

Workshop – Introduction into R programming for health researchers

R Basics

Yannick Rothacher
Data Scientist

Welcome to the workshop!

Lecturers:



Andreas Limacher

Verantwortlicher Methodenberatung
Functioning Information Reference Lab



Yannick Rothacher

Data Scientist
Functioning Information Reference Lab

Materials can be found at:

<https://github.com/Swiss-Paraplegic-Research/Workshop/>

Aims of the workshop

Workshop goals:

- Become familiar with the programming language **R**
 - Use **R** (and **RStudio**) to ...
 - manipulate
 - visualize
 - analyse
- ... your data.



Workshop schedule

Day 1 (Tuesday 11.3)

| Topic | Time |
|--|---------------|
| R Basics Lecture & Exercise | 09:00 – 10:30 |
| R Basics II (Functions) Lecture & Exercise | 10:30 – 12:00 |
| Lunch break | |
| Data Manipulation Lecture & Exercise | 13:30 – 15:00 |

Day 2 (Tuesday 18.3)

| Topic | Time |
|--|---------------|
| Data Visualisation Lecture & Exercise | 09:00 – 10:30 |
| Tidyverse Data Manipulation Lecture & Exercise | 10:30 – 12:00 |
| Lunch break | |
| Statistical Analysis Lecture & Exercise | 13:30 – 15:00 |

R questionnaire

To get an impression of your experience with R, think about the following statements:

- “I have never used R before.”
- “I have used R occasionally but have rather little experience.”
- “I have used R repeatedly and know the basics.”
- “I am proficient in R but would like to get a formal introduction.”

R and RStudio

R is:

- A programming language for statistical analysis
- Free to use, open-source
- Very flexible (> 10'000 add-on packages)
- Widely used among statisticians



RStudio is:

- An “integrated development environment” (IDE) for R
- Supposed to make working with R more comfortable and efficient



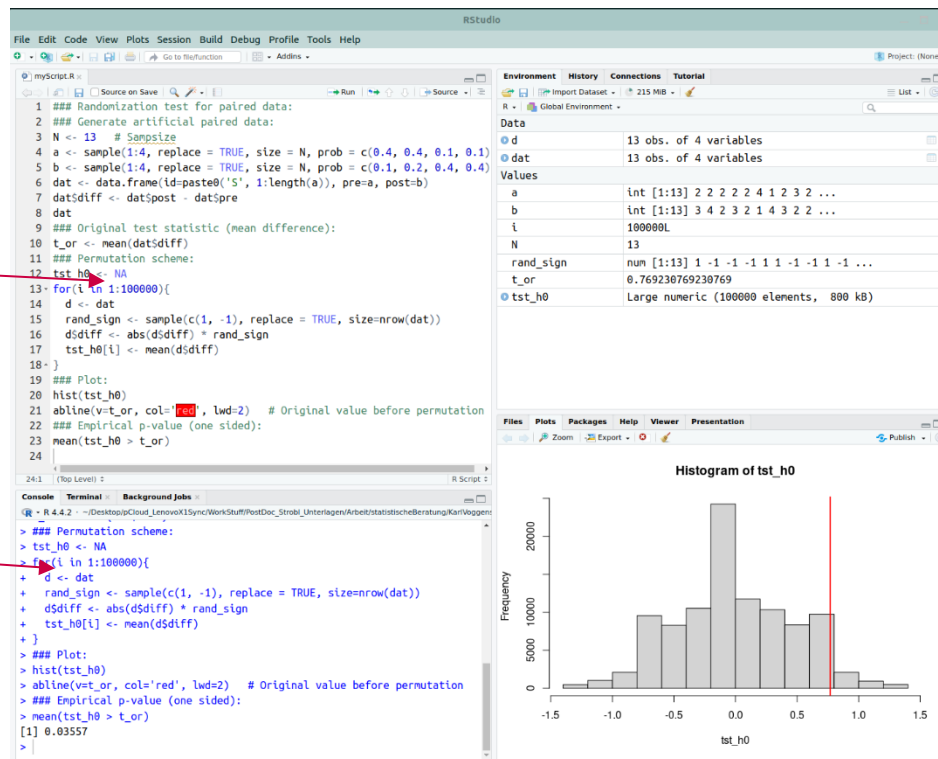
First look at RStudio

- Classically we work in Rstudio in an **R-script**

- Create new script under
“File > New File > R Script”

- Execute the code from the script in the **R-console**

- “Source” to execute the whole script
- “Run” to execute current line of code (or selection of code)



First look at R code

- Using R as a calculator:

```
> 1 + 1
```

```
[1] 2
```

```
> 10 * 30
```

```
[1] 300
```

```
> 10 / 4
```

```
[1] 2.5
```

```
> 2^3
```

```
[1] 8
```

- ...and to create objects:

```
> a <- 3
```

```
> b <- 24
```

```
> a + b
```

```
[1] 27
```

```
> longName <- 0.4
```

```
> b + longName
```

```
[1] 24.4
```

- We can use the **#** symbol to add comments (ignored when executing code)

Vectors (numerical and character)

- A vector in R is a combination of multiple elements and is created with **c()**:

```
> myvec <- c(20, 122, 39)
```

```
> myvec
```

```
[1] 20 122 39
```

- A vector containing the integers from x to y can be created with **x:y** :

```
> 1:10
```

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

- Not only numerical values are possible, here an example of a character-vector:

```
> texvec <- c("Random", "two words", "A small sentence")
```

```
> texvec
```

```
[1] "Random" "two words" "A small sentence"
```

(In a vector all elements must be of the same type)

Vectors (numerical and character)

- We can do calculations with numeric vectors:

```
> vec1 <- 1:3
```

```
> vec1
```

```
[1] 1 2 3
```

```
> vec1 + 10
```

```
[1] 11 12 13
```

```
> vec2 <- c(0, 100, -100)
```

```
> vec2
```

```
[1] 0 100 -100
```

```
> vec1 + vec2
```

```
[1] 1 102 -97
```

```
> vec3 <- vec1 + vec2
```

```
> vec3
```

```
[1] 1 102 -97
```

Selection of elements in a vector

- When processing data one is often interested in selecting specific elements. Such selections can be performed with the square brackets:

```
> vecA <- c(2, 6, 7, 9)
```

```
> vecA
```

```
[1] 2 6 7 9
```

```
> vecA[2]
```

```
[1] 6
```

- We can select multiple elements by passing the indices **as a vector**:

```
> vecA[ c(2, 4) ]
```

```
[1] 6 9
```

- Example of a nested call:

```
> s <- c(2, 4)
```

```
> vecA[s]
```

```
[1] 6 9
```

- We can also use the square brackets to assign new values to elements:

```
> vecA[2] <- 1000
```

```
> vecA
```

```
[1] 2 1000 7 9
```

Data frames

- Data frames are commonly used to represent tabular data in R. The columns of a data frame are vectors. Data frames can be created with the **data.frame** function:

```
> vecA <- c(2, 6, 7, 9)
> vecB <- c("gut", "schlecht", "mittel", "gut")
> vecC <- c(0.4, 0.2, 0.33, 0.9)
> dat <- data.frame(vecA, vecB, vecC)
> dat
```

| | vecA | vecB | vecC |
|---|------|----------|------|
| 1 | 2 | gut | 0.40 |
| 2 | 6 | schlecht | 0.20 |
| 3 | 7 | mittel | 0.33 |
| 4 | 9 | gut | 0.90 |

Selection of elements in data frames

- In a data frame, whole columns can be selected with the \$ sign:

```
> dat
```

| | vecA | vecB | vecC |
|---|------|----------|------|
| 1 | 2 | gut | 0.40 |
| 2 | 6 | schlecht | 0.20 |
| 3 | 7 | mittel | 0.33 |
| 4 | 9 | gut | 0.90 |

```
> dat$vecC
```

```
[1] 0.40 0.20 0.33 0.90
```

```
> dat$vecC[1]
```

```
[1] 0.40
```

- We can use the \$ sign to add a new column or remove a column:

```
> dat$newCol <- 1:2
```

```
> dat
```

| | vecA | vecB | vecC | newCol |
|---|------|----------|------|--------|
| 1 | 2 | gut | 0.40 | 1 |
| 2 | 6 | schlecht | 0.20 | 2 |
| 3 | 7 | mittel | 0.33 | 1 |
| 4 | 9 | gut | 0.90 | 2 |

```
> dat$vecB <- NULL
```

```
> dat
```

| | vecA | vecC | newCol |
|---|------|------|--------|
| 1 | 2 | 0.40 | 1 |
| 2 | 6 | 0.20 | 2 |
| 3 | 7 | 0.33 | 1 |
| 4 | 9 | 0.90 | 2 |

Selection of elements in data frames

- The square brackets can also be used to directly select elements in a data frame. In that case, the wanted row and column indices are separated by a comma (**rows left, columns right**):

```
> dat
  vecA      vecB vecC
1     2          gut 0.40
2     6 schlecht 0.20
3     7   mittel 0.33
4     9          gut 0.90
```

```
> dat[2, 3]
[1] 0.20
```

```
> dat[3, 1]
[1] 7
```

- We can leave a side empty to select all rows/columns:

```
> dat[1, ]
      vecA  vecB  vecC
1         2   gut 0.40
```

```
> dat[, 1]
[1] 2 6 7 9
```

```
> dat[, "vecA"] # selection with name
[1] 2 6 7 9
```

- To select multiple rows/columns, we again have to supply them as vectors:

```
> dat[ c(1, 4), c(1, 2) ]
  vecA      vecB
1     2          gut
4     9          gut
```

Logicals

- Logicals are an important class in R and can only take the values TRUE or FALSE:

```
> Lvec <- c(TRUE, FALSE, TRUE, TRUE)
```

```
> Lvec
```

```
[1] TRUE FALSE TRUE TRUE
```

- Logicals are the result of logical operators:

```
> 5 > 3
```

```
[1] TRUE
```

```
> 6 == 5    # is equal to
```

```
[1] FALSE
```

```
> c(2, 4, 6) < 4    # applied to a vector
```

```
[1] TRUE FALSE FALSE
```

Selection of elements with logicals

- Logicals can be used to select elements:

```
> alter <- c(18, 20, 31, 34)
```

```
> alter
```

```
[1] 18 20 31 34
```

```
> alter[ c(3, 4) ]
```

```
[1] 31 34
```

```
> alter[ c(FALSE, FALSE, TRUE, TRUE) ]
```

```
[1] 31 34
```

- Using a logical operator to create logical vector:

```
> alter > 30
```

```
[1] FALSE FALSE TRUE TRUE
```

- Use directly in square brackets:

```
> alter[alter > 30]
```

```
[1] 31 34
```


Selection of elements with logicals

- Logicals can be used to select elements:

```
> alter <- c(18, 20, 31, 34)
```

```
> alter
```

```
[1] 18 20 31 34
```

```
> alter[ c(3, 4) ]
```

```
[1] 31 34
```

```
> alter[ c(FALSE, FALSE, TRUE, TRUE) ]
```

```
[1] 31 34
```

- Using a logical operator to create logical vector:

```
> alter > 30
```

```
[1] FALSE FALSE TRUE TRUE
```

- Use directly in square brackets:

```
> alter[alter > 30]
```

```
[1] 31 34
```

Same thing

Selection of elements with logicals

- Logicals can be used to select **rows** in a data frame:

```
> dat
```

| | vecA | vecB | vecC |
|---|------|----------|------|
| 1 | 2 | gut | 0.40 |
| 2 | 6 | schlecht | 0.20 |
| 3 | 7 | mittel | 0.33 |
| 4 | 9 | gut | 0.90 |

```
> dat[c(FALSE, FALSE, TRUE, TRUE), ]
```

| | vecA | vecB | vecC |
|---|------|--------|------|
| 3 | 7 | mittel | 0.33 |
| 4 | 9 | gut | 0.90 |

- Using a logical operator:

```
> dat[dat$vecA > 6, ]
```

| | vecA | vecB | vecC |
|---|------|--------|------|
| 3 | 7 | mittel | 0.33 |
| 4 | 9 | gut | 0.90 |

```
> dat[dat$vecB == "gut", ]
```

| | vecA | vecB | vecC |
|---|------|------|------|
| 1 | 2 | gut | 0.40 |
| 4 | 9 | gut | 0.90 |

Exercise: R basics

- Find the exercise at:

https://github.com/Swiss-Paraplegic-Research/Workshop/tree/main/Part1_RBasics/Exercise