

# **Intro to This Course; Getting Started with R**

**PSICOSTAT**  
Enrico Toffalini

# Why becoming an R user?

- **Mainstream in academia** for statistical computing and data science, increasingly used in business. *Job market advantage!*
- **Free & open-source**: wherever you go, R will be with you at no costs (unlike *MATLAB, MPLUS, SPSS*, etc.)
- **Real programming language**: difficult at the beginning, but: 1) gives you lots of flexibility; 2) has transfer on other programming languages (e.g., *Python*).
- **Vast community support** thanks to a large and active community (plus *chatGPT*, *Gemini*, *Lucrez-IA*, etc., know it pretty well!).
- **Huge ecosystem**, >23,000 packages on CRAN, more from other sources (e.g., GitHub), to do amazing stuff with statistical data analysis, machine learning, data visualization, developing webapps [*shiny*], writing reports and even entire books [*bookdown*, *rmarkdown*]); also, can integrate with *Quarto*, *GitHub*.
- **Facilitates reproducible scientific research** by sharing code and workflows.

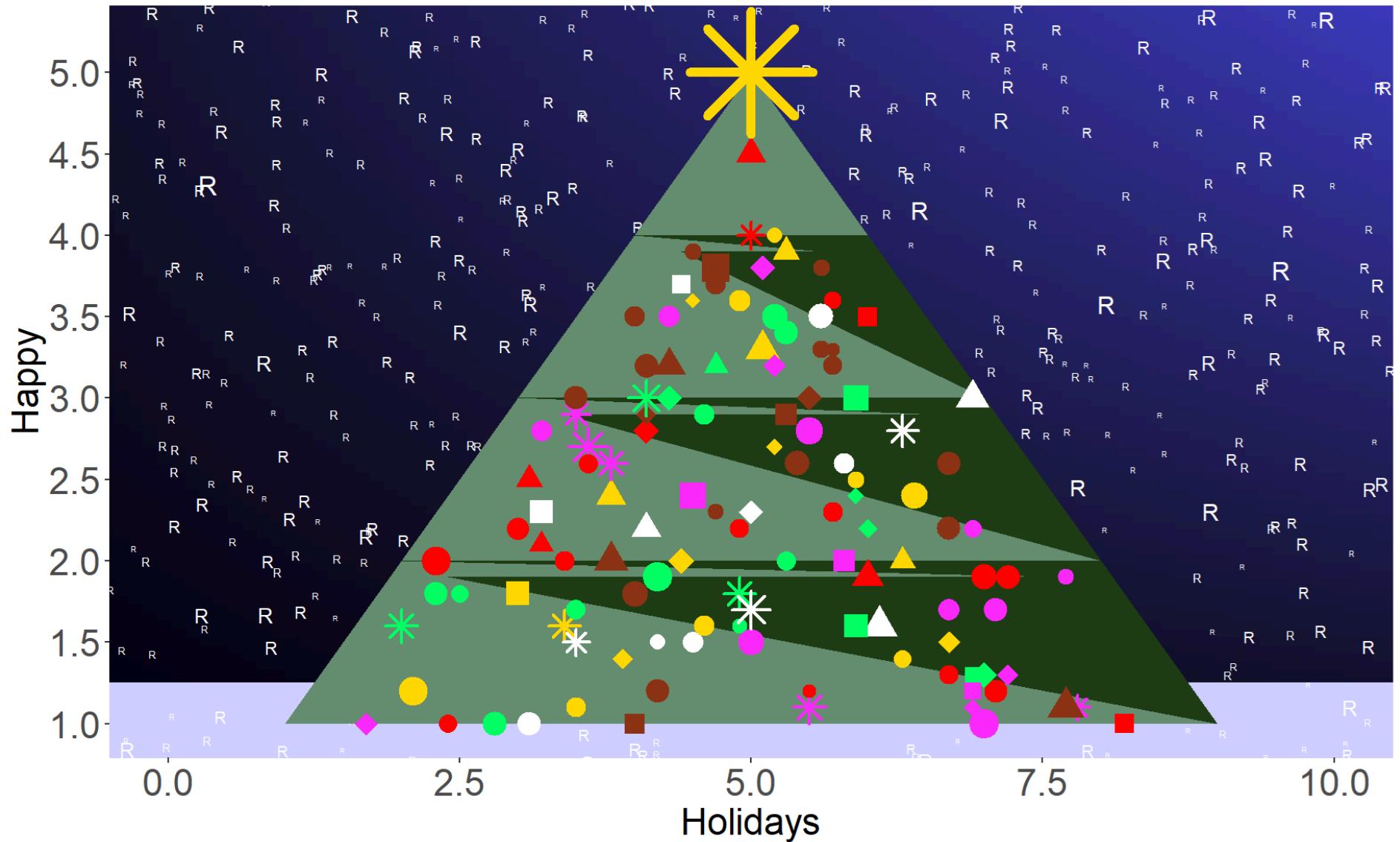
# What you may expect to learn in this course:

- Executing fundamental operations and using basic functions;
- Working with essential data types and structures;
- Gaining some proficiency in managing and manipulating data with vectors and dataframes;
- Understanding some fundamental concepts of programming.

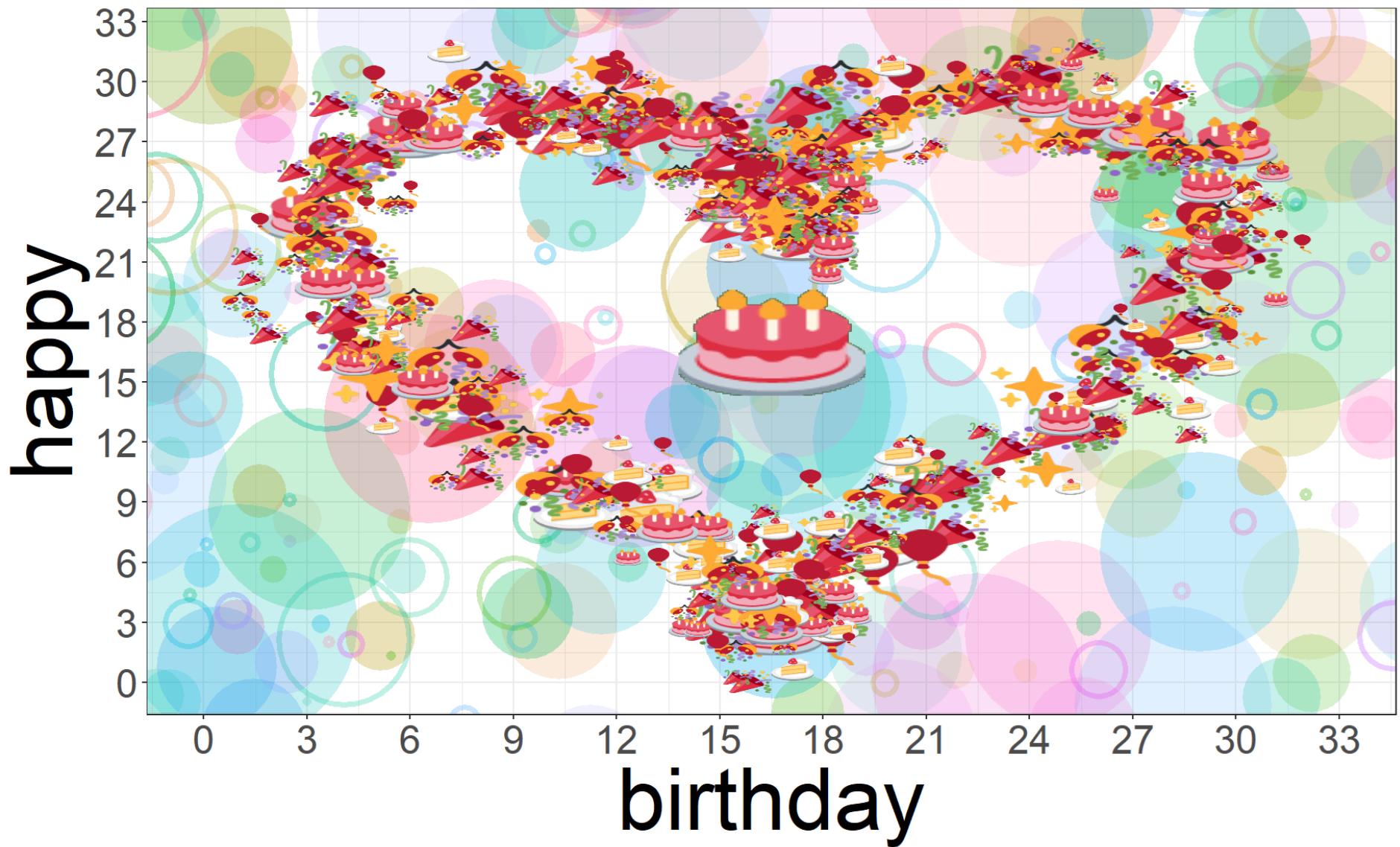
Over the next two years, following this PhD program, you will or may have the opportunity to use R to perform at least some fundamentals about:

- Core statistical inference methods;
- LM/LMM/GLMM: (Generalized) linear (mixed-effects) models;
- Data visualization using *ggplot2*;
- Power analysis & more via data simulation;
- SEM: Structural Equation Modeling;
- Conducting meta-analysis.

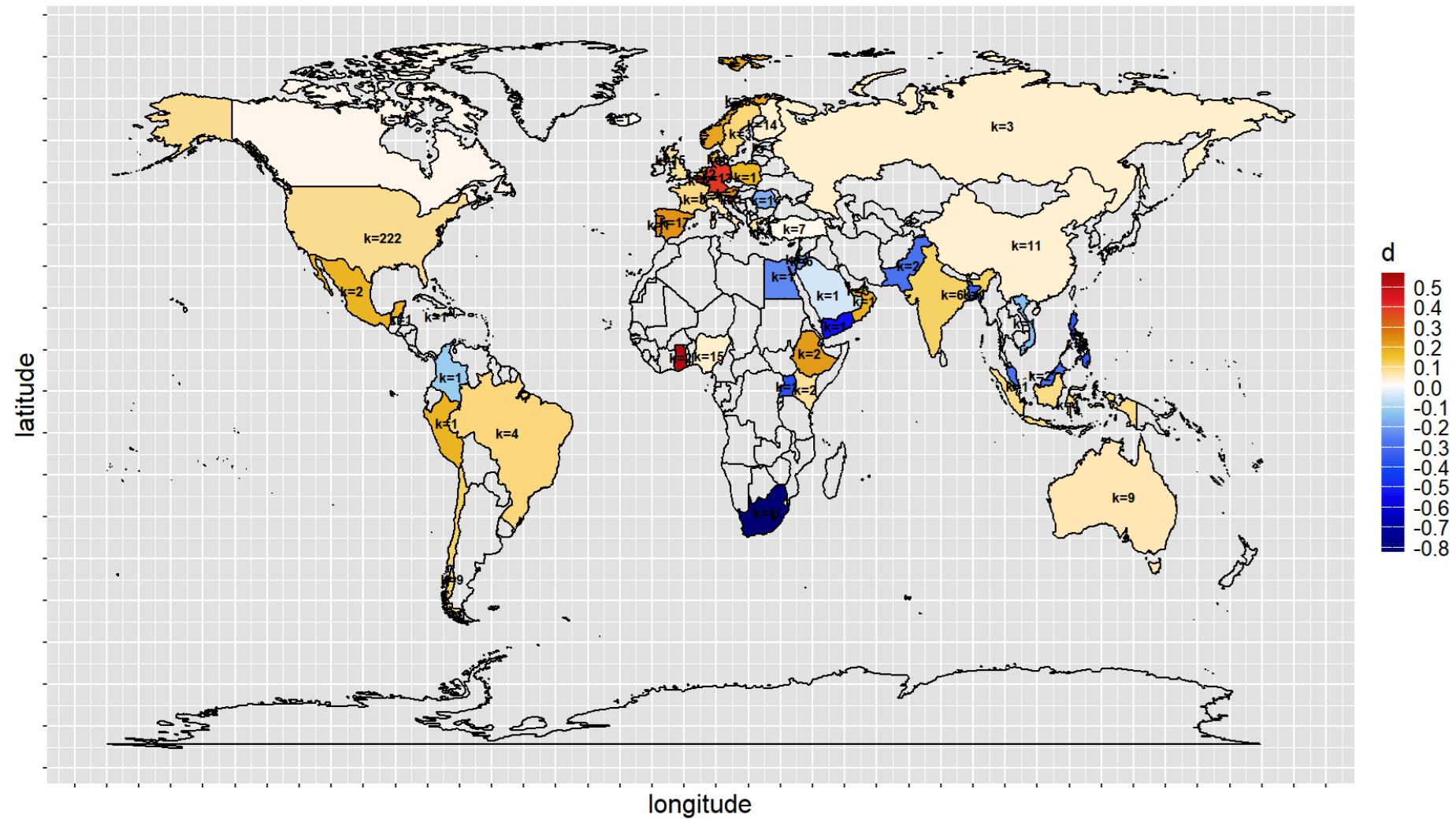
# you may even create greeting cards



# you may even create greeting cards

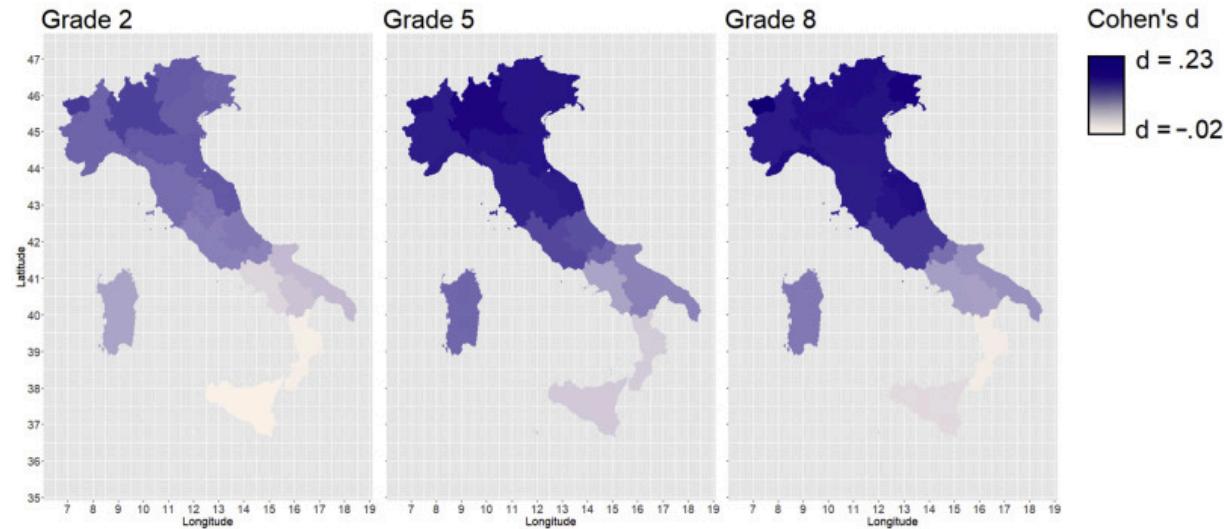


# or like fancy infographics

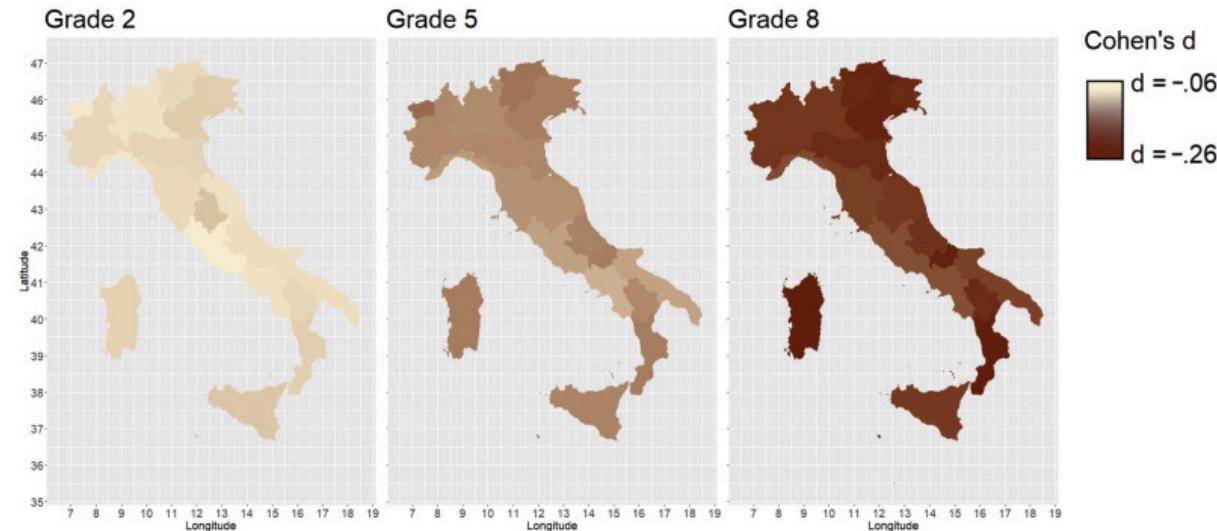


# or like fancy infographics

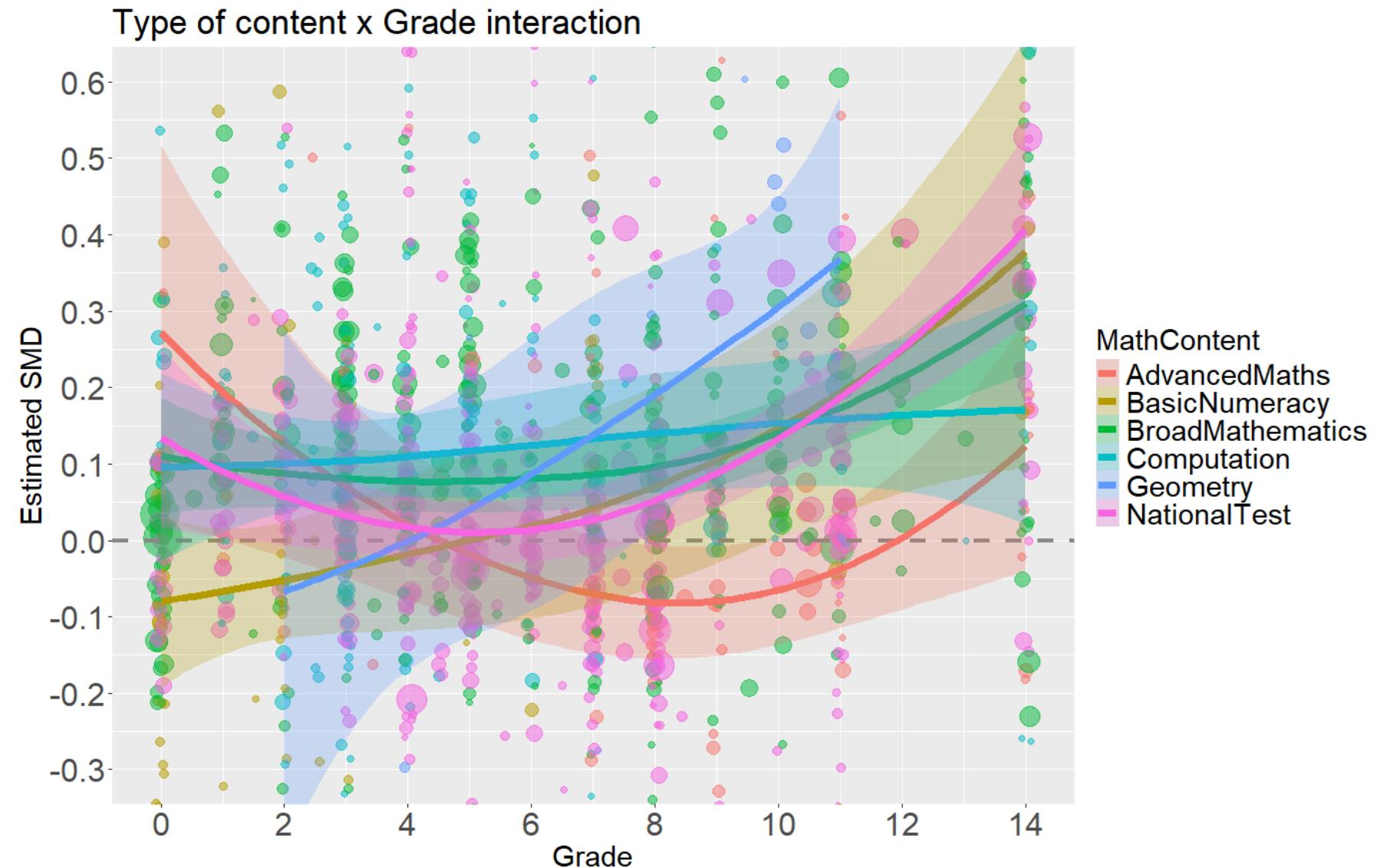
A.



B.



# perform fancy moderated meta-analyses



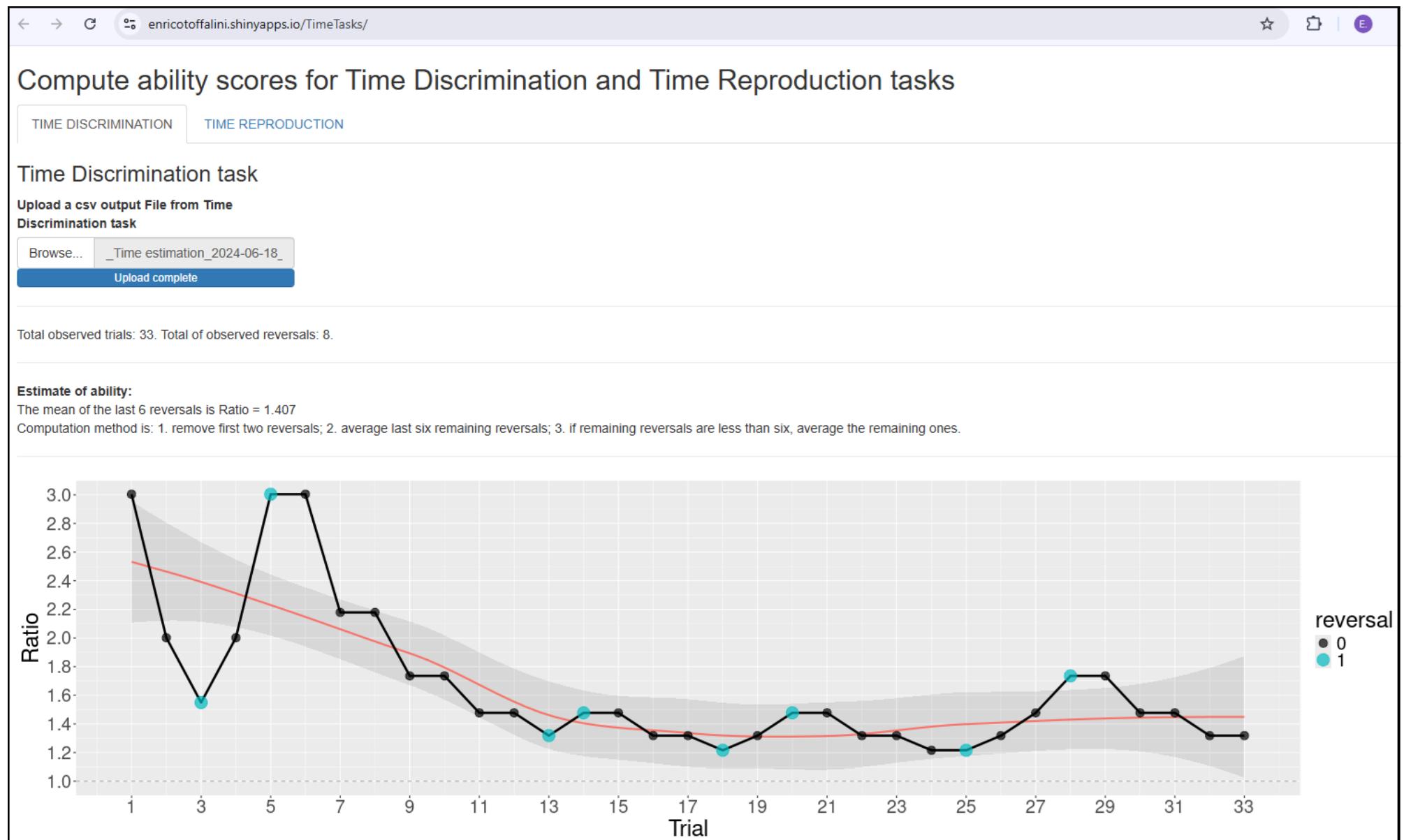
# you may create interactive webapps with Shiny

see [Shiny gallery](#)

here's a couple of recent real examples from Psicostat members:

- this [game-like shiny app](#) developed for the science4all event in Padova; see [here](#) some explanation in Italian
- practical [ad-hoc shiny app](#) for scoring experimental data collected by students

# you may create interactive webapps with Shiny



# or entire websites and books

examples of other resources that can be created within the R ecosystem, integrating other tools such as *Github* and *Quarto*:

- this very **course support material** is a website in its own right
- this very **course textbook** is a book/website
- **this book** by Daniël Lakens explaining Statistical Inference

# R + Integrated Development Environment(s)

Make sure you install:

- R as the programming language interpreter and the basic environment and packages
- RStudio is the IDE of election to make writing R code easy

Interesting alternatives to installing RStudio:

- Positron (based on MS Visual Studio)
- Posit.cloud (fully online, actually RStudio)
- Google Colab (fully online, make sure to set *R runtime type*; actually a Jupyter notebook)

# R Console (just base R)

The screenshot shows the RGui (64-bit) interface. On the left is a window titled "C:\Users\toffalini\Desktop\RstudioShow.R - Editor di R" containing R code. On the right is a window titled "R Console" showing the execution of the same code. The word "SCRIPT" is overlaid in red on the editor window, and the word "CONSOLE" is overlaid in red on the console window.

```
# Simulation example
niter = 10000
results = data.frame(iter=1:niter,
                      pvalue=NA)
N = 50
b0 = 0
b1 = 0.5
sigma = 1
for(i in 1:niter){
  x = rnorm(N,0,1)
  y = b0 + x*b1 + rnorm(N,0,sigma)
  df = data.frame(x,y)
  fit = lm(y~x, data=df)
  results$pvalue[i] =
    summary(fit)$coefficients["x","Pr(>|t|)"]
}
(power = mean(results$pvalue<0.05))

> results = data.frame(iter=1:niter,
+                         pvalue=NA)
> N = 50
> b0 = 0
> b1 = 0.5
> sigma = 1
> for(i in 1:niter){
+   x = rnorm(N,0,1)
+   y = b0 + x*b1 + rnorm(N,0,sigma)
+   df = data.frame(x,y)
+   fit = lm(y~x, data=df)
+   results$pvalue[i] =
+     summary(fit)$coefficients["x","Pr(>|t|)"]
+ }
> (power = mean(results$pvalue<0.05))
[1] 0.9155
```

# R Studio (full IDE)

The screenshot displays the R Studio interface with four main sections labeled in red:

- SCRIPT**: The top-left pane shows an R script named "index.Rmd" containing code for a simulation example. The code includes a loop to fit linear models and calculate p-values.
- CONSOLE**: The top-right pane shows the R console output, which is identical to the script, indicating the execution of the R code.
- ENVIRONMENT**: The bottom-left pane shows the R environment browser, listing objects like df, fit, results, b0, b1, i, N, niter, power, etc., with their corresponding values.
- FILE EXPLORER, ETC.**: The bottom-right pane shows the file explorer, displaying a folder structure and files related to the project.

**Script Content (index.Rmd):**

```
1 # Simulation example
2 niter = 10000
3 results = data.frame(iiter=1:niter,
4                         pvalue=NA)
5 N = 50
6 b0 = 0
7 b1 = 0.5
8 sigma = 1
9 for(i in 1:niter){
10   x = rnorm(N,0,1)
11   y = b0 + x*b1 + rnorm(N,0,sigma)
12   df = data.frame(x,y)
13   fit = lm(y~x, data=df)
14   results$pvalue[i] = summary(fit)$coefficients["x","Pr(>|t|)"]
15 }
16 (power = mean(results$pvalue<0.05))
17 (power = mean(results$pvalue<0.05))
```

**Console Output:**

```
+ }
> (power = mean(results$pvalue<0.05))
[1] 0.918
> # Simulation example
> niter = 10000
> results = data.frame(iiter=1:niter,
+                         pvalue=NA)
> N = 50
> b0 = 0
> b1 = 0.5
> sigma = 1
> for(i in 1:niter){
+   x = rnorm(N,0,1)
+   y = b0 + x*b1 + rnorm(N,0,sigma)
+   df = data.frame(x,y)
+   fit = lm(y~x, data=df)
+   results$pvalue[i] = summary(fit)$coefficients["x","Pr(>|t|)"]
+ }
> (power = mean(results$pvalue<0.05))
[1] 0.9105
>
```

**Environment View:**

Data	Value
df	50 obs. of 2 variables
fit	List of 12
results	10000 obs. of 2 variables
Values	
b0	0
b1	0.5
i	10000L
N	50
niter	10000
power	0.9105

**File Explorer View:**

Name	Size	Modified
faviconspsicostat2.png	79.8 KB	Oct 31, 2024, 10:25 AM
psicostatLogo.png	264 KB	Mar 4, 2024, 2:05 PM
RstudioShow.R	378 B	Oct 31, 2024, 5:08 PM

# Google Colab (online notebook)

The screenshot shows the Google Colab interface with several red annotations:

- A red arrow points from the "Share" button in the top right to the text "shareable via link like any Google document".
- A red arrow points from the "add stuff" button in the toolbar to the "execute" button.
- A red arrow points from the heading "Simple Power Simulation Example - R" to the word "HEADING".
- A red arrow points from the text "First of all, define parameters and pre-allocate dataframe for results." to the word "paragraph".
- A red arrow points from the R code block to the text "R chunk".
- A red arrow points from the text "Now run the simulation loop, and store result (p-value) at each iteration." to the word "paragraph".
- A red arrow points from the R code block to the text "R chunk".
- A red callout bubble with the text "remember to set R" has a red arrow pointing to the R code block.
- A red arrow points from the text "Finally, compute and inspect empirical power" to the word "paragraph".
- A red arrow points from the R code block to the text "R chunk with final result".
- A red arrow points from the "RAM" and "Disk" status indicators in the top right to the text "inspect use of resources".

Code snippets shown in the notebook:

```
[4] 0s
niter = 10000
results = data.frame(iter = 1:niter, pvalue = NA)
N = 50; b0 = 0; b1 = 0.5; sigma = 1

[5] 13s
for(i in 1:niter){
  x = rnorm(N,0,1)
  df = data.frame(
    x = x,
    y = b0 + x*b1 + rnorm(N,0,sigma)
  )
  fit = lm(y ~ x, data = df)
  results$pvalue[i] = summary(fit)$coefficients["x","Pr(>|t|)"]
}

[6] 0s
(power = mean(results$pvalue < 0.05))
... 0.9166
```

# Let's Test the Environment!

Let's run a few commands in RStudio to familiarize with its console and see if the installation works properly

```
rnorm(10) # draw 10 random values from a Standard Normal distribution
```

```
[1] 2.1893126 -0.3920248  0.6233079 -0.3044554  1.0206581 -0.8628790  
[7] 0.9408118  1.1756534  0.6488329 -0.9160198
```

```
?rnorm # open the help tab for the "rnorm" function
```

```
round( rnorm(10, mean=100, sd=15) ) # draw 10 values from IQ distribution, round them
```

```
[1] 89 89 75 104 84 59 111 92 94 77
```

```
install.packages("psych") # install a package from CRAN
```

```
library(psych) # Load the newly installed package
```

```
fisherz(rho=0.5) # use it to transform a correlation into a Fisher's z
```

```
[1] 0.5493061
```