286
## Residual Std. Error           3.416 (df = 107)
## F Statistic                12.123*** (df = 4; 107)
## ======================================================
## Note:                          *p<0.1; **p<0.05; ***p<0.01
##                     Robust standard errors in parenthesis
```

```
par(mfrow=c(1,2))
plot(mod6$residuals, type = "l",main = "mod6 - Residuals")
acf(mod6$residuals)
```






```
bgtest(mod6,order=4)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 4
```

```
##
## data:  mod6
## LM test = 0.052204, df = 4, p-value = 0.9997
```

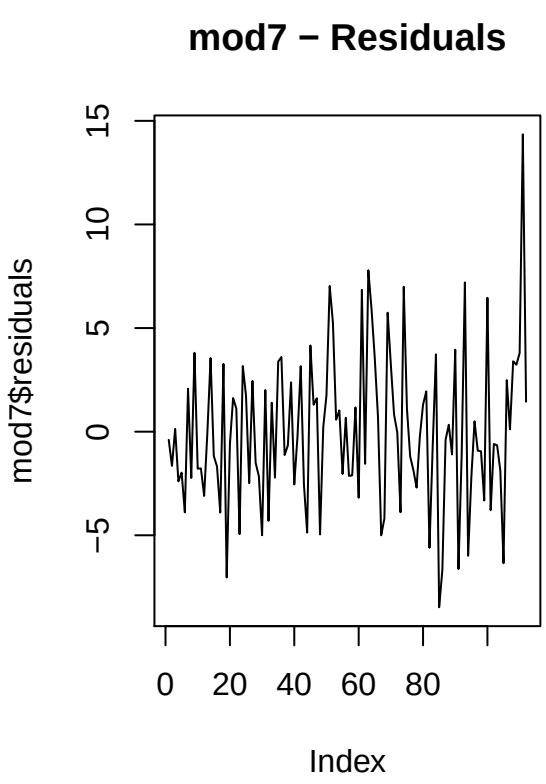
Autoregressive Models

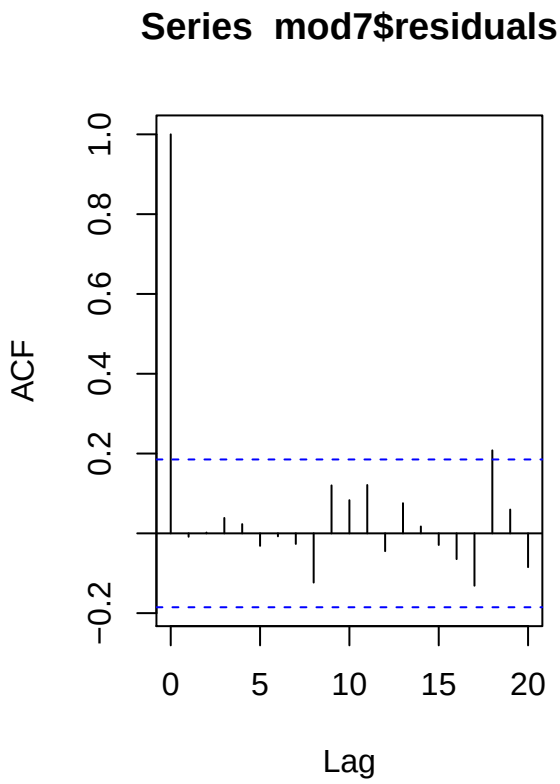
Now we remove the GDP growth rate altogether from the model.

```
mod7 <- lm(d_lur~lag(d_lur,1)+lag(d_lur,2),data = reg_data)
stargazer_HAC(mod7)
```

```
##
## ===============================================================
##                               Dependent variable:
##                           -----------------------------------
##                                          d_lur
## ---------------------------------------------------------------
## lag(d_lur, 1)                           0.192**
##                                         (0.090)
##
## lag(d_lur, 2)                          0.348***
##                                         (0.098)
##
## Constant                                -0.134
##                                         (0.358)
##
## ---------------------------------------------------------------
## Observations                              112
## R2                                       0.165
## Adjusted R2                              0.149
## Residual Std. Error                3.729 (df = 109)
## F Statistic                      10.736*** (df = 2; 109)
## ===============================================================
## Note:                               *p<0.1; **p<0.05; ***p<0.01
##                         Robust standard errors in parenthesis
```

```
par(mfrow=c(1,2))
plot(mod7$residuals, type = "l",main = "mod7 - Residuals")
acf(mod7$residuals)
```






```
bgtest(mod7,order=4)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 4
##
## data:  mod7
## LM test = 0.82121, df = 4, p-value = 0.9356
```

Let's look at all these models together in one table.

```
stargazer_HAC(mod4,mod5,mod6,mod7,type_out = "text", omit.stat = "f")
```

```
##
## ================================================================================
##                                        Dependent variable:
##                    ------------------------------------------------------------
##                                              d_lur
##                           (1)             (2)             (3)             (4)
## --------------------------------------------------------------------------------
## lag(d_lur, 1)                              0.067           0.076          0.192**
##                                           (0.105)         (0.101)         (0.093)
##
## lag(d_lur, 2)                            0.268***        0.268***        0.348***
##                                           (0.082)         (0.083)         (0.083)
##
## d_lgdp                     0.061          -0.166*
##                           (0.212)         (0.086)
##
```

```
## lag(d_lgdp, 1)                          -0.804***        -0.625***
##                                          (0.137)          (0.053)
##
## lag(d_lgdp, 2)                           -0.107          -0.588***
##                                          (0.771)          (0.117)
##
## Constant                  -0.464          0.104            0.229            -0.134
##                          (0.473)          (0.460)          (0.324)          (0.382)
##
## ------------------------------------------------------------------------------------
## Observations               113              111              112              112
## R2                        0.002            0.307            0.312            0.165
## Adjusted R2               -0.007           0.274            0.286            0.149
## Residual Std. Error 4.026 (df = 111) 3.418 (df = 105) 3.416 (df = 107) 3.729 (df = 109)
## ====================================================================================
## Note:                                                    *p<0.1; **p<0.05; ***p<0.01
##                                            Newey-West standard errors in parenthesis
```

Information criteria

Let's say we wanted to figure out whether it would be better to include more lags. In addition to models `mod6` (ADL) and `mod7` (AR), we shall estimate the equivalent models with 4 lags. In order to then decide which model is best we look at an information criteria. This recognises that the inclusion of additional variables (lags) will improve the fit, but it will also reduce the precision with which the parameters are estimated. That can be detrimental especially for forecasting. Information criteria take this trade-off into account and offer a way to chose the best model.

```
mod6_4 <- lm(d_lur~lag(d_lur,1)+lag(d_lur,2)+lag(d_lur,3)+lag(d_lur,4)+
               lag(d_lgdp,1)+lag(d_lgdp,2)+lag(d_lgdp,3)+lag(d_lgdp,4),data = reg_data)
mod7_4 <- lm(d_lur~lag(d_lur,1)+lag(d_lur,2)+lag(d_lur,3)+lag(d_lur,4),data = reg_data)

stargazer_HAC(mod6,mod7,mod6_4,mod7_4,type_out = "text", omit.stat = "f")
```

```
##
## ====================================================================================
##                                        Dependent variable:
##                     ---------------------------------------------------------------
##                                              d_lur
##                            (1)              (2)              (3)              (4)
## ------------------------------------------------------------------------------------
## lag(d_lur, 1)             0.076           0.192**           0.073           0.183*
##                          (0.101)          (0.093)          (0.117)          (0.101)
##
## lag(d_lur, 2)           0.268***         0.348***         0.272***         0.354***
##                          (0.083)          (0.083)          (0.088)          (0.094)
##
## lag(d_lur, 3)                                             0.013            0.032
##                                                          (0.095)          (0.096)
##
## lag(d_lur, 4)                                             0.048           -0.010
##                                                          (0.091)          (0.100)
##
## lag(d_lgdp, 1)          -0.625***                        -0.638***
```

```
##                          (0.053)                          (0.058)
##
## lag(d_lgdp, 2)         -0.588***                        -0.660***
##                          (0.117)                          (0.173)
##
## lag(d_lgdp, 3)                                            0.365
##                                                          (0.903)
##
## lag(d_lgdp, 4)                                            0.247
##                                                          (0.908)
##
## Constant                 0.229           -0.134          -0.018          -0.102
##                         (0.324)         (0.382)         (0.564)         (0.392)
##
## ------------------------------------------------------------------------------------
## Observations              112             112             110             110
## R2                       0.312           0.165           0.317           0.167
## Adjusted R2              0.286           0.149           0.263           0.135
## Residual Std. Error 3.416 (df = 107) 3.729 (df = 109) 3.503 (df = 101) 3.794 (df = 105)
## ====================================================================================
## Note:                                                  *p<0.1; **p<0.05; ***p<0.01
##                                             Newey-West standard errors in parenthesis
```

```
AIC(mod6, mod7, mod6_4,mod7_4)
```

```
## Warning in AIC.default(mod6, mod7, mod6_4, mod7_4): models are not all fitted to
## the same number of observations
```

```
##        df      AIC
## mod6    6 599.8829
## mod7    4 617.6051
## mod6_4 10 598.5663
## mod7_4  6 612.3832
```

We chose the model with the smallest value for the information criterion and on this occasion this is the AR model with 4 lags (despite lags 3 and 4 not bein statistically significant).