Statistical Methods And Data Analysis In Developmental Psychology

Course Preparation

Filippo Gambarota

Updated on 2023-04-25

Lectures

The lectures are summarized in the following table:

Day	Date	Time	Room
Wednesday	26/04/2023	10:30-12:30	3L
Thursday	27/04/2023	10:30-12:30	3L
Tuesday	02/05/2023	10:30-12:30	3L
Wednesday	03/05/2023	10:30-12:30	3L
Thursday	04/05/2023	10:30-12:30	3L
Tuesday	09/05/2023	10:30-12:30	3L
Wednesday	10/05/2023	10:30-12:30	3L
Thursday	11/05/2023	10:30-12:30	3L
Tuesday	16/05/2023	10:30-12:30	3L
Wednesday	17/05/2023	10:30-12:30	3L
Thursday	18/05/2023	10:30-12:30	3L

Contacts

- **Email**: filippo.gambarota@unipd.it
- **Office**: 027, 1° Floor, Psico 1
- **Office hours**: Schedule an appointment writing an email

R

The course materials are created using R (4.3.0). To organize the materials I used **R Projects**, a feature available with R Studio. I highly suggest you to use R Projects that significantly improves your workflow. A brief tutorial is available here https://r4ds.had.co.nz/workflow-projects.html.

R Materials

All the slides and extra materials are available on Moodle. During the course and for the exercises we will use some **custom functions** that I wrote for the course. To download and use the function you can download the `utils-glm.R` file from Moodle or from here **https://stat-teaching.github.io/SMDA-2023/R/utils-glm.R**

R Packages

We will use, directly or indirectly (for custom functions), several R packages. You can easily install all packages using:

```r
pksg <- c("effects", "tidyverse", "MASS",
          "broom", "reshape2", "ggeffects",
          "performance", "see", "car", "devtools")
install.packages(pkgs)
```

Pipes

Sometimes in my code you will see a symbol like this `|>`, this is called **pipe**. Sometimes it is also written as `%>%` that is a different pipe coming from the `magrittr` package. The pipe is a very simple way to write R code when you need to apply multiple functions in succession. Practically, the pipe apply a function to an element, for example:

```r
# these two are the same
mean(x)

# to x apply the mean function
x |> mean()
```

Beyond this silly example, when we need to use multiple nested functions the pipe makes the code more readable:

```r
x <- runif(10)

# without pipe
exp(min(round(x, 2)))
```

```
## [1] 1.377128
```

```r
# with pipe
x |>
    round(2) |>
    min() |>
    exp()
```

```
## [1] 1.377128
```

Essentially, with the pipe we concatenate multiple functions where implicitly the first argument of the function is assigned to the object before the pipe.