Universiteit Utrecht

Workshop Data Handling in R

09.30-09.40 : Raw data handling: steps to think about, which we will cover in the Jupyter notebook. Setting the stage! (powerpoint presentation)

09.40-10.50 : Practical Data handling in R (via Jupyter notebook)

- 10 minute coffee break-

11.00-11.20 : Good code formatting and commenting practices (powerpoint presentation)

11.20-11.30 : R studio -ultra quick tour- (in Rstudio)

11.30-12.00 : Final Exercise. Can you still interpret the code of the introduction? Comment and format the code (in Rstudio)

Universiteit Utrecht

Raw Data handling

Part of: Workshop Data handling in R
Tessa Pronk, Data- and Informationspecialist, UB Utrecht
Wietze Pasma, Datamanager and PhD, UMCU

Data Science Day & RDM Support

Universiteit Utrecht

Which files to keep?

DATA1: Keep the original raw data -as is-

Keep SCRIPT1

DATA2: Keep the cleaned data

Keep SCRIPT2

DATA3: Keep the analyzed data (result)

Steps from raw data to cleaned – ready for analyses - data

Step 1. Import and check raw datafile

Step 2. Integrate with other data

Step 3. Change the data types to appropriate ones

Step 4. Make values consistent and within correct range

Step 5. Organize data to your preference

Step 6. Make sub selections

Step 7. Plot the data for inspection

Step 8. Export the cleaned data as a clean datafile

Environment:

We will work in R…

Which is embedded in a Jupyter notebook…

Which runs on a Surf facility

Dataset:

We will work with a small fictitious patient dataset, with some general measurements (heart rate, blood pressure) and some factors per patient (i.e. hospital code, date).

Universiteit Utrecht

Workshop Data Handling in R

The files (notebook, data1, data2, .r script) can be found in the SURF environment.

Part 1:
Go to the SURF environment at https://uu-its.jove.surfsara.nl
Open Jupyter notebook
Kernel: Restart and clear output
Anc
Part 2:
Install R and Rstudio if neccesary
Open Rstudio
Open file 'DataHandling.R' in Rstudio on your computer

Universiteit Utrecht

Best practices for code formatting and commenting

Part of: Workshop Data handling in R
Tessa Pronk, Data- and Informationspecialist, UB Utrecht
Wietze Pasma, Datamanager and PhD, UMCU

Data Science Day & RDM Support

Contents of this tutorial:

- ✓ Naming conventions of your files
- ✓ The order of things in your script
- ✓ Layout of your script
- ✓ Commenting your code: best practices

Universiteit Utrecht

File naming conventions

Use fixed elements in your filename:
Ie. description content, project number, name researcher/team.

File name **don't** →

- Use special characters (&%$#) or points or whitespace.

File name **do** →

- Note in a separate document what element codes in your filename mean or describe each of the files' contents.
- Keep names short and relevant, about 25 characters.
- Go from generic to specific (handy with sorting and finding)
- Use '_' or 'capitalLetters' in each combined filename.

TIP: In most operating systems 'Batch renaming software' exist.

Universiteit Utrecht

Example list of consistent filenames in a project:

prEd = Project Education
stud = students
teach = teachers
data = result
raw= Raw data
clean = cleaned data
script = script
comp = comparison

subject

content

project

prEd_stud_data_raw.xls
prEd_stud_data_clean.xls
prEd_stud_script_cleaning.r
prEd_stud_script_analyses.r
prEd_stud_report.docx

prEd_teach_data_raw.xls
prEd_teach_data_clean.xls
prEd_teach_script_cleaning.r
prEd_teach_script_analyses.r
prEd_teach_report.docx

prEd_compStudTeach_script_analyses.r
prEd_compStudTech_report.docx

The order of code in your script

1. Header with information on the script i.e.
    - Date
    - Context
    - Author
    - Purpose
    - What is needed to use the script
2. Set working directory
3. Libraries used
4. Source another script that you use
5. Functions in the current script

6. THE REST OF THE SCRIPT.

```
1  ##############################################################################
2  ### August 2016 version 1 CC-by
3  ### Script belonging to the workshop 'DataHandlinginR'.
4  ### Tessa Pronk, Utrecht University Library
5  ###
6  ### Specifically, this script provides an example to the order of things in a script.
7  ### This script needs the datafile "handle_measure.txt" and the packages 'lubridate' and 'tm'.
8  ##############################################################################
9
10 setwd("C:/Users/Desktop/Datahandling")
11
12 library(lubridate)
13 library(tm)
14 source(mysource)
15
16 myfunction <- function(argument1, argument2, ... ){
17   statement1
18   statement2
19   ...
20   return(object)
21 }
22
23 # Here starts the executive part of the script:
24
25 dist<-read.delim("handle_measure.txt",header=FALSE)      # read the data file
26 colnames(dist)<-dist[2,]                                  # set column names
27 max_cat<-30
28 dist30<-as.numeric(dist[1,1:max_cat])                     # select part to plot
29
30 windows(9,4)
31 barplot(as.numeric(dist30),col="red", xlab="publications per year", ylab='number of researchers',
32         cex.names=0.8,cex.axis=0.8)
33
34 ################## end script ##############################
```

Formatting practices

Do not use very long lines

- Split to more lines if possible.
- Maximum of 80 or 100 characters

```
# Good
x1 <- barplot(
  Pdata_plot$HR,
  col = cols,
  ylim = c(0, 200),
  xlab = "Patient number",
  ylab = "Heart rate",
  names.arg = Pdata_plot$PATNO
)

# Bad
x1 <- barplot(Pdata_plot$HR, col=cols, ylim=c(0,200), xlab="Patient number", ylab="H
```

Formatting practices

Use empty lines to separate related commands

```
16  # Good
17  dist <- read.delim("handle_dist.csv", header = FALSE)
18  colnames(dist) <- dist[2, ]
19
20  max_cat <- 30
21  dist30 <- as.numeric(dist[1:1 + max_cat])
22
23  windows(4, 9)
24  barplot(as.numeric(dist))
25
26
27  # Bad
28  dist <- read.delim("handle_dist.csv", header = FALSE)
29  colnames(dist) <- dist[2, ]
30  max_cat <- 30
31  dist30 <- as.numeric(dist[1:1 + max_cat])
32  windows(4, 9)
33  barplot(as.numeric(dist))
34
```

Formatting practices

Use indentation within a multiline statement

- First bracket '{' after the code, last bracket '}' on its own line
- Tab or spaces in between

```
38  # Good
39  if (x <= 1){
40    x <- NA
41  }
42
43
44  # Bad
45  if (x <= 1){
46  x <- NA
47  }
48
```

Formatting practices

Use spaces between operators (=, +, <-, etc.) and brackets.

```
50  # Good
51  if(class(cars) == 'data.frame') {
52    make.plot <- TRUE
53    plot(cars, ylim = c(0, 200))
54  }
55
56
57  # Bad
58  if(class(cars)=='data.frame'){
59    make.plot<-TRUE
60    plot(cars,ylim=c(0,200))
61  }
62
```

Formatting practices

Avoid putting two commands on one line using ';'

```
64  # Good
65  colnames(dist) <- dist[2,]
66  max_cat <- 30
67  dist30 <- as.numeric(dist[1:1 + max_cat])
68
69
70  # Bad
71  colnames(dist) <- dist[2,]; max_cat <- 30; dist30 <- as.numeric(dist[1:1 +
72
```

Commenting your code: best practices

Commenting your code is essential, for you as an individual as well as for your collaborators/ supervisor/ future users.

Probably you'll never, ever, in the future, regret having commented your code well!

Universiteit Utrecht

Commenting your code: best practices

- humans should be able to read your code
- Use a descriptive block as a header
- Write comments as you code. Revise afterwards.
- Put long comments – before-, short ones - after- a command
- Align code after commands as much as possible
- Keep comments short
- Do not comment the obvious (data<-read.delim("data.txt") # read the delimited data
- Also consider: change to understandable code rather than comment?
    - Use meaningful consistent identifiers and constants names
    - Do not use magic numbers in your code. For instance not: Part <- 3 / 13

  But: Part <- SumCakes / TotalPersons
- Do not comment every single line, comment the function of a few related lines.

Final assignment:

After a short introduction to Rstudio, do the following:

Have a look at the (unformatted, uncommented) code from the jupyter notebook in Rstudio.

Format and comment the code according to the best practices in the next 30 minutes!