UNIVERSITÄT BERN

Statistical methods for archaeological data analysis I: Basic methods

02 - Introduction into R

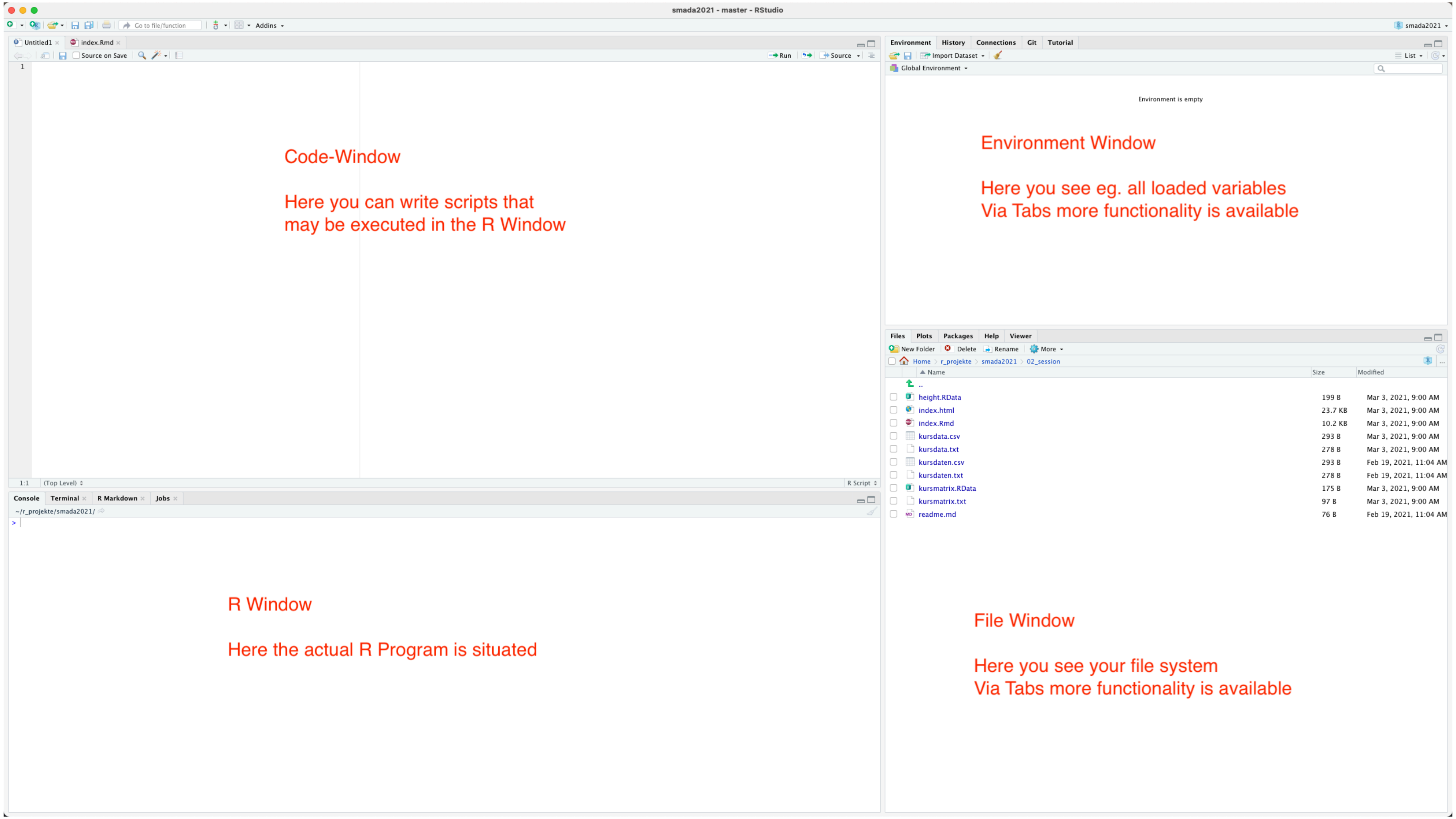
Martin Hinz

Institut für Archäologische Wissenschaften, Universität Bern

03.03.2021

You can download a pdf of this presentation.

Start R-Studio



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

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

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:

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

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

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

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

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

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

Getting help

Call of the help function:

```r
help(sqrt)
```

Even simpler?

```r
? sqrt
```

Searching the help:

```r
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
```

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
```

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("kursdaten.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
```

```
## Matthias   Jannick   Nicolas
##      181       170       185
##   Silvia      Till      Anna
##      163       175       163
##   Ilaria     Sarah     Clara
##      162       172       172
##    Alain    Adrian    Marlen
##      180       187       158
##  Michael    Helena   Nephele
##      184       156       168
```

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

```
## [1] 2576
```

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

```
## [1] 15
```

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

```
## [1] 171.7333
```

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

```
## [1] 171.7333
```

Functions on vectors [2]

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

```
##   Helena    Marlen   Ilaria   Silvia     Anna  Nephele  Jannick    Sarah
##      156       158      162      163      163      168      170      172
##    Clara      Till    Alain Matthias  Michael  Nicolas   Adrian
##      172       175      180      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:

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

```
## Matthias  Jannick  Nicolas   Silvia     Till     Anna   Ilaria    Sarah
##     1.81     1.70     1.85     1.63     1.75     1.63     1.62     1.72
##    Clara    Alain   Adrian   Marlen  Michael   Helena  Nephele
##     1.72     1.80     1.87     1.58     1.84     1.56     1.68
```

but:

```r
test<-c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)
height.in.m + test
```

```
## Matthias  Jannick  Nicolas   Silvia     Till     Anna   Ilaria    Sarah
##     2.81     3.70     4.85     5.63     6.75     7.63     8.62     9.72
##    Clara    Alain   Adrian   Marlen  Michael   Helena  Nephele
##    10.72    11.80    12.87    13.58    14.84    15.56    16.68
```

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
```

Data access by index

Access by position:

```
height[1]
```

```
## Matthias
##      181
```

```
height[5]
```

```
## Till
##  175
```

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

```
## Matthias  Jannick  Nicolas
##      181      170      185
```

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

```
##  Silvia    Till    Anna  Ilaria   Sarah   Clara   Alain  Adrian  Marlen Michael
##     163     175     163     162     172     172     180     187     158     184
##  Helena Nephele
##     156     168
```

Access by name:

```
height["Clara"]
```

```
## Clara
##   172
```

Data entry into vectors

Entry by position:

```
height
```

```
## Matthias  Jannick  Nicolas   Silvia     Till     Anna   Ilaria    Sarah
##      181      170      185      163      175      163      162      172
##    Clara    Alain   Adrian   Marlen  Michael   Helena  Nephele
##      172      180      187      158      184      156      168
```

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

```
## Matthias  Jannick  Nicolas   Silvia     Till     Anna   Ilaria    Sarah
##      168      170      185      163      175      163      162      172
##    Clara    Alain   Adrian   Marlen  Michael   Helena  Nephele
##      172      180      187      158      184      156      168
```

Entry by name:

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

```
## Matthias  Jannick  Nicolas   Silvia     Till     Anna   Ilaria    Sarah
##      168      170      185      163      181      163      162      172
##    Clara    Alain   Adrian   Marlen  Michael   Helena  Nephele
##      172      180      187      158      184      156      168
```

Logical values

true/false-values:

```
pi>4
```

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

```
height > 175
```

```
## Matthias  Jannick  Nicolas   Silvia     Till     Anna   Ilaria    Sarah 
##    FALSE    FALSE     TRUE    FALSE     TRUE    FALSE    FALSE    FALSE 
##    Clara    Alain   Adrian   Marlen  Michael   Helena  Nephele 
##    FALSE     TRUE     TRUE    FALSE     TRUE    FALSE    FALSE 
```

Logical values

Can be used for selection of values:

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

```
## Nicolas    Till   Alain  Adrian Michael
##     185     181     180     187     184
```

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

```
## Nicolas    Till   Alain  Adrian Michael
##       3       5      10      11      13
```

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

```
## [1] 0.3333333
```

Factors

For encoding nominal values:

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

sex
```

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

missing (NA) values

Problem: values are missing

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

mean(height)
```

```
## [1] 160.7333
```

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

```
## [1] 185.4615
```

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

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

mean(height)
```

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

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

```
## [1] 172.2143
```

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
## [15,]   21  168
```

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

```
##            height age
## Matthias       39 181
## Jannick        34 170
## Nicolas        23 185
## Silvia         38 163
## Till           23 175
## Anna           21 163
## Ilaria         23 162
## Sarah          31 172
## Clara          25 172
## Alain          31 180
## Adrian         24 187
## Marlen         23 158
## Michael        23 184
## Helena         39 156
```

matrices [2]

Operations on matrices

```
kursmatrix / 100
```

```
##          height  age
## Matthias   0.39 1.81
## Jannick    0.34 1.70
## Nicolas    0.23 1.85
## Silvia     0.38 1.63
## Till       0.23 1.75
## Anna       0.21 1.63
## Ilaria     0.23 1.62
## Sarah      0.31 1.72
## Clara      0.25 1.72
## Alain      0.31 1.80
## Adrian     0.24 1.87
## Marlen     0.23 1.58
## Michael    0.23 1.84
## Helena     0.39 1.56
## Nephele    0.21 1.68
```

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

```
## Matthias  Jannick  Nicolas   Silvia     T
##     0.39     0.34     0.23     0.38     0
##    Clara    Alain   Adrian   Marlen  Mich
##     0.25     0.31     0.24     0.23     0
```

```
kursmatrix / c(1:15, rep(2, 15))
```

```
##             height  age
## Matthias 39.000000 90.5
## Jannick  17.000000 85.0
## Nicolas   7.666667 92.5
## Silvia    9.500000 81.5
## Till      4.600000 87.5
## Anna      3.500000 81.5
## Ilaria    3.285714 81.0
## Sarah     3.875000 86.0
## Clara     2.777778 86.0
```

Data frames [1]

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

```
##          age height sex
## Matthias 181     39   m
## Jannick  170     34   m
## Nicolas  185     23   m
## Silvia   163     38   f
## Till     175     23   m
## Anna     163     21   f
## Ilaria   162     23   f
## Sarah    172     31   f
## Clara    172     25   f
## Alain    180     31   m
## Adrian   187     24   m
## Marlen   158     23   f
## Michael  184     23   m
## Helena   156     39   f
```

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

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

```
kursdata$age
```

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

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 0.21
```

```
summary(kursdata)
```

```
##       age            height          sex
##  Min.   :156.0   Min.   :21.00   f:8
##  1st Qu.:163.0   1st Qu.:23.00   m:7
##  Median :172.0   Median :24.00
##  Mean   :171.7   Mean   :27.87
##  3rd Qu.:180.5   3rd Qu.:32.50
##  Max.   :187.0   Max.   :39.00
```

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

```
##        f        m
## 27.62500 28.14286
```

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
```

Data export through save

Simple text file:

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

Data frame as simple text file:

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

Data frame as csv file:

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

Attention: decimal separator is . not ,

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

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

```r
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