

To R is Human

R is a computing environment specially made for doing statistics/econometrics.

It is becoming the standard for advanced dealing with empirical data, also in finance.

Good parts

- ▶ It is freely available for download from the internet, and can be installed on a wide range of operating systems, such as Windows, Mac OS, Unix, etc.
- ▶ There is a large number of packages available for specialized purposes, also freely downloadable from the web.

Some issues with R

- ▶ It is command-driven, and learning to use it to its full extent takes some time and effort.
- ▶ The documentation is comprehensive, but not always easy to navigate. It is sometimes necessary to do a bit of googling before one finds examples of how somebody else has performed a particular task.
- ▶ Since anybody can contribute their own specialized packages, there are several packages doing the same (or similar) things, leading to some confusion among the potential package users.
 - Over time may expect to see some convergence to “standard” packages for various things.

Example: Time series handlers.

Input

Data is usually read in, but can also be specified in R:

```
> data <- c(1, 2, 3, 4, 5, 4, 3, 2, 1)
```

Note the `<-`, which is the assignment operator in R. An equality sign can be used in some cases, but to be on the safe side, stick to `<-`.

To get data in otherwise, typically read into tables from text files, see later examples

```
> data <- read.table(file,...)
```

```
> data <- read.csv(file,...)
```

```
> data <- c(1, 2, 3, 4, 5, 4, 3, 2, 1)
```

```
> mean(data)
```

```
[1] 2.777778
```

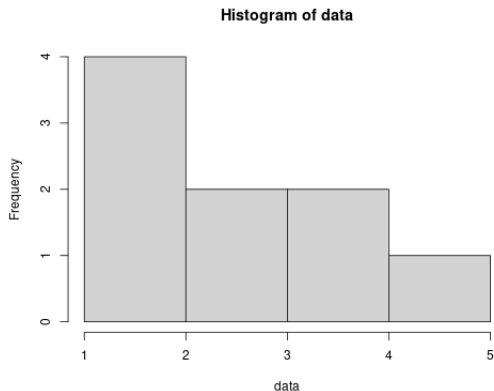
```
> summary(data)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.000	2.000	3.000	2.778	4.000	5.000

Describing data

Pictures always nice, e.g. Histogram

```
> hist(data)
```



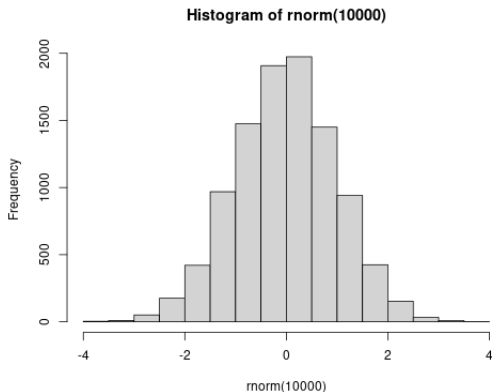
Simulating data

Specify probability distribution, e.g. normal

Simulate 1000 random $N(0,1)$ numbers

```
> rnorm(1000)
```

Result of the simulation



Regression

Consider the standard regression equation

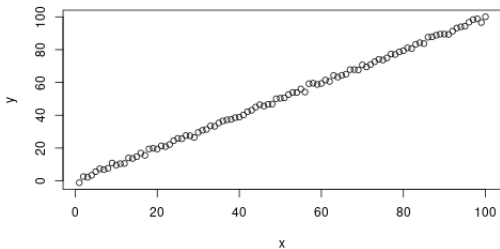
$$y = a + bx + \varepsilon$$

Let us simulate this regression with $a = 0$, $b = 1$ and $\varepsilon \sim N(0, 1)$, where $x = 1, 2, \dots, 100$:

```
> x <- 1:100  
> y=x+rnorm(100)
```

Plot the resulting simulated values

```
> plot(x,y)
```



Perform the regression

```
> regr <- lm(y~x)
```

To actually get the results specify what you want

```
> summary(regr)
```

Call:

```
lm(formula = y ~ x)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.93989	-0.54106	0.03258	0.43836	2.25277

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.065595	0.172075	-0.381	0.704
x	1.000924	0.002958	338.351	<2e-16 ***

Residual standard error: 0.8539 on 98 degrees of freedom

Multiple R-squared: 0.9991, Adjusted R-squared: 0.9991

F-statistic: 1.145e+05 on 1 and 98 DF, p-value: < 2.2e-16

Lots of other interesting output, such as variance covariance matrix

```
> vcov(regr)
```

```
              (Intercept)                x  
(Intercept)  0.0425121439 -6.345096e-04  
x            -0.0006345096  1.256455e-05
```

and ANOVA analysis

```
> anova(regr)
```

Analysis of Variance Table

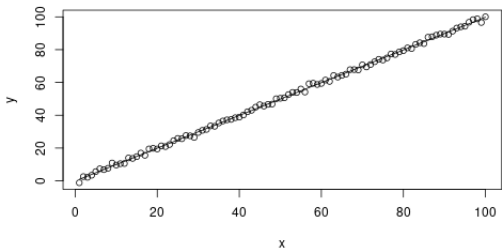
Response: y

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
x	1	83574	83574	79827	< 2.2e-16 ***
Residuals	98	103	1		

Compare observations to fitted values for the regression in the same plot

Note that the plotting comand is in two parts, the scatterplot and the lines:

```
> plot(x,y)
> lines(x,fitted(regr))
```



Once one have results, export them to text processing tools.
For example, for the \LaTeX users, library xtable

```
> library(xtable)
> xtable(regr)
```

Gives the output table

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.0656	0.1721	-0.38	0.7039
x	1.0009	0.0030	338.35	0.0000

Libraries

The `xtable` command:

An example of the use of a *library*, an extension of basic R.

The number of such libraries is very large, and one needs to spend some time on *CRAN* (comprehensive R archive network) searching for it.

Once one have identified the R libraries to get, they need to be installed.

One-time occurrence, and many of the more common R libraries are typically installed by default. (Check your installation)

Otherwise it is a matter of using the command:

```
> install.packages(c("xtable"))
```

Depending on the platform you are at, you will be asked where to download from and where to place the resulting library.

Libraries - setup file

One can also specify every library you ever use in a setup file, Handy when you change computer, etc, to have this sitting in one place.

```
install.packages(c("car"),
                 repos=c("http://cran.uib.no"),
                 lib=c("/usr/lib/R/site-library"))
install.packages(c("zoo"),repos=c("http://cran.uib.no"))
install.packages(c("lmtree"),repos=c("http://cran.uib.no"))
install.packages(c("sandwich"),repos=c("http://cran.uib.no"))
install.packages(c("gmm"),repos=c("http://cran.uib.no"))
install.packages(c("urca"),repos=c("http://cran.uib.no"))
install.packages(c("strucchange"),repos=c("http://cran.uib.no"))
install.packages(c("dynlm"),repos=c("http://cran.uib.no"))
install.packages(c("vars"),repos=c("http://cran.uib.no"))
....
```

R Libraries for time series

Example of library in R: Handling and organizing time series data, one of the most common tasks confronting financial (and other) econometricians.

A time series is an ordered series of observations indexed by a date. If there are multiple observations on a date, also use the time of observation as well as the date.

There are a number of operations typically involved when dealing with time series:

- ▶ Aligning on date - Two or more time series, pick only those dates where have observations of all, and place the observations on the same place in each time series
- ▶ Changing frequency - E.g. from daily to monthly. Issue: Monthly average or last observation?
- ▶ Lead - Lag operations
- ▶ ...

Common operations, belong in a library.

Good part of R: Of course there is a library.

“Bad” part of R: There are lots of different libraries, not always compatible.

Illustrate some of the more common libraries (There are many others).

Most basic: `ts`, part of standard *R*. can only hold data at given frequencies (monthly, quarterly, annual)

Most common library: `zoo`: Indexing by date. Special case when having regular (fixed frequency) observations: `zooreg`

Integrate various time series libraries, try to make interoperable: `xts`. Very good for picking subperiods.

Will illustrate some usage using a time series of stock prices for the Norwegian company *Akers Mekaniske Verksted*

30	12	1910	500
22	12	1911	690
20	12	1912	750
31	12	1913	805
31	12	1914	640
31	12	1915	760

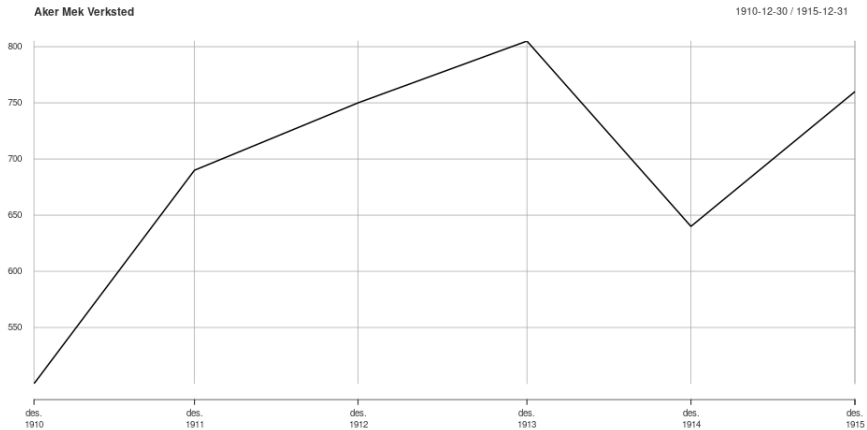
In R, will be reading the file `aker.txt`:

Date	Price
30.12.1910	500
22.12.1911	690
20.12.1912	750
31.12.1913	805
31.12.1914	640
31.12.1915	760

Using the standard time series package ts:

```
> data <- read.table("../data/aker.txt",header=TRUE)
> aker <- ts(data$Price,frequency=1,start=1910)
> print(aker)
Time Series:
Start = 1910
End = 1915
Frequency = 1
[1] 500 690 750 805 640 760
> plot(aker,main="Aker Mek Verksted",ylab="Price",xlab="Year")
```

Results in the following plot:

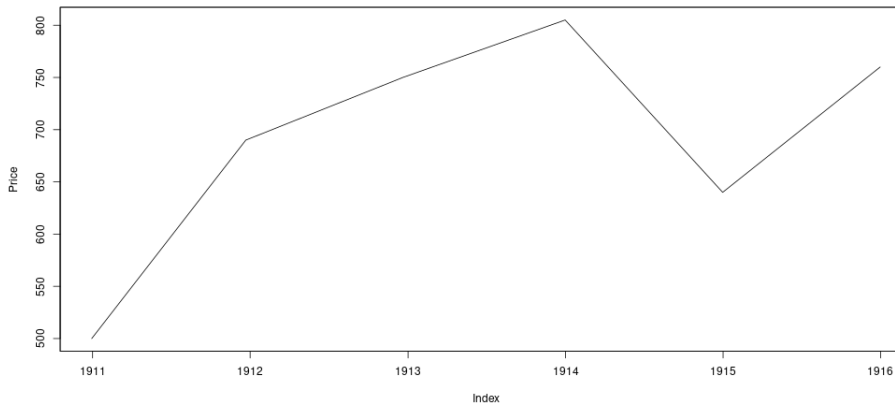


Doing the same in zoo:

```
> library(zoo)
> akerZ <- read.zoo("../data/aker.txt",format="%d.%m.%Y",he
> akerZ
1910-12-30 1911-12-22 1912-12-20 1913-12-31 1914-12-31 1915
           500           690           750           805           640

> plot(akerZ,main="Aker Mek Verksted",ylab="Price");
```


Aker Mek Verksted



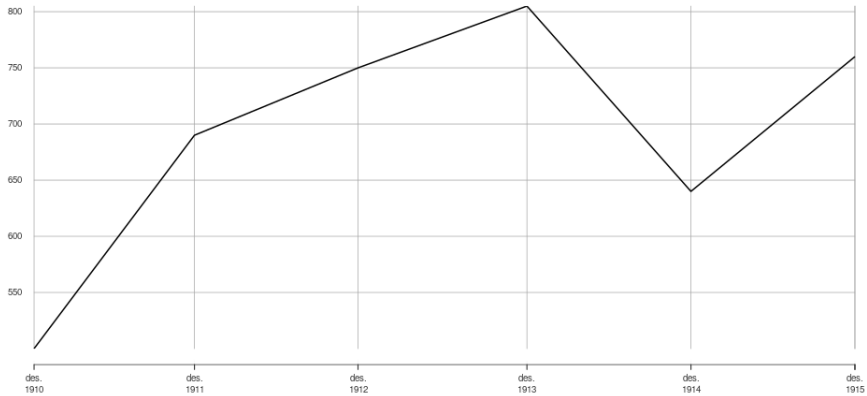
Getting an xts object is typically done by converting another type of time series object, such as zoo:

```
> library(xts)
> akerX <- as.xts(akerZ)
> print(akerX)
           [,1]
1910-12-30  500
1911-12-22  690
1912-12-20  750
1913-12-31  805
1914-12-31  640
1915-12-31  760

> plot(akerX,main="Aker Mek Verksted",ylab="Price");
```

Aker Mek Verksted

1910-12-30 / 1915-12-31



Using R to access data directly

For some internet sources of data, there are routines that lets R directly access and import this data, almost magically.

One such routine is in the library `quantmod`, which can import various time series of interest for finance and economics, such as

- ▶ Yahoo finance – individual stocks, etc
- ▶ Google finance – similar to yahoo finance
- ▶ FRED – St Louis Fed database of macroeconomic variables
- ▶ OANDA – Foreign exchange quotes

Using R to access data directly, ctd

If one knows the names of the series it is a simple matter of the following sequence of R commands to get the time series of Ford (F) from Yahoo Finance and JPY/USD exchange rates (DEXJPUS) from the St Louis Fed Fred database

```
library(quantmod)
getSymbols("F")
getSymbols("DEXJPUS",src="FRED")
```

Note that yahoo finance is the default data source.

Using R to access data directly, ctd

How to access the data afterwards:

Getting prices of Ford

```
> library(quantmod)
```

```
> getSymbols("F")
```

```
> head(F)
```

	F.Open	F.High	F.Low	F.Close	F.Volume	F.Adjusted
2007-01-03	7.56	7.67	7.44	7.51	78652200	7.12
2007-01-04	7.56	7.72	7.43	7.70	63454900	7.30
2007-01-05	7.72	7.75	7.57	7.62	40562100	7.23
2007-01-08	7.63	7.75	7.62	7.73	48938500	7.33
2007-01-09	7.75	7.86	7.73	7.79	56732200	7.39
2007-01-10	7.79	7.79	7.67	7.73	42397100	7.33

```
> summary(F)
```

Index	F.Open	F.High	F.Low
Min. :2007-01-03	Min. : 1.31	Min. : 1.55	Min. : 1
1st Qu.:2008-10-26	1st Qu.: 7.60	1st Qu.: 7.74	1st Qu.: 7
Median :2010-08-21	Median :10.51	Median :10.68	Median :10
Mean :2010-08-21	Mean :10.42	Mean :10.56	Mean :10
3rd Qu.:2012-06-14	3rd Qu.:13.42	3rd Qu.:13.59	3rd Qu.:13

Using R to access data directly, ctd

```
> getSymbols("DEXJPUS",src="FRED")
```

```
> head(DEXJPUS)
```

```
          DEXJPUS  
1971-01-04  357.73  
1971-01-05  357.81  
1971-01-06  357.86  
1971-01-07  357.87  
1971-01-08  357.82  
1971-01-11  357.95
```

```
> summary(DEXJPUS)
```

Index	DEXJPUS
Min. :1971-01-04	Min. : 75.72
1st Qu.:1981-10-27	1st Qu.:108.24
Median :1992-08-19	Median :127.80
Mean :1992-08-19	Mean :165.52
3rd Qu.:2003-06-12	3rd Qu.:233.50
Max. :2014-04-04	Max. :358.44
	NA's :433

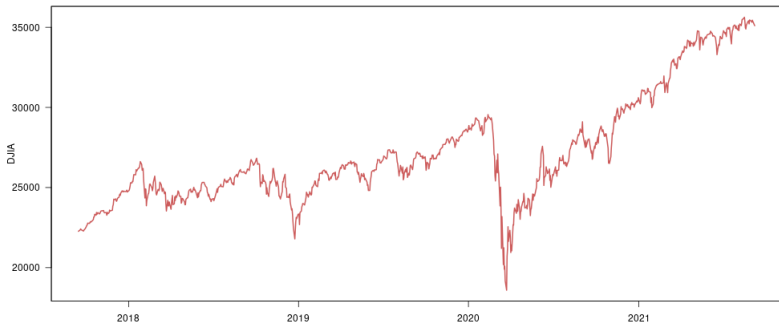
Updating your presentations

```
outdir <- ".././results/2018_03_plot_current/"
library(quantmod)
library(xts)
getSymbols("^DJI")
dj <- xts(as.vector(DJI$DJI.Close),
          order.by=index(DJI))
names(dj) <- "dj"
n <- length(dj)
dj <- dj[(n-1000):n,]
filename <- paste0(outdir,"djia_recent_evolution.png")
png(filename, width=1000, height=500)
plot.zoo(dj,
         col=c("indianred"),
         main="DJIA, recent evolution",
         ylab="DJIA",
         lwd=2,
         lty=1,
         las=1,
         xlab="")
dev.off()
```

10

20

DJIA, recent evolution



Documentation

Using R, need to look up documentation.

Options

- ▶ When learning R, keep a couple of books on your shelf.
Examples: *The R book* / *R in a nutshell*
- ▶ Help in R: “?” command - but need to know command
- ▶ Internet: CRAN
 - ▶ Manuals
 - ▶ Task views
 - ▶ Package documentation: How to use the functions provided by the package.
- ▶ Internet: Googling