UNIVERSITÄT
BERN

Statistical methods for archaeological data analysis I: Basic methods

02 - Introduction into R

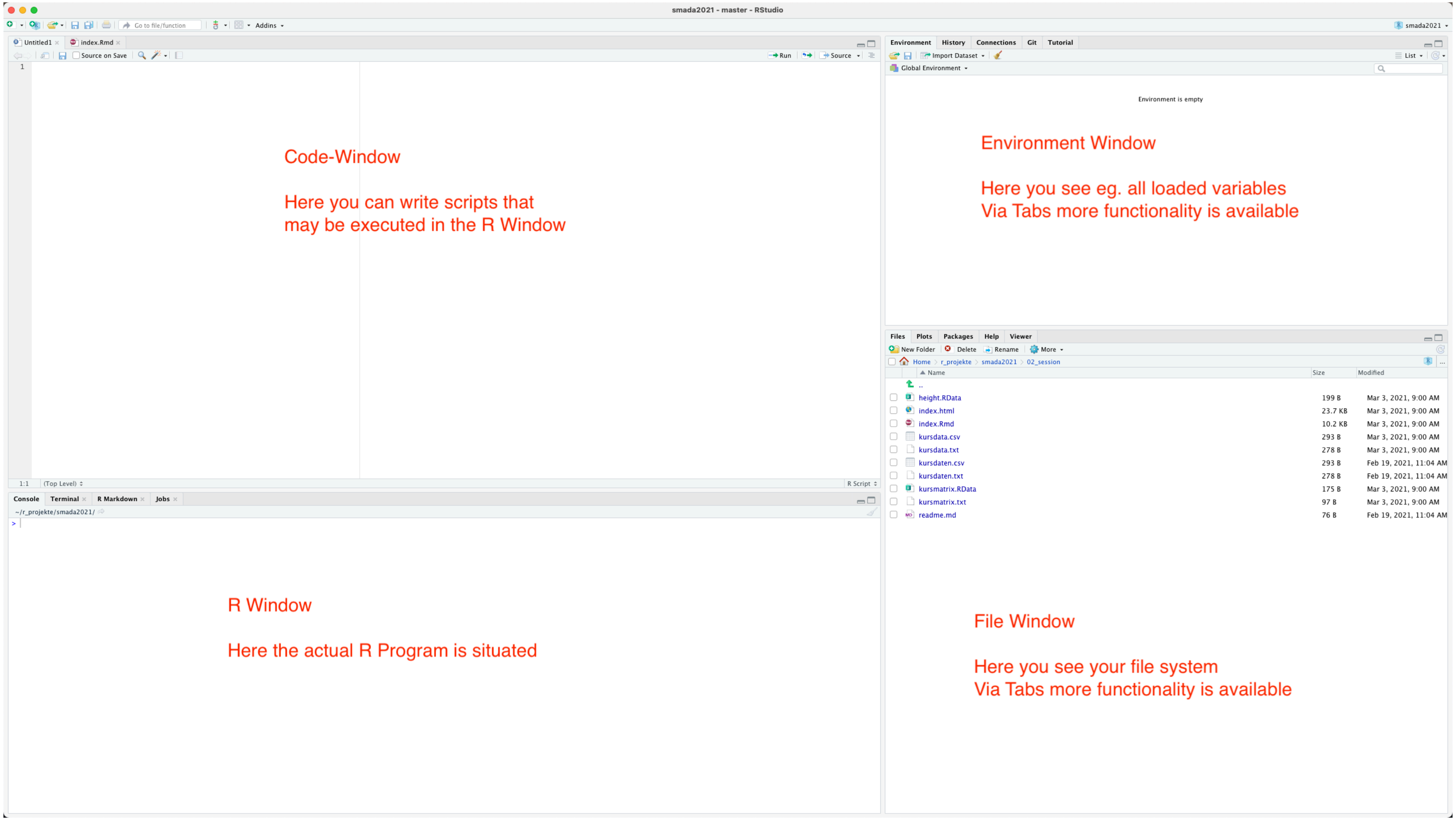
Martin Hinz

Institut für Archäologische Wissenschaften, Universität Bern

25.02.2025

You can download a pdf of this presentation.

Start R-Studio



Environment is empty

```
~/r_projekte/smada2021/
>
```

Name	Size	Modified
..		
height.RData	199 B	Mar 3, 2021, 9:00 AM
index.html	23.7 KB	Mar 3, 2021, 9:00 AM
index.Rmd	10.2 KB	Mar 3, 2021, 9:00 AM
kursdata.csv	293 B	Mar 3, 2021, 9:00 AM
kursdata.txt	278 B	Mar 3, 2021, 9:00 AM
kursdaten.csv	293 B	Feb 19, 2021, 11:04 AM
kursdaten.txt	278 B	Feb 19, 2021, 11:04 AM
kursmatrix.RData	175 B	Mar 3, 2021, 9:00 AM
kursmatrix.txt	97 B	Mar 3, 2021, 9:00 AM
readme.md	76 B	Feb 19, 2021, 11:04 AM

Code-Window

Here you can write scripts that may be executed in the R Window

Environment Window

Here you see eg. all loaded variables
Via Tabs more functionality is available

R Window

Here the actual R Program is situated

File Window

Here you see your file system
Via Tabs more functionality is available

Using R

Start of the system:

After R is started, you end on the prompt.

```
>
```

Change the working directory:

```r
getwd() # or something else

setwd("U:\R") # or something else
```

Change the path according to your needs

R as calculator

Simplest way of use:

```
2+2
```

```
## [1] 4
```

```
2^2
```

```
## [1] 4
```

Multiple commands are separated by ;

```
(1 - 2) * 3; 1 - 2 * 3
```

```
## [1] -3
```

```
## [1] -5
```

R as calculator

Using functions:

```r
sqrt(2) #square root
```

```
## [1] 1.414214
```

```r
log(10) #logarith base e
```

```
## [1] 2.302585
```

```r
log(10, 10) #logarith base 10, like log(10, base=10)
```

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

Getting help

Call of the help function:

```
help(sqrt)
```

Even simpler?

```
? sqrt
```

Searching the help:

```
help.search('logarithm')
```

Assignment of data to variables

Naming variables for Values (Assignment):

```
x <- 2 # no message will be given back

x
```

```
## [1] 2
```

```
pi # build in variable
```

```
## [1] 3.141593
```

Arrow or equal sign?

Classic assignment symbol in R is the arrow. Also possible:

```
x=2
```

Both are possible. Matter of tast. <- is clearer, I am using it that way

Working with variables

Display of already uses variables:

```
ls()
```

```
## [1] "x"
```

Delete a variable:

```
rm(x) # no message will be given back
ls()
```

```
## character(0)
```

Using variables

Calculations with variables:

```
x <- 2
y <- 2 * x
z <- sqrt(x) # no message will be given back
```

```
ls()
```

```
## [1] "x" "y" "z"
```

```
y
```

```
## [1] 4
```

```
z
```

```
## [1] 1.414214
```

Exercise variables

Calculation of a circle:

Given is a circle with the radius r=5. Calculate the diameter d (2 * r), the circumference u (2 * π * r) and the area a (π * r^2).

Add area a and circumference u, assign the result to the variable v and delete u and a.

Scalars, vectors, matrices, data frames

Data types in R

Scalar

A single number or date

```
pi
```

```
## [1] 3.141593
```

Vector

A row of numbers or data

```
ls()
```

```
## [1] "x" "y" "z"
```

Scalars, vectors, matrices, data frames

Data types in R

Matrix:

A table of data of the same kind

```
euro.cross
```

```
##              ATS         BEF         DEM         ESP         FIM         FRF
## ATS  1.000000000  2.93161486 0.142135709  12.0917422 0.432093050 0.476702543
## BEF  0.341108927  1.00000000 0.048483759   4.1246012 0.147390797 0.162607493
## DEM  7.035529673 20.62546336 1.000000000  85.0718109 3.040003477 3.353854885
## ESP  0.082701069  0.24244768 0.011754775   1.0000000 0.035734557 0.039423810
## FIM  2.314316324  6.78468413 0.328946992  27.9841163 1.000000000 1.103240477
## FRF  2.097744212  6.14977811 0.298164361  25.3653822 0.906420695 1.000000000
## IEP 17.471976881 51.22110711 2.483391826 211.2666399 7.549519785 8.328935807
## ITL  0.007106602  0.02083382 0.001010102   0.0859312 0.003070713 0.003387735
## LUF  0.341108927  1.00000000 0.048483759   4.1246012 0.147390797 0.162607493
## NLG  6.244151907 18.30544854 0.887516960  75.5026750 2.698054644 2.976603092
## PTE  0.068636087  0.20121457 0.009755639   0.8299299 0.029657176 0.032718997
##              IEP         ITL         LUF         NLG         PTE
## ATS 0.0572345080  140.714229  2.93161486 0.160149851  14.5695951
## BEF 0.0195232016   47.998880  1.00000000 0.054628544   4.9698190
## DEM 0.4026750791  989.999131 20.62546336 1.126739032 102.5048189
## ESP 0.0047333550   11.637217  0.24244768 0.013244564   1.2049211
```

Scalars, vectors, matrices, data frames

Data types in R

Data frame:

A table of data of different kind

```
mtcars
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
```

Download data for further tasks

- height.RData
- kursmatrix.txt
- kursdata.txt
- kursdata.csv

Data import through reading of files

remember:

```
getwd()
setwd("my/location/of/my/working/directory")
```

Simple text file:

```
kursmatrix <- matrix(scan("kursmatrix.txt"),ncol=2)
```

Data frame as simple text file:

```
kursdata <- read.table("kursdata.txt")
```

Data frame as csv file:

```
kursdata <- read.csv2("kursdata.csv")
```

Read with rownames

```
kursdaten <- read.csv2("kursdata.csv",row.names = 1)
```

Using c() for data entry

Assignment of values to a vector:

```
places <- c("Leubingen", "Melz", "Bruszczewo")
```

```
categories <- c("Grab", "Hort", "Siedlung")
categories
```

```
## [1] "Grab"     "Hort"     "Siedlung"
```

```
c(places, categories)
```

```
## [1] "Leubingen"  "Melz"       "Bruszczewo" "Grab"       "Hort"
## [6] "Siedlung"
```

Naming the positions in a vector

```
names(places)<-categories
places
```

```
##         Grab         Hort     Siedlung
##  "Leubingen"       "Melz" "Bruszczewo"
```

Functions on vectors [1]

Data:

```
load("height.RData")
height
```

```
##         Bilbo         Frodo
##           181           170
##       Aragorn       Boromir
##           185           163
##        Pippin  Gandalf grey
##           175           163
##         Merry       Samwise
##           162           172
##       Theoden         Eowyn
##           172           180
##         Arwen Gandalf white
##           187           158
##         Gimly        Gollum
##           184           156
```

```
# Sum:
sum(height)
```

```
## [1] 2408
```

```
# Count:
length(height)
```

```
## [1] 14
```

```
# Mean:
sum(height)/length(height)
```

```
## [1] 172
```

```
# Or more convenient:
mean(height)
```

```
## [1] 172
```

Functions on vectors [2]

```
# sort:
sort(height)
```

```
##         Gollum Gandalf white         Merry       Boromir  Gandalf grey 
##            156           158           162           163           163 
##          Frodo       Samwise       Theoden        Pippin         Eowyn 
##            170           172           172           175           180 
##          Bilbo         Gimly       Aragorn         Arwen 
##            181           184           185           187
```

```
# minimum:
min(height)
```

```
## [1] 156
```

```
# maximum:
max(height)
```

```
## [1] 187
```

```
# Or more convenient:
range(height)
```

```
## [1] 156 187
```

Functions on vectors [3]

Change of the values through calculation:

```
height.in.m <- height/100
height.in.m
```

```
##         Bilbo         Frodo       Aragorn       Boromir        Pippin
##          1.81          1.70          1.85          1.63          1.75
##  Gandalf grey         Merry       Samwise       Theoden         Eowyn
##          1.63          1.62          1.72          1.72          1.80
##         Arwen Gandalf white         Gimly        Gollum
##          1.87          1.58          1.84          1.56
```

but:

```
test<-c(1,2,3,4,5,6,7,8,9,10,11,12,13,14)
height.in.m + test
```

```
##         Bilbo         Frodo       Aragorn       Boromir        Pippin
##          2.81          3.70          4.85          5.63          6.75
##  Gandalf grey         Merry       Samwise       Theoden         Eowyn
##          7.63          8.62          9.72         10.72         11.80
##         Arwen Gandalf white         Gimly        Gollum
##         12.87         13.58         14.84         15.56
```

Exercise vectors

Data collection ceramics:

An excavation produced the following numbers of flint artefacts:

flakes	blades	cores	debris
506	104	30	267

Assign the values to a named vector, calculate the proportion of the artefacts and sort the vector according to their percentage

During the data collection on box with artefacts was missing, the following numbers has to be added to the vector:

flakes	blades	cores	debris
52	24	15	83

Moreover were 10 items each artefact type missing. Make a vector for the box, add it and the 10 missing to the original data and repeat the calculations.

Sequences and repeated data

Simple sequence:

```
1:10
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

Sequence with start value, end value and step size:

```
seq(1,10,by=2)
```

```
## [1] 1 3 5 7 9
```

```
seq(1,20,length=5)
```

```
## [1]  1.00  5.75 10.50 15.25 20.00
```

Repeated data:

```
rep(1,10)
```

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

```
rep(1:3,3)
```

Data access by index

Access by position:

```
height[1]
```

```
## Bilbo
##   181
```

```
height[5]
```

```
## Pippin
##    175
```

```
height[1:3]
```

```
##   Bilbo   Frodo Aragorn
##     181     170     185
```

```
height[-(1:3)]
```

```
##       Boromir        Pippin  Gandalf grey         Merry       Samwise
##           163           175           163           162           172
##       Theoden         Eowyn         Arwen Gandalf white         Gimly
##           172           180           187           158           184
##        Gollum
##           156
```

Access by name:

```
height["Frodo"]
```

```
## Frodo
##   170
```

Data entry into vectors

Entry by position:

```
height
```

```
##         Bilbo         Frodo       Aragorn       Boromir        Pippin
##           181           170           185           163           175
##  Gandalf grey         Merry       Samwise       Theoden         Eowyn
##           163           162           172           172           180
##         Arwen Gandalf white         Gimly        Gollum
##           187           158           184           156
```

```
height[1] <- 168
height
```

```
##         Bilbo         Frodo       Aragorn       Boromir        Pippin
##           168           170           185           163           175
##  Gandalf grey         Merry       Samwise       Theoden         Eowyn
##           163           162           172           172           180
##         Arwen Gandalf white         Gimly        Gollum
##           187           158           184           156
```

Entry by name:

```
height["Bilbo"] <- 181
height
```

```
##         Bilbo         Frodo       Aragorn       Boromir        Pippin
##           181           170           185           163           175
##  Gandalf grey         Merry       Samwise       Theoden         Eowyn
##           163           162           172           172           180
##         Arwen Gandalf white         Gimly        Gollum
```

Logical values

true/false-values:

```
pi>4
```

```
## [1] FALSE
```

```
height > 175
```

```
##         Bilbo         Frodo       Aragorn       Boromir        Pippin
##          TRUE         FALSE          TRUE         FALSE         FALSE
##  Gandalf grey         Merry       Samwise       Theoden         Eowyn
##         FALSE         FALSE         FALSE         FALSE          TRUE
##         Arwen Gandalf white         Gimly        Gollum
##          TRUE         FALSE          TRUE         FALSE
```

Logical values

Can be used for selection of values:

```
height[height>175]
```

```
##   Bilbo Aragorn   Eowyn   Arwen   Gimly
##     181     185     180     187     184
```

```
which(height>175)
```

```
##   Bilbo Aragorn   Eowyn   Arwen   Gimly
##       1       3      10      11      13
```

```
sum(height>175)/length(height)
```

```
## [1] 0.3571429
```

Factors

For encoding nominal values:

```r
sex <- factor(c("m", "m", "m", "m", "m", "m", "m",
                "m", "m", "f", "f", "m", "m", "m"))

sex
```

```
##  [1] m m m m m m m m m f f m m m
## Levels: f m
```

missing (NA) values

Problem: values are missing

```
height["Arwen"] <- 0

mean(height)
```

```
## [1] 158.6429
```

```
sum(height)/13
```

```
## [1] 170.8462
```

therefore: code as N(ot)A(vailable)

```
height["Arwen"] <- NA

mean(height)
```

```
## [1] NA
```

```
mean(height, na.rm=T)
```

```
## [1] 170.8462
```

matrices [1]

Data of the same kind (numbers, factors...)

```
kursmatrix
```

```
##       [,1] [,2]
##  [1,]   39  181
##  [2,]   34  170
##  [3,]   23  185
##  [4,]   38  163
##  [5,]   23  175
##  [6,]   21  163
##  [7,]   23  162
##  [8,]   31  172
##  [9,]   25  172
## [10,]   31  180
## [11,]   24  187
## [12,]   23  158
## [13,]   23  184
## [14,]   39  156
```

```
rownames(kursmatrix) <- names(height)
colnames(kursmatrix)<-c("height","age"
kursmatrix
```

```
##                height age
## Bilbo              39 181
## Frodo              34 170
## Aragorn            23 185
## Boromir            38 163
## Pippin             23 175
## Gandalf grey       21 163
## Merry              23 162
## Samwise            31 172
## Theoden            25 172
## Eowyn              31 180
## Arwen              24 187
## Gandalf white      23 158
## Gimly              23 184
## Gollum             39 156
```

matrices [2]

Operations on matrices

```
kursmatrix / 100
```

```
##                height  age
## Bilbo           0.39 1.81
## Frodo           0.34 1.70
## Aragorn         0.23 1.85
## Boromir         0.38 1.63
## Pippin          0.23 1.75
## Gandalf grey    0.21 1.63
## Merry           0.23 1.62
## Samwise         0.31 1.72
## Theoden         0.25 1.72
## Eowyn           0.31 1.80
## Arwen           0.24 1.87
## Gandalf white   0.23 1.58
## Gimly           0.23 1.84
## Gollum          0.39 1.56
```

```
kursmatrix[, 1] / 100
```

```
##         Bilbo         Frodo       Aragorn
##          0.39          0.34          0.23
##  Gandalf grey         Merry       Samwise
##          0.21          0.23          0.31
##         Arwen Gandalf white         Gimly
##          0.24          0.23          0.23
```

```
kursmatrix / c(1:14, rep(2, 14))
```

```
##                  height  age
## Bilbo         39.000000 90.5
## Frodo         17.000000 85.0
## Aragorn        7.666667 92.5
## Boromir        9.500000 81.5
## Pippin         4.600000 87.5
## Gandalf grey   3.500000 81.5
## Merry          3.285714 81.0
## Samwise        3.875000 86.0
## Theoden        2.777778 86.0
## Eowyn          3.100000 90.0
```

Data frames [1]

```
kursdata <-
  data.frame(age = kursmatrix[,2],
             height = kursmatrix[,1],
             sex=sex)
kursdata
```

```
##               age height sex
## Bilbo         181     39   m
## Frodo         170     34   m
## Aragorn       185     23   m
## Boromir       163     38   m
## Pippin        175     23   m
## Gandalf grey  163     21   m
## Merry         162     23   m
## Samwise       172     31   m
## Theoden       172     25   m
## Eowyn         180     31   f
## Arwen         187     24   f
## Gandalf white 158     23   m
## Gimly         184     23   m
## Gollum        156     39   m
```

```
kursdata[,"age"]
```

```
##  [1] 181 170 185 163 175 163 162 172 172 18
```

```
kursdata$age
```

```
##  [1] 181 170 185 163 175 163 162 172 172 18
```

Data frames [2]

Operation on data frames

```
kursdata$height / 100
```

```
##  [1] 0.39 0.34 0.23 0.38 0.23 0.21 0.23 0.31 0.25 0.31 0.24 0.23 0.23 0.39
```

```
summary(kursdata)
```

```
##       age             height        sex
##  Min.   :156.0   Min.   :21.00   f: 2
##  1st Qu.:163.0   1st Qu.:23.00   m:12
##  Median :172.0   Median :24.50
##  Mean   :172.0   Mean   :28.36
##  3rd Qu.:180.8   3rd Qu.:33.25
##  Max.   :187.0   Max.   :39.00
```

```
tapply(kursdata$height, kursdata$sex, mean, na.rm=T)
```

```
##    f    m
## 27.5 28.5
```

Build in datasets

```
data()
```

```
Data sets in package 'datasets':

AirPassengers           Monthly Airline Passenger Numbers 1949-1960
BJsales                 Sales Data with Leading Indicator
BJsales.lead (BJsales)
                        Sales Data with Leading Indicator
BOD                     Biochemical Oxygen Demand
CO2                     Carbon Dioxide Uptake in Grass Plants
ChickWeight             Weight versus age of chicks on different diets
DNase                   Elisa assay of DNase
EuStockMarkets          Daily Closing Prices of Major European Stock
                        Indices, 1991-1998
Formaldehyde            Determination of Formaldehyde
HairEyeColor            Hair and Eye Color of Statistics Students
Harman23.cor            Harman Example 2.3
Harman74.cor            Harman Example 7.4
Indometh                Pharmacokinetics of Indomethacin
InsectSprays            Effectiveness of Insect Sprays
JohnsonJohnson          Quarterly Earnings per Johnson & Johnson Share
LakeHuron               Level of Lake Huron 1875-1972
LifeCycleSavings        Intercountry Life-Cycle Savings Data
Loblolly                Growth of Loblolly pine trees
Nile                    Flow of the River Nile
Orange                  Growth of Orange Trees
```

Data export through save

Simple text file:

```
write(kursmatrix,"kursmatrix.txt")
```

Data frame as simple text file:

```
write.table(kursdata,"kursdata.txt")
```

Data frame as csv file:

```
write.csv2(kursdata,"kursdata.csv")
```

Attention: decimal separator is . not ,

```
kursdata$height <- kursdata$height/100
write.csv(kursdata,"kursdata.csv")
```

problems with importing such csv into e.g. Excel therefore:

```
write.csv2(kursdata,"kursdata.csv")
```

R <-> Excel

Always save as csv

There are packages for R to read and write Excel files but for them additional software (Perl, Python e.a.) is neccessary