Practice R1: Time Series in R

February 2024

R is the current standard open software for statistical data analysis. In this practice we will use R for classical analysis of time series. I use the frame RStudio.

We will see

- Framework of RStudio
- Graphical representation of a time series.
- Transformation of a series.
- Linear regression. Estimating a linear trend.
- Estimating the seasonal component.
- Filtering.
- Differencing.

1. Introduction. Reading data in RStudio.

RStudio reads files "name.xlsx". Open the file "Air_Passengers.xlsx".

First of all we have to create the variables of interest. Use the instruction

```
x<-Filename$Variablename
```

In this case we can write

```
x<-Air_Passengers$passengers
t<-Air_Passengers$date
```

We are creating vectors x and t.

The option "packages" allow to incorporate new packages from Internet. We need to charge the package "tseries". After doing this, do

```
xt<-ts(x,start=1949,frequency=12)
xt
```

and see the result.

Function "ls()" give the list of defined objects. Function "rm(name of an object)" allows to remove an object.

There is a useful help function.

This file is also in the set “datasets” under the name AirPassengers and with a time series format. It can be called by *data(AirPassengers)*

Create and see the time series by

AP<- Air Passengers
AP

The instruction “class(AP)” shows the type of file we have at hand. The following instruction makes a summary of the series: *summary(AP)*

We have two identical objects, a time series: xt or AP. We continue with xt

2. Graphical representation.

With the following instructions, we have different option for graphical representation of data:

plot(xt, main=”Airline Passengers”, xlab=”time”, ylab="passengers", type=”o”)

plot(t, x, main=”Airline Passengers”, xlab="time", ylab="passengers", type=”l”)

See the help of “plot” function. See in particular the different option of “type”.

Another interesting option is to use *aggregate* and *cycle* to make a yearly summary. Try:

layout (1:2)

plot(aggregate(xt))

boxplot(xt~cycle(xt))

and see the result. The symbol ~ is written making AltGr 4 Space.

The command layout is a way to present joint graphics. See the help of this command.

3. Transforming a time series

Do

y<-log(x)
y
plot (t, y, xlab="time", ylab="passengers", type="o")

Vector y contains the series of logarithms of the monthly number of passengers. Note that in the graphical representation the apparent no stationarity in variance has disappeared.

R is very powerful representing functions. Do for example

```
t<-1:500
c<-2*cos(2*pi*t/50+0.6*pi)
plot(t, c, type="l")
```

4. Estimating a linear trend.

We have two objects defined, t and x. To obtain the best linear approximation to data we can do

```
t<-1:144
line=lm(y~t)
summary(line)
plot(t,y,type=”o”,xlab=”time”,ylab=”passengers”);abline(line)
```

To extract the trend to the logarithmic data we can do

```
y<-log(x)
lm(y~t)
trend<-0.01005*t+4.81367
trend
z<-y-trend
z
plot.ts(z)
```

Variable z is the residual of the series after log transforming and extracting the linear trend.

5. Estimating the seasonal component.

To eliminate seasonality of z we can do

```
dim(z)<-c(12,12)
z
```

We have converted the series in a matrix where rows are data of the same month.

Now we compute monthly row means, the global mean and the seasonal component.

```
e<-apply(z,1,mean)
ee<-mean(e)
s<-e-ee
plot (s,type=”o”)
```

We have identified the seasonal component. Now we do

```
est<-array(s,144)
est
plot (est,type=”o”)
```

Finally,

```
dim(z)<-144
u<-z-est
u
plot (u,type=”o”)
```

Note that series u is a series without trend nor seasonality. What can we do with it?

6. Filtering.

Given x, the following instruction applies a moving average of order q=1 to the series:

```
v<-filter(x,sides=2,rep(1,3)/3)
```

The function creates the vector (1,1,1). Note that the first and the last values in v series are loosed.

Change progressively 3 by 5 and 7 and see the graphics.

7. Differencing.

Given x, do

```
y<-log(x)
dy<-diff(y)
dy
plot (dy,type=”o”)
```

This is the graphical of first differences.

To obtain differences of order 12 we can do

```
de<-diff(y, lag=12)
```

If we do

```
de<-diff(y,12,2)
```

we are doing second differences of order 12.

Do

```
y<-log(x)
dy12<-diff(y, lag=12)
plot (dy12, xlab=”time”, ylab=”passengers”, type=”o”)
ddy12<-diff(dy12)
plot (ddy12, xlab=”time”, ylab=”passengers”, type=”o”)
```

The last graphical is the series of residuals after applying nabla12 and nabla.

8. Decomposing.

Given a time series like xt we can make the decomposition directly, using only moving average methods. Here it is an example:

```
plot (decompose(xt))
```

Or a more detailed construction:

```
dec<-decompose(xt, type="additive")
plot(dec)
trend<-dec$trend
seasonal<-dec$seasonal
random<-dec$random
layout(1:3)
plot(trend)
plot(seasonal)
plot(random)
```