

# Advanced R

## Unit 1

Sereina Herzog

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

05.03.2025

# Course Content - Advanced R (Unit 1)

- ▶ Short repetition
  - Reproducibility - Rmarkdown for reports
  - Project structure
  - Visualization with ggplot

# Repetition

# What is reproducibility in science?

- ▶ Ability to reproduce results by a peer
- ▶ Requires data, methods, and procedures
- ▶ Increasingly, science is supposed to be reproducible

Be nice to your future selves!

# Reproducibility with RStudio & R

- ▶ R with RMarkdown can be used to produce different types of documents [see: <http://rmarkdown.rstudio.com/gallery.html>]
  - standardised reports (html, pdf)
  - word documents (.docx)
  - slides for presentations (html, pdf, powerpoint)
  - journal articles. using the rarticles package (.pdf)
  - ...

# Reproducibility with RStudio & R

- ▶ R with RMarkdown can be used to produce different types of documents [see: <http://rmarkdown.rstudio.com/gallery.html>]
  - standardised reports (html, pdf)
  - word documents (.docx)
  - slides for presentations (html, pdf, powerpoint)
  - journal articles. using the rarticles package (.pdf)
  - ...

⇒ **making transparent and reproducible analysis**

# 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
    - ▶ Rfiles
    - ▶ Rmarkdown
    - ▶ Routput



# Folder structure

Suggestion how to structure your project folder

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

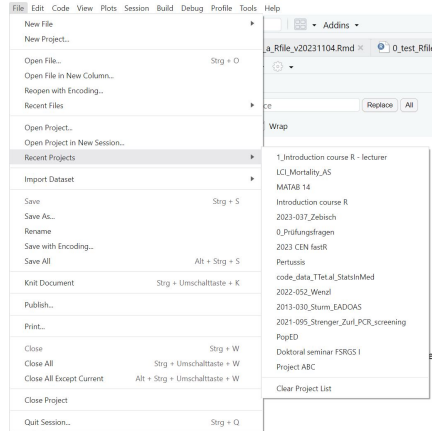
**Hint: never touch the original data!**

# 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



# Create folder structure & R project

## 1) Download prepared folder structure

- download 'projectstructure\_for\_students.zip' from GitHub
- unzip the file
- put folder 'Course Advanced R' wherever you want to have it

## 2) Generate a 'R project' (together)

- File → New Project... → Existing Directory

# 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*?

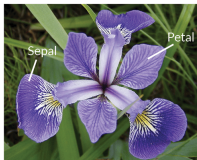
- ▶ powerful data visualization package in R
  - wide range of high-quality plots and graphics
  - provides a consistent syntax
  - a layered approach to building plots
- ▶ consists of three main components:
  - **data**
    - ▶ represents the dataset being visualized
  - **aesthetics** (aes)
    - ▶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

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**



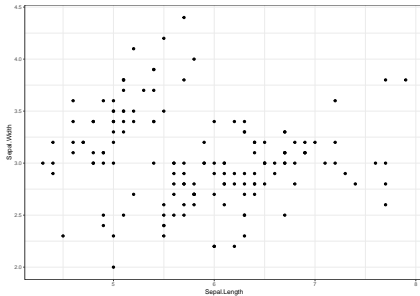
**Iris Setosa**



**Iris Virginica**

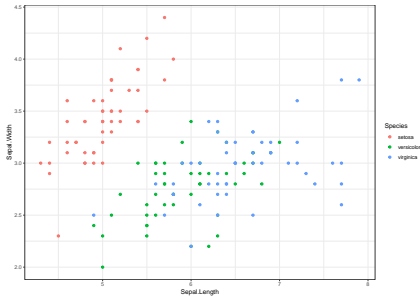
## Example - Iris

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



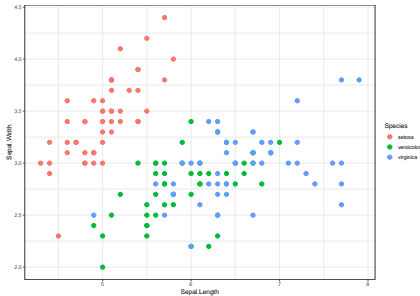
## Example - Iris: including species as colour

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



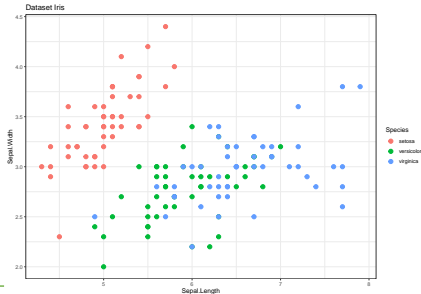
## Example - Iris: increase point size

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



## Example - Iris: adding title

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

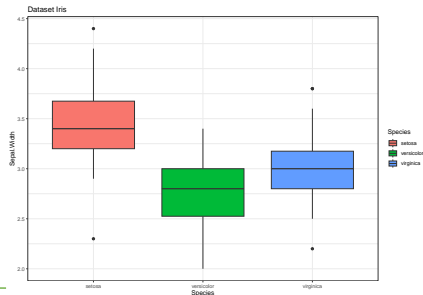


# Example - Iris



## Example - Iris: using another geom

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





## 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)
```

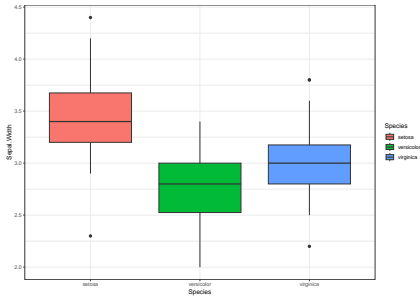
# Exercise repetition

- ▶ Work through 'Unit 1 - Exercise 1'

# Placeholders

## Example - Iris

```
ggplot(data = iris,  
       aes(x = Species, y = Sepal.Width, fill = Species)) +  
  geom_boxplot() +  
  theme_bw()
```



# Working with variables as placeholders

```
ggplot(data = iris,  
       aes(x = Species, y = Sepal.Width, fill = Species)) +  
  geom_boxplot() +  
  theme_bw()
```

# Working with variables as placeholders

```
ggplot(data = iris,  
       aes(x = Species, y = Sepal.Width, fill = Species)) +  
  geom_boxplot() +  
  theme_bw()
```

```
var_int <- "Sepal.Width"  
group_int <- "Species"
```

# Working with variables as placeholders

```
ggplot(data = iris,  
       aes(x = Species, y = Sepal.Width, fill = Species)) +  
  geom_boxplot() +  
  theme_bw()
```

```
var_int <- "Sepal.Width"  
group_int <- "Species"
```

```
ggplot(data = iris,  
       aes(x = get(group_int), y = get(var_int), fill = get(group_int))) +  
  geom_boxplot() +  
  theme_bw()
```

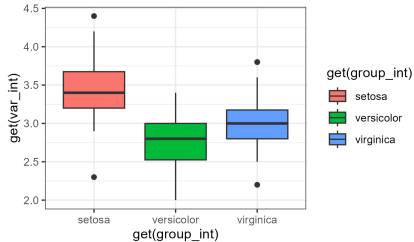
# Working with variables as placeholders

```
ggplot(data = iris,  
       aes(x = get(group_int), y = get(var_int), fill = get(group_int))) +  
  geom_boxplot() +  
  theme_bw()
```



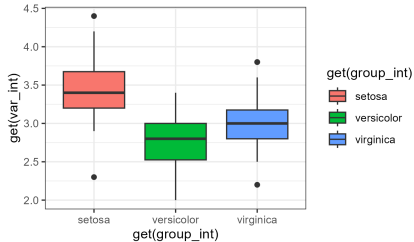
# Working with variables as placeholders

```
ggplot(data = iris,  
       aes(x = get(group_int), y = get(var_int), fill = get(group_int))) +  
  geom_boxplot() +  
  theme_bw()
```



# Working with variables as placeholders

```
ggplot(data = iris,  
       aes(x = get(group_int), y = get(var_int), fill = get(group_int))) +  
  geom_boxplot() +  
  theme_bw()
```



Problem: axis labels and legend title → need to adapt them too

# Working with variables as placeholders

```
var_int <- "Sepal.Width"  
var_int_lab <- "Sepal width [cm]"  
  
group_int <- "Species"  
group_int_lab <- "Species Iris"
```

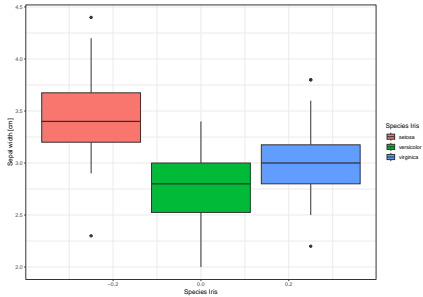
# Working with variables as placeholders

```
var_int <- "Sepal.Width"  
var_int_lab <- "Sepal width [cm]"
```

```
group_int <- "Species"  
group_int_lab <- "Species Iris"
```

```
ggplot(data = iris,  
       aes(x = get(group_int), y = get(var_int), fill = get(group_int))) +  
  geom_boxplot() +  
  
  guides(fill = guide_legend(group_int_lab)) +  
  
  xlab(group_int_lab) +  
  ylab(var_int_lab) +  
  
  theme_bw()
```

# Working with variables as placeholders



# Working with variables as placeholders

Advantage - can reuse same code for plots and only need to change things at one place

# Working with variables as placeholders

Advantage - can reuse same code for plots and only need to change things at one place

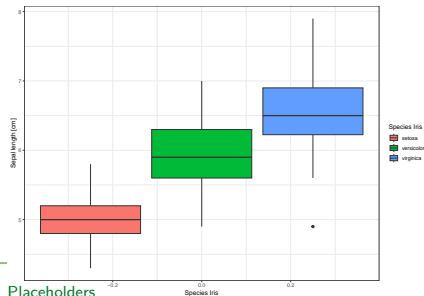
```
var_int <- "Sepal.Length"  
var_int_lab <- "Sepal length [cm]"  
  
group_int <- "Species"  
group_int_lab <- "Species Iris"
```

# Working with variables as placeholders

Advantage - can reuse same code for plots and only need to change things at one place

```
var_int <- "Sepal.Length"
var_int_lab <- "Sepal length [cm]"

group_int <- "Species"
group_int_lab <- "Species Iris"
```





# Exercise placeholders

- ▶ Work through 'Unit 1 - Exercise 2'

# 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/>)