Time series analysis with R

Computer class 23th April

Outline

1. Getting started
    1.1. Downloading and Installing R
    1.2. Installing ‘tseries’

2. Working with series

3. Model specification and estimation

4. Diagnostic checking

5. Simulations of ARMA(p,q) models

1.1. Downloading and Installing R

- Go to the web page: http://www.r-project.org/

- Then, go to the box Getting started and click on **download R**

- Then select any of the options, for example University of Bristol and choose which type of download you prefer:
  - Download R for Linux
  - Download R for (Mac) OS X
  - Download R for Windows

1.1. Downloading and Installing R

- Now, you are ready to work with R!!!

1.2. INSTALLING ‘TSERIES’

- We are going to install package “tseries” to work with time series.

  This can be done either by

  1. Clicking in package/install package/tseries

  2. In R console, type the command type

  the command

  install.packages("tseries") # *install it … you'll be asked to choose the closest CRAN mirror*

  *library(*tseries*)*   *# then load it (has to be done at the start of each session)*

1.2. Installing 'tseries'

- For information about the package write

help(tseries)

- Other useful packages: timeSeries, atsa, timeDate, Rcmdr,TeachingDemos…

2. Working with series

- Today we are going to specify an ARIMA model for the European Industrial Production Index.

- To read the data type:

```
ipi <- read.csv(file= "C:/Users/Juan de Dios/Desktop/jtena/Clases/Clases_Carlos_III/Temas_doctorado/IPI.dat ", head=TRUE, sep= "")
```

2. Working with series

- Give the command at least one of three arguments. The first argument is the name of file. The second argument indicates whether or not the first row is a set of labels. The third argument indicates that there is a comma between each number of each line.

- ipi *# print it to the screen It gives the European industrial production index and its five disaggregates by economic destination*

2. Working with series

- Other useful commands are:

ipi[,1] # *the first column*

ipi[1,] # *the first row*

nipi=ipi[-(1:80),] # *keep everything EXCEPT the first 80 rows*

dim(ipi) # *gives the number of rows and columns of the matrix*

nrow(ipi) # *number of rows*

ncol(ipi) # *number of columns*

2. Working with series

- *However, what we have now is just a matrix of data. To make time series of these objects type*

  y =ipi[,1] #take the first column of the matrix

  y=ts(y,star=1991,frequency=12) #transform the vector into a time series

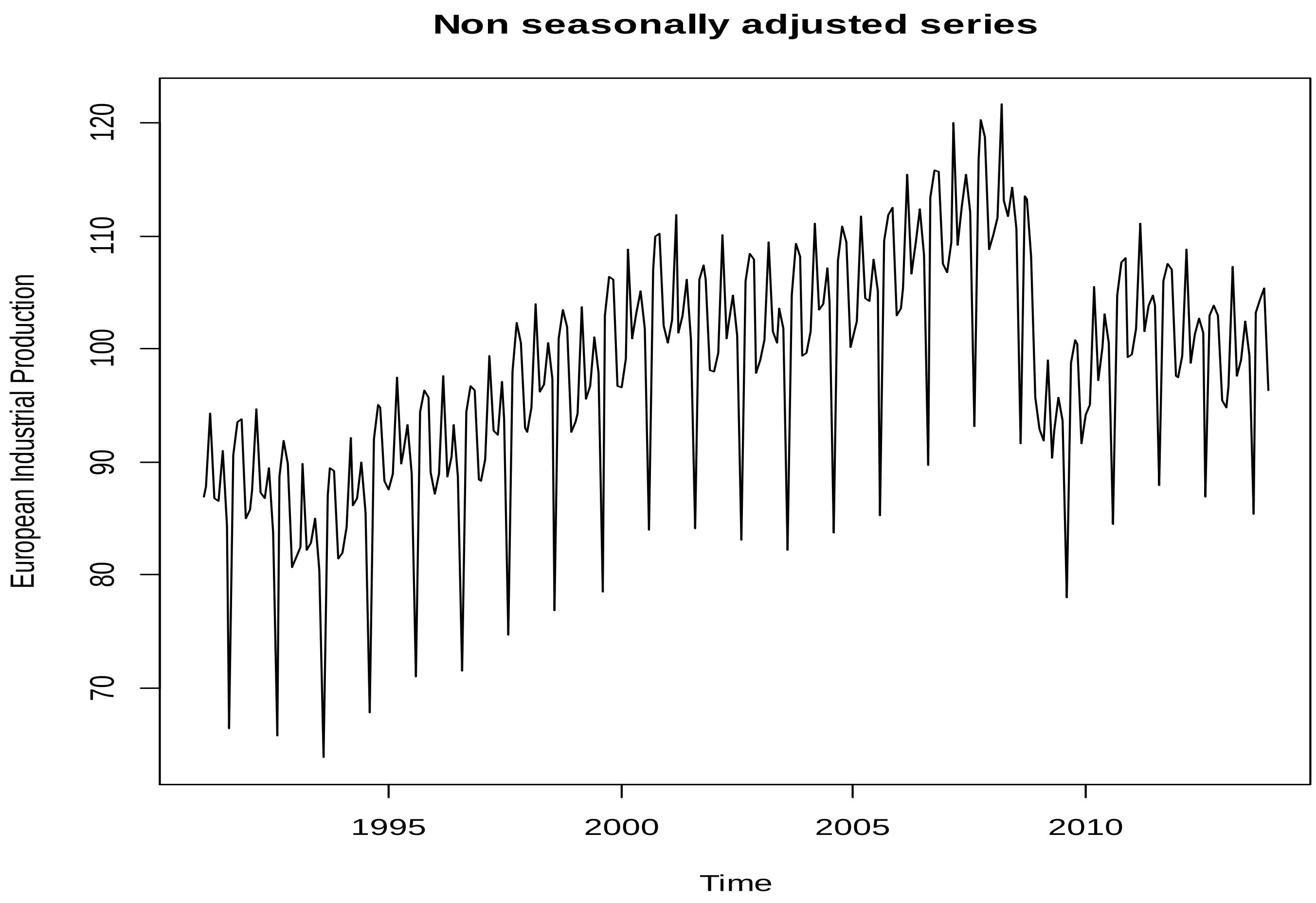
2. Working with series

- We can plot the data

  ```
  plot(y, ylab=" European Industrial Production ", main=" Non seasonally adjusted series ")
  ```

- We can also copy the figure by clicking on the right bottom of the mouse. Then select ‘Copy as a metafile’ and past it to your document.

2. WORKING WITH SERIES



2. Working with series

- What do you think about the features of this series:
  - Trend
  - Seasonality
  - Logarithmic transformation
  - Effect of the economic crisis

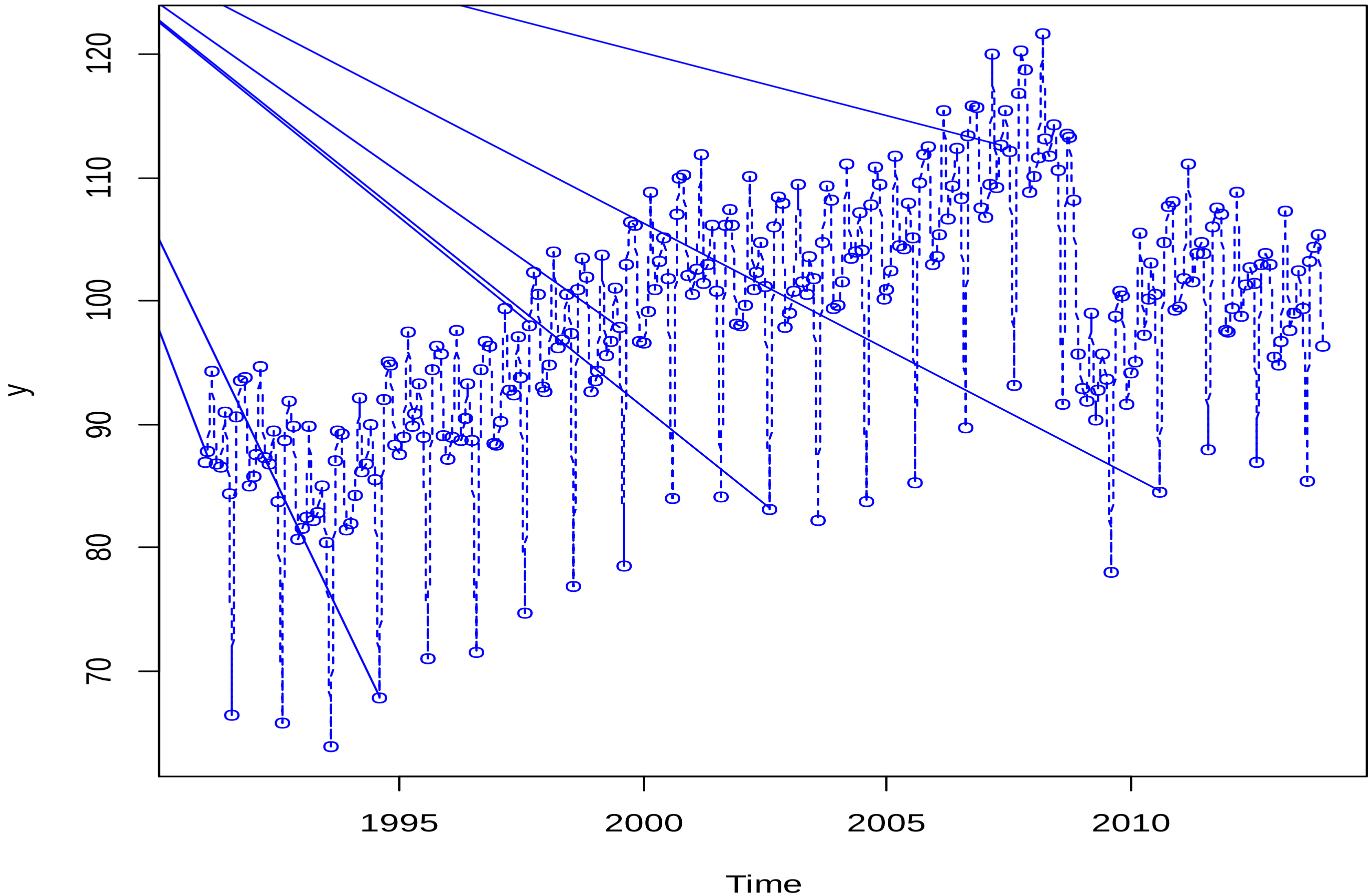
2. Working with series

- Try these and see what happens:

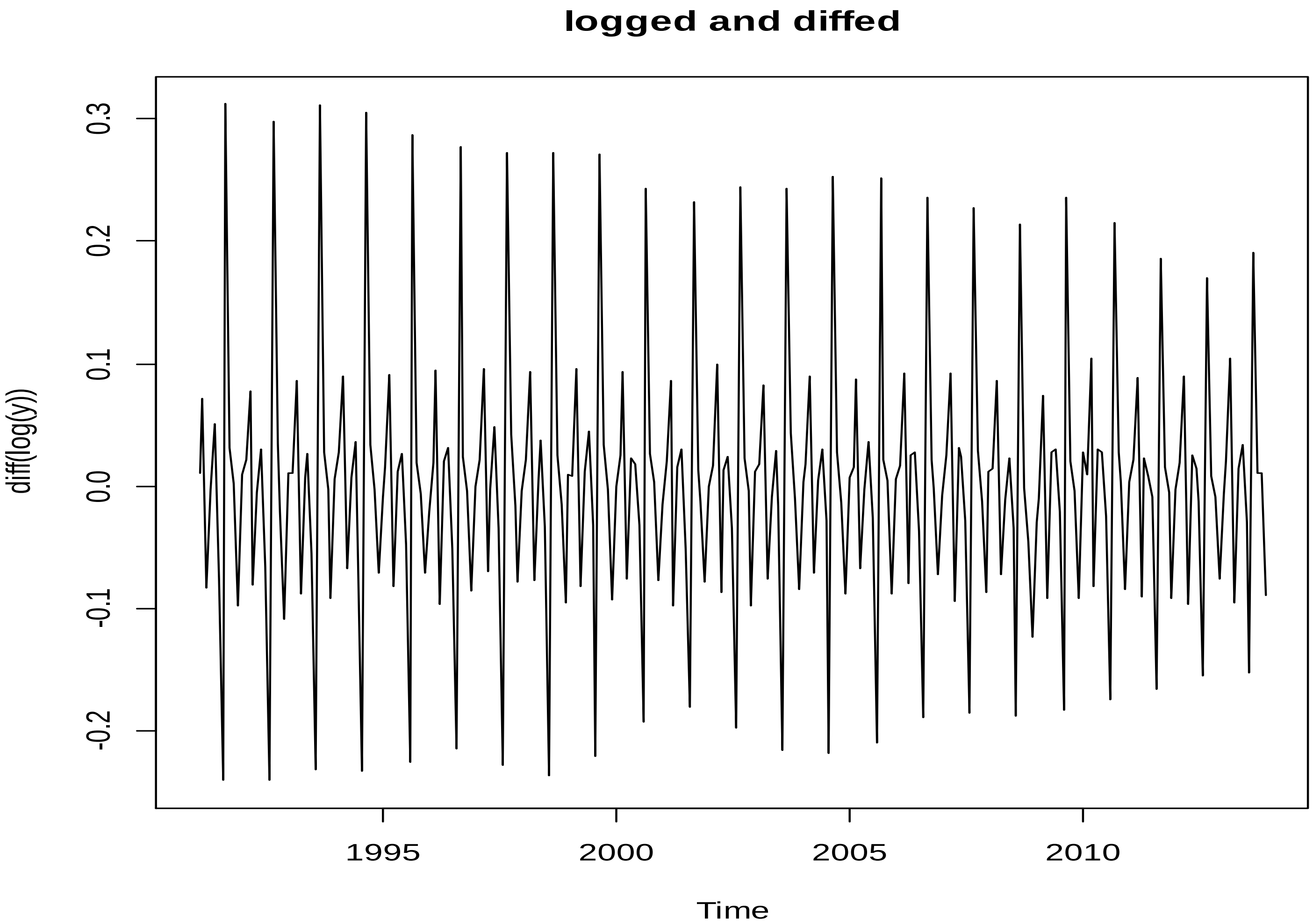
```
plot(y, type="o", col="blue", lty="dashed")

plot(diff(log(y)), main="logged and diffed")
```

2. Working with series



2. Working with series



3. Model specification and estimation

- We need to decide about the presence of a unit root. This decisions is undertaken by means of Dickey-Fuller tests.

```
adf.test(log(y))
adf.test(diff(log(y)))
adf.test(diff(log(y),12))
adf.test(diff(diff(log(y)),12))
```

3. Model specification and estimation

- We also look at the correlograms

```
acf(log(y),48)
acf(diff(log(y)),48)
acf(diff(log(y),12),48)
acf(diff(diff(log(y)),12),48)
```

3. Model specification and estimation

- How many differences do you think this series requires to become stationary?

3. Model specification and estimation

- Although the series with only one regular difference seems to be stationary according to the ADF test, the correlogram reveals the presence of a seasonal behavior that is not stationary.

- The series with only one seasonal difference shows a persistent autocorrelation behavior in its correlogram.

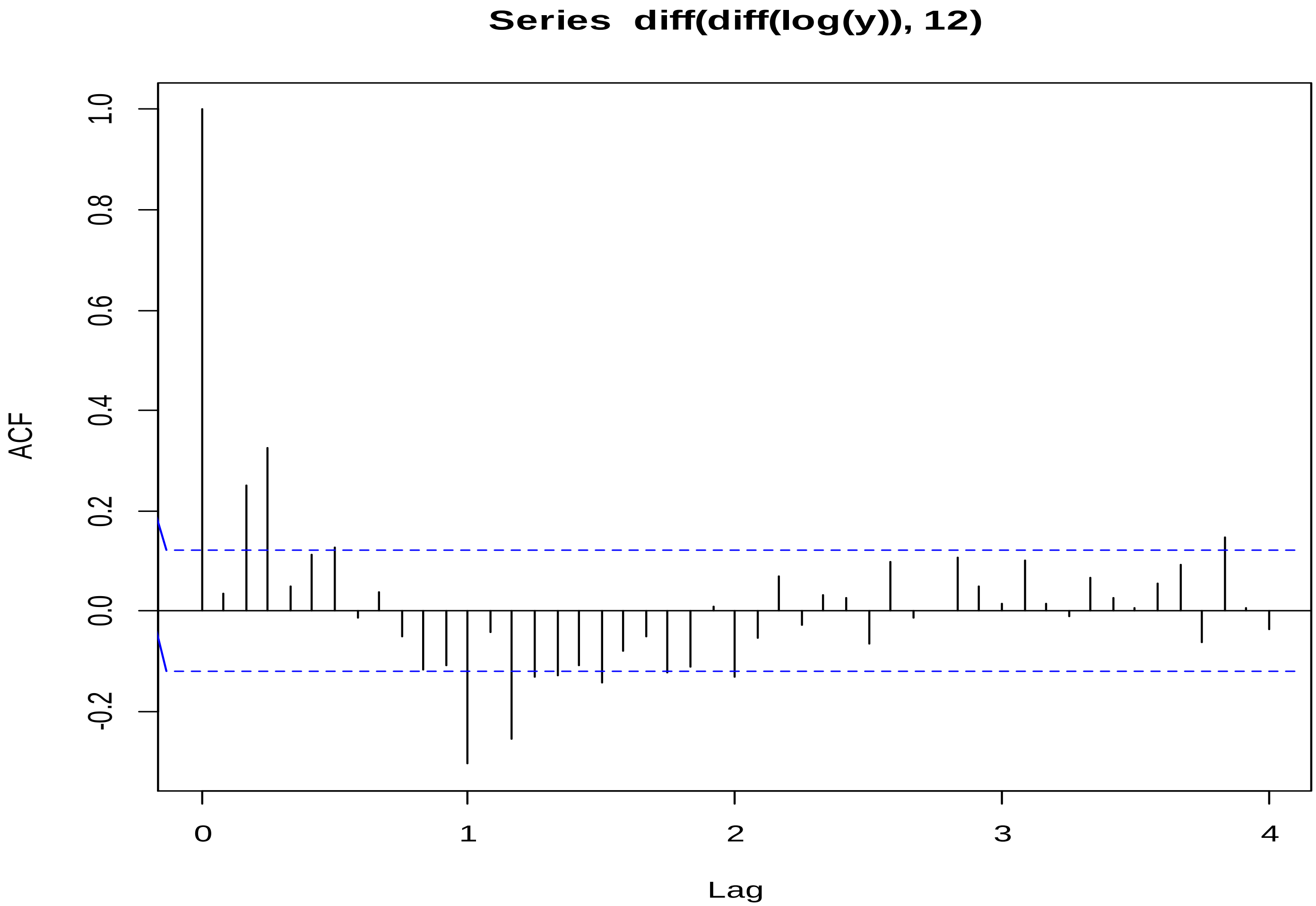
- So, we undertake a conservative approach an consider that the IPI series need a regular and a seasonal difference to become stationary.

3. Model specification and estimation

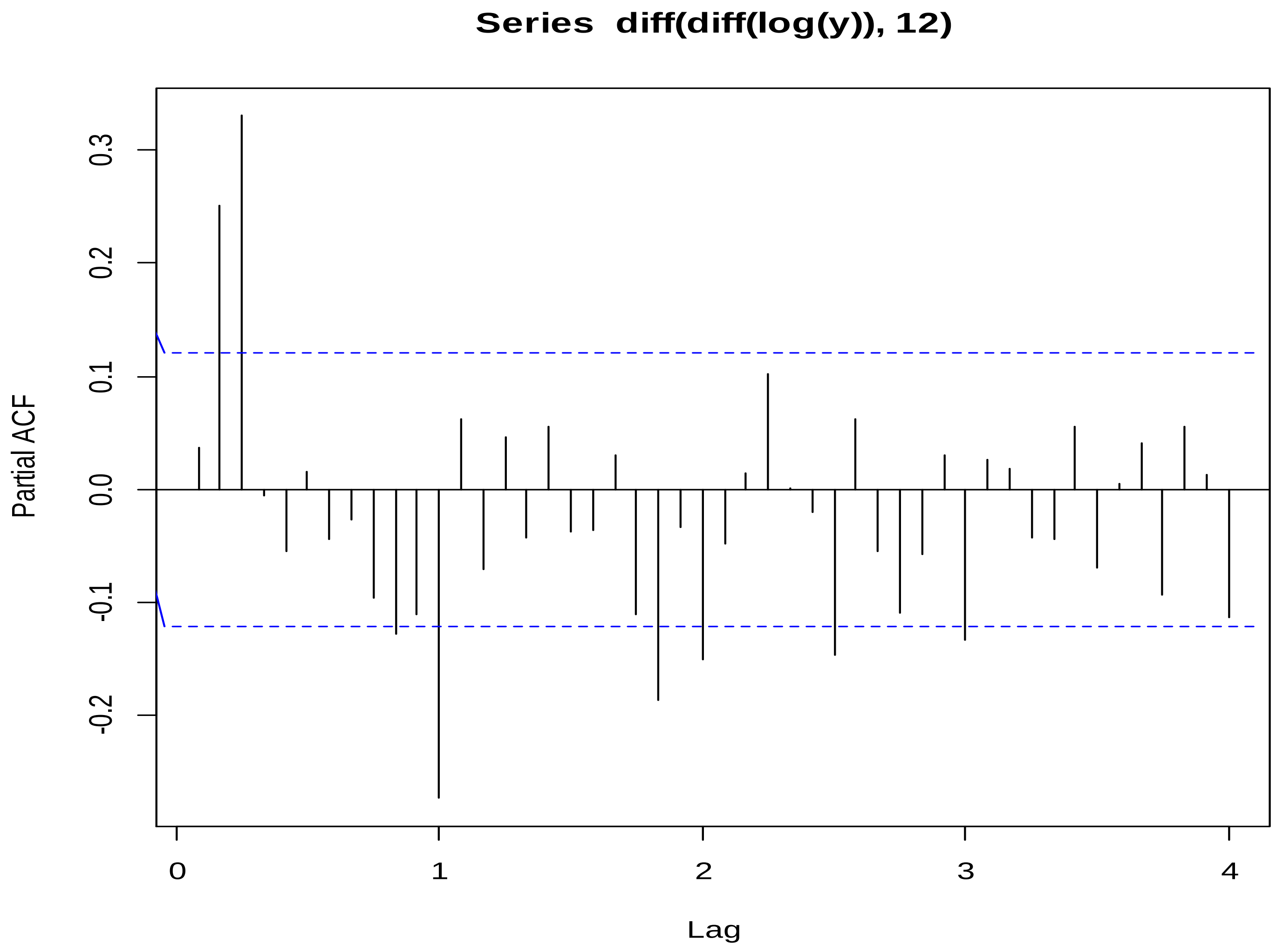
- We propose an arima model for this series based on its correlagram and partial correlogram

```
acf(diff(diff(log(y)),12),48)
pacf(diff(diff(log(y)),12),48)
```

3. Model specification and estimation



3. Model specification and estimation



3. Model specification and estimation

- There is not a clear structure but it seems there is a cut-off at the regular and seasonal lags. Therefore, a tentative specification is an

  ARIMA(0,1,1)x(0,1,1) model.

```
model <- arima(diff(diff(log(y)),12), c(0, 1, 1),
  seasonal = list(order = c(0, 1, 1), period = 12))
```

3. Model specification and estimation

- What do you think about the estimation?
- What alternative estimation would you propose?

Type the following commands

```
aicmodel<-model$aic

res<-residuals(model)
```

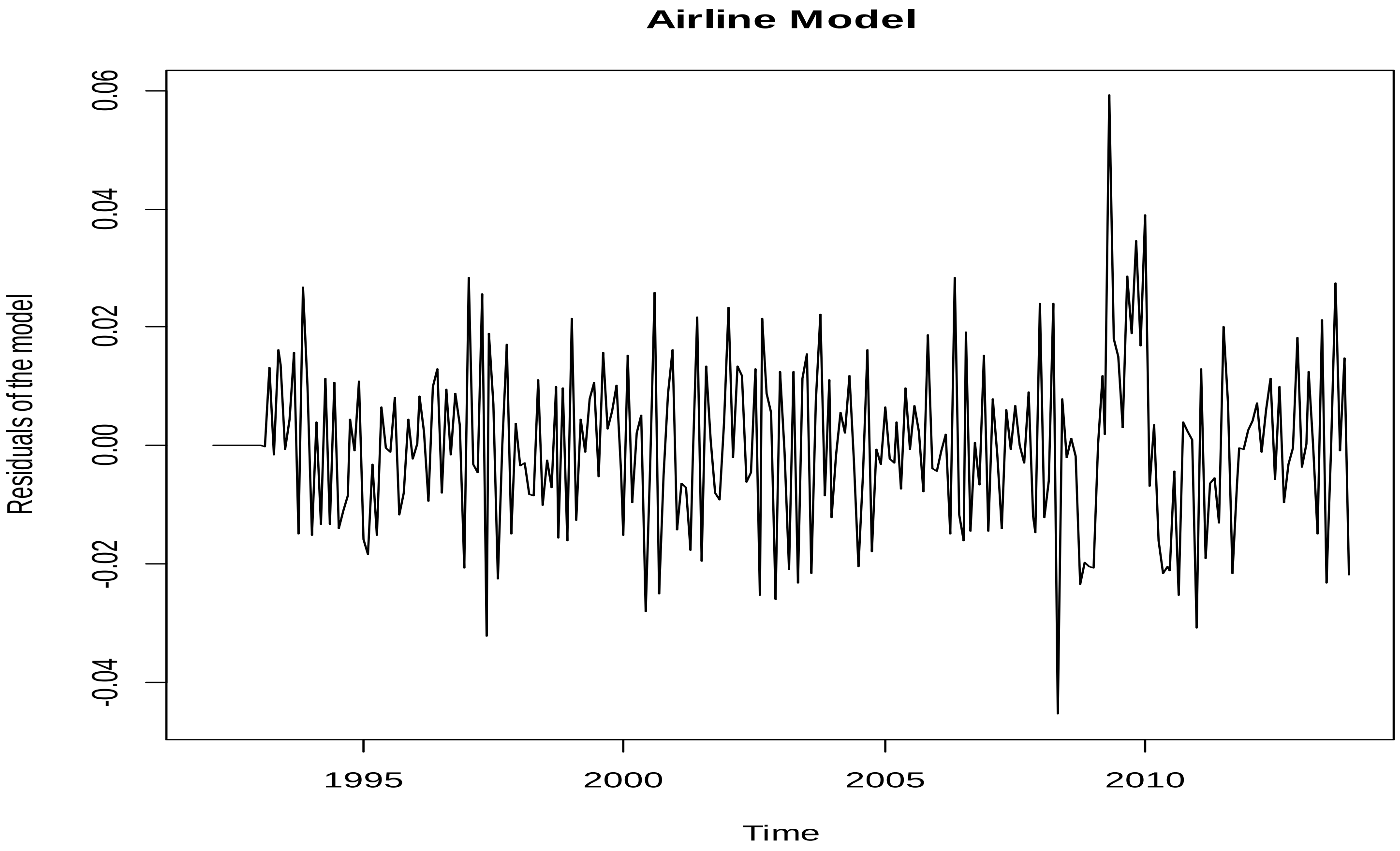
4. Diagnostic checking

- The two estimated parameters are significant.

- Let's have a look at the residuals

```
plot(residuals(model), ylab=" Residuals of the
   model ", main=" Airline Model ")
```

4. Diagnostic checking

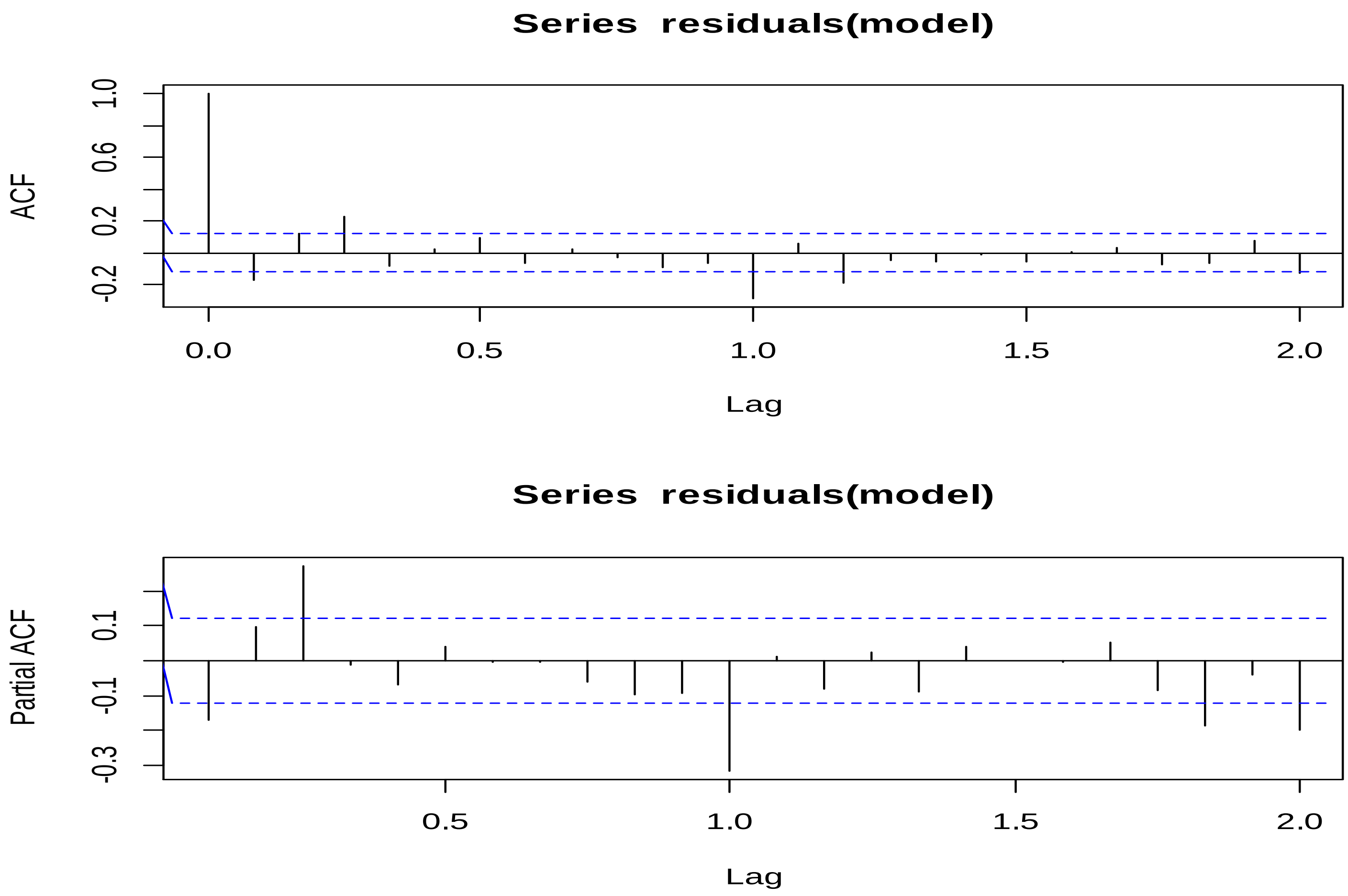


4. Diagnostic checking

- You may prefer to show the correlagram and partial correlogram of the residuals in a single panel of figures:

```
par(mfrow=c(2,1))
acf(residuals(model))
pacf(residuals(model))
acf2(residuals(model))
```

4. Diagnostic checking



4. Diagnostic checking

- Do they seem the residuals of a white noise process?

- How could you improve model specification?

5. Simulations of ARMA(p,q) models

- Simulation is a very useful tool to:
  - Learn about the properties of different models.
  - Obtaining critical values of different tests.
  - Estimation

5. Simulations of ARMA(p,q) models

- Simulation of AR(1) process with

```
y1 <-arima.sim(n=100, list(ar=0.9),
  innov=rnorm(100))
```

- Simulation of AR(2) process with and

```
y2 <- arima.sim(n=1000, list(ar=c(0.6,-0.28)))
```