Introduction to R

Day 1

Sereina Herzog

Institute for Medical Informatics, Statistics and Documentation
Medical University of Graz

29.01.2025

Course Aim

- Introduction to R using RStudio
  - How to use R and RStudio
- Project structure
  - Using R as an example
- Data visualization with R
  - Using ggplot for typical plots
- Report generation using Rmarkdown
  - Advantage of avoiding "copy & paste"
  - Reproducible reports

⇒ **Help for self-help**

Course Content - Introduction R (Day 1)

- Introduction to R using RStudio
- Project structure
- Data visualization with R

Data visualization

This is where we started to draw taller bars to please you guys.
GRUHN

Source: www.googleplussuomi.com

Purpose

- Exploring and presenting data in form of graphs
- Summarizing - data reduction (mean, variance, median etc.)
- Presenting data in form of tables and/or graphs

Summarizing data (graphs)

Visualize data in graphs

- Bar chart
- Histogram
- Box-and-whisker plot
- Time series plot
- Scatterplot
- ...

When to use what

(graphs & scales)

Idea of data cleaning

- Check data using
  - Key figures (e.g. median)
  - Graphs (e.g. histogram)
- Data quality
  - consult original source (e.g. patient health record, lab journal)
- Plausibility

R & RStudio

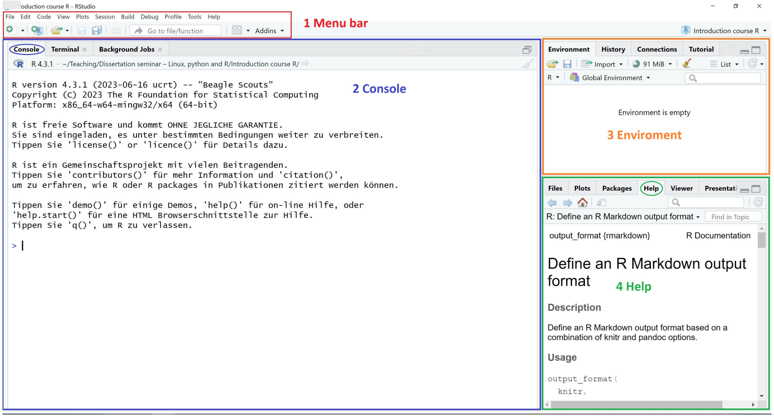
What is R and RStudio?

- R: The R Project for Statistical Computing ▸ Link R project
  - is an open-source programming languages
  - works with *R packages*

What is R and RStudio?

- R: The R Project for Statistical Computing ▸ Link R project
  - is an open-source programming languages
  - works with *R packages*
- RStudio
  - is an integrated development environment (IDE)
    - specifically designed for working with the R programming language
  - has a user-friendly interface
  - has code editing features
    - code completion feature
    - syntax-highlighting editor

RStudio - Interface



```
R version 4.3.1 (2023-06-16 ucrt) -- "Beagle Scouts"
Copyright (C) 2023 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R ist freie Software und kommt OHNE JEGLICHE GARANTIE.
Sie sind eingeladen, es unter bestimmten Bedingungen weiter zu verbreiten.
Tippen Sie 'license()' or 'licence()' für Details dazu.

R ist ein Gemeinschaftsprojekt mit vielen Beitragenden.
Tippen Sie 'contributors()' für mehr Information und 'citation()',
um zu erfahren, wie R oder R packages in Publikationen zitiert werden können.

Tippen Sie 'demo()' für einige Demos, 'help()' für on-line Hilfe, oder
'help.start()' für eine HTML Browserschnittstelle zur Hilfe.
Tippen Sie 'q()', um R zu verlassen.

> 
```

output_format {rmarkdown} R Documentation

Define an R Markdown output format

Description

Define an R Markdown output format based on a combination of knitr and pandoc options.

Usage

```
output_format(
  knitr,
```

RStudio - Getting started

- Open RStudio
- Work through ‘Day 1 - Exercise 1’ (together)

Data types and structures in R

- Data types
  - character
  - numeric (real or decimal)
  - integer
  - logical
  - complex
- Data structures
  - atomic vector (i.e. only holds data of a single data type)
  - list
  - matrix
  - data frame
  - factors
  - ...

Examine features in R

- Examine features
  - *class()* - what kind of object is it (high-level)?
  - *typeof()* - what is the object's data type (low-level)?
  - *length()* - how long is it? What about two dimensional objects?
  - *attributes()* - does it have any metadata?
  - ...

Example examing features (I)

```r
x <- "dataset"
typeof(x)
```

```
## [1] "character"
```

```r
attributes(x)
```

```
## NULL
```

Example examing features (II)

```r
y <- 1:10
y
```

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

```r
typeof(y)
```

```
## [1] "integer"
```

```r
length(y)
```

```
## [1] 10
```

Example examing features (III)

```r
z <- as.numeric(c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10))
z
```

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

```r
class(z)
```

```
## [1] "numeric"
```

```r
typeof(z)
```

```
## [1] "double"
```

Folder structure and R projects in RStudio

Folder structure

Suggestion how to structure your project folder

- project1
  - literature
  - reports
  - . . .
  - R

Folder structure

Suggestion how to structure your project folder

- project1
  - literature
  - reports
  - . . .
  - R
    - orig
    - Rdata
    - Rmarkdown
    - Routput
    - Rfiles

Folder structure

Suggestion how to structure your project folder

- project1
  - literature
  - reports
  - ...
  - R
    - orig
    - Rdata
    - Rmarkdown
    - Routput
    - Rfiles

Hint: never touch the original data!

Folder structure

Idea: set path at the beginning of your file with syntax related to your *R* folder and everything else relative to that.

```r
path <- "C:/myname/work/project1/R"
setwd(path)
```

Folder structure

Idea: set path at the beginning of your file with syntax related to your *R* folder and everything else relative to that.

```
path <- "C:/myname/work/project1/R"
setwd(path)
```

For example, data `example0.csv` is in your `Rdata` folder

```
library(readr)
dat <- read_csv(file = "Rdata/example0.csv")
```

Folder structure

Idea: set path at the beginning of your file with syntax related to your *R* folder and everything else relative to that.

```
path <- "C:/myname/work/project1/R"
setwd(path)
```

For example, data `example0.csv` is in your `Rdata` folder

```
library(readr)
dat <- read_csv(file = "Rdata/example0.csv")
```

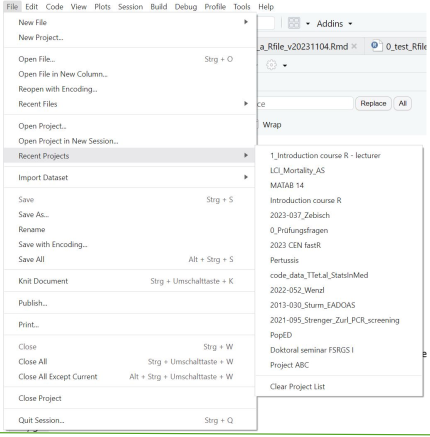
OR: use ‘R project’ option!

R project

- An R project
    - is a way to organize files and folders related to a specific analysis or project
        - easy to switch different projects
        - the working directory is the project's root folder

R project

- An R project
  - is a way to organize files and folders related to a specific analysis or project
    - easy to switch different projects
    - the working directory is the project's root folder



TO DO - Create folder structure

1) Generate following folder structure

- Course Introduction R
    - 0_slides
    - 1_exercises
    - ...
    - R
        - orig
        - Rdata
        - Rfiles
        - Rmarkdown
        - Rfiles

TO DO - Create R project

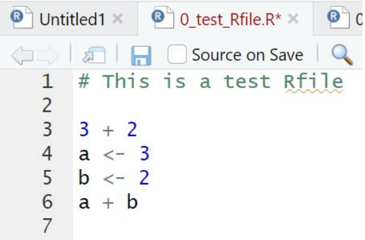
2) Generate a ‘R project’ (together)

- File → New Project... → Existing Directory

R files

R files

- An R file (*.R*) is
  - a script written in R
  - contains code that can be executed within the R software environment

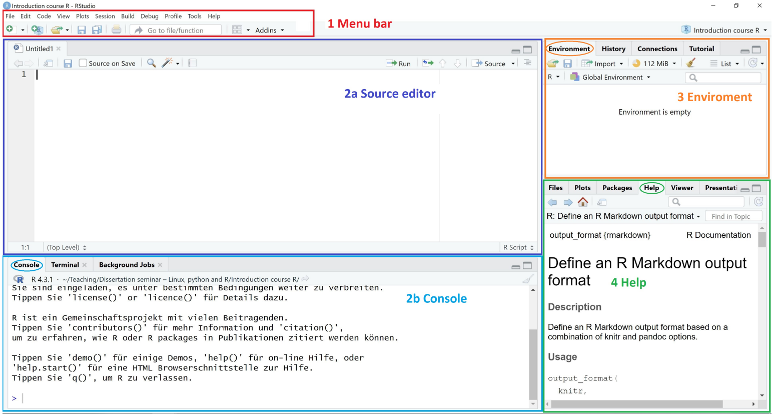


```
# This is a test Rfile

3 + 2
a <- 3
b <- 2
a + b

```

RStudio - Interface with open script



```
Sie sind eingeladen, es unter bestimmten Bedingungen weiter zu verbreiten.
Tippen Sie 'license()' or 'licence()' für Details dazu.

R ist ein Gemeinschaftsprojekt mit vielen Beitragenden.
Tippen Sie 'contributors()' für mehr Information und 'citation()',
um zu erfahren, wie R oder R packages in Publikationen zitiert werden können.

Tippen Sie 'demo()' für einige Demos, 'help()' für on-line Hilfe, oder
'help.start()' für eine HTML Browserschnittstelle zur Hilfe.
Tippen Sie 'q()', um R zu verlassen.

>
```

output_format {rmarkdown} R Documentation

Define an R Markdown output format

Description

Define an R Markdown output format based on a combination of knitr and pandoc options.

Usage

```
output_format(
  knitr,
```

R file - Getting started

- Switch to RStudio
- Work through ‘Day 1 - Exercise 2’ (together)

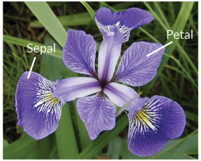
Data visualization with *ggplot*

Example - Iris

A famous iris data set gives the measurements in centimeters of the variables

- sepal length
- sepal width
- petal length
- petal width

for 50 flowers from each of 3 species of iris (*Iris setosa*, *versicolor*, and *virginica*).

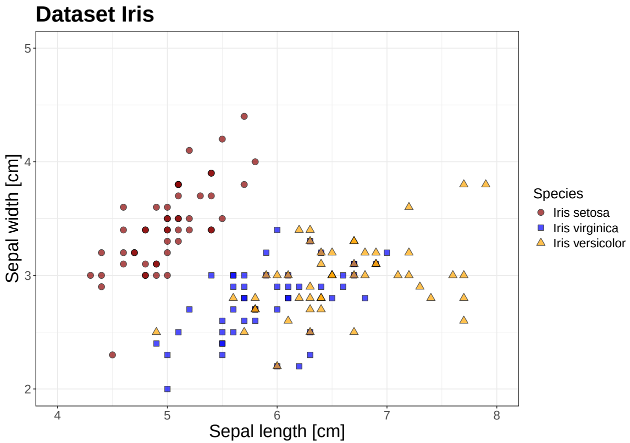


Iris Versicolor

Iris Setosa

Iris Virginica

Example - Iris



What is *ggplot*?

- powerful data visualization package in R
  - wide range of high-quality plots and graphics
  - provides a consistent syntax
  - a layered approach to building plots

What is *ggplot*?

- powerful data visualization package in R
  - wide range of high-quality plots and graphics
  - provides a consistent syntax
  - a layered approach to building plots
- consists of three main components:

What is *ggplot*?

- powerful data visualization package in R
    - wide range of high-quality plots and graphics
    - provides a consistent syntax
    - a layered approach to building plots
- consists of three main components:
    - **data**
        - represents the dataset being visualized

What is *ggplot*?

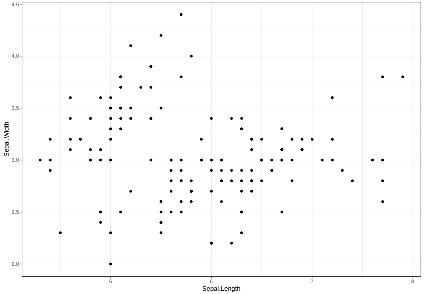
- powerful data visualization package in R
  - wide range of high-quality plots and graphics
  - provides a consistent syntax
  - a layered approach to building plots
- consists of three main components:
  - **data**
    - represents the dataset being visualized
  - **aesthetics** (aes)
    - define how variables are mapped to visual properties (e.g., x-axis, y-axis, color)

What is *ggplot*?

- powerful data visualization package in R
  - wide range of high-quality plots and graphics
  - provides a consistent syntax
  - a layered approach to building plots
- consists of three main components:
  - **data**
    - represents the dataset being visualized
  - **aesthetics** (aes)
    - define how variables are mapped to visual properties (e.g., x-axis, y-axis, color)
  - **geometric objects** (geom)
    - determine the type of plot (e.g., points, lines, bars)

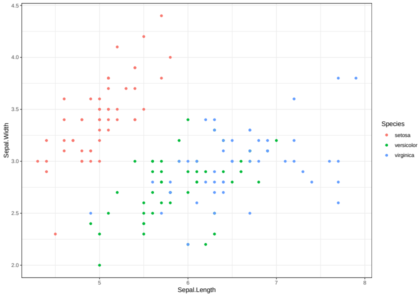
Example - Iris

```r
ggplot(data = iris,
       aes(x = Sepal.Length, y = Sepal.Width)) +
    geom_point() +
    theme_bw()
```



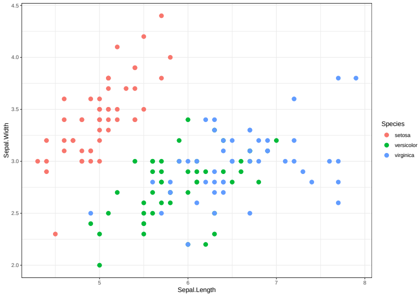
Example - Iris: including species as colour

```
ggplot(data = iris,
       aes(x = Sepal.Length, y = Sepal.Width, colour = Species)) +
    geom_point() +
    theme_bw()
```



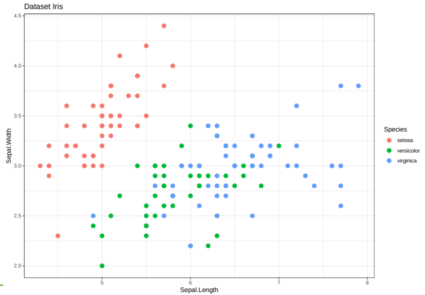
Example - Iris: increase point size

```
ggplot(data = iris,
       aes(x = Sepal.Length, y = Sepal.Width, colour = Species)) +
    geom_point(size = 3) +
    theme_bw()
```



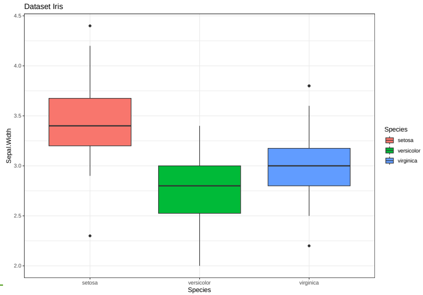
Example - Iris: adding title

```r
ggplot(data = iris,
       aes(x = Sepal.Length, y = Sepal.Width, colour = Species)) +
    geom_point(size = 3) +
    labs(title = "Dataset Iris") +
    theme_bw()
```



Example - Iris: using another geom

```
ggplot(data = iris,
       aes(x = Species, y = Sepal.Width, fill = Species)) +
    geom_boxplot() +
    labs(title = "Dataset Iris") +
    theme_bw()
```



ggplot - Getting started

- Work through ‘Day 1 - Exercise 3’
- Switch to RStudio
- Open Rmd file: *day1_ex3_ggplot_vYYYYMMDD.R*
    - is on GitHub

Saving ggplots

```
plot_iris <-
  ggplot(data = iris,
         aes(x = Sepal.Length, y = Sepal.Width, colour = Species)) +
      geom_point() +
      theme_bw()

ggsave(filename = "../Routputs/example_iris.png", plot = plot_iris,
       units = "cm", width = 12, height = 7)
```

Saving ggplots

```r
plot_iris <-
  ggplot(data = iris,
         aes(x = Sepal.Length, y = Sepal.Width, colour = Species)) +
      geom_point() +
      theme_bw()

ggsave(filename = "../Routputs/example_iris.png", plot = plot_iris,
       units = "cm", width = 12, height = 7)
```

- Try to save your last plot in the ‘Day 1 - Exercise 3’
  - test different formats and values for width/height

Links

Links (I)

- Introduction to R
  - R for Data Science (https://r4ds.hadley.nz/)
- Plots using ggplot
  - Overview with further links to course material: https://ggplot2.tidyverse.org/
- Display tables using flextable
  - flextable bool https://ardata-fr.github.io/flextable-book/
  - Function references https://davidgohel.github.io/flextable/reference/index.html
- knit_child()
  - link (https://bookdown.org/yihui/rmarkdown-cookbook/child-document.html)

Links (II)

- Download R
  - CRAN (https://cran.r-project.org/)
- Download RStudio
  - RStudio Desktop (https://posit.co/download/rstudio-desktop/)