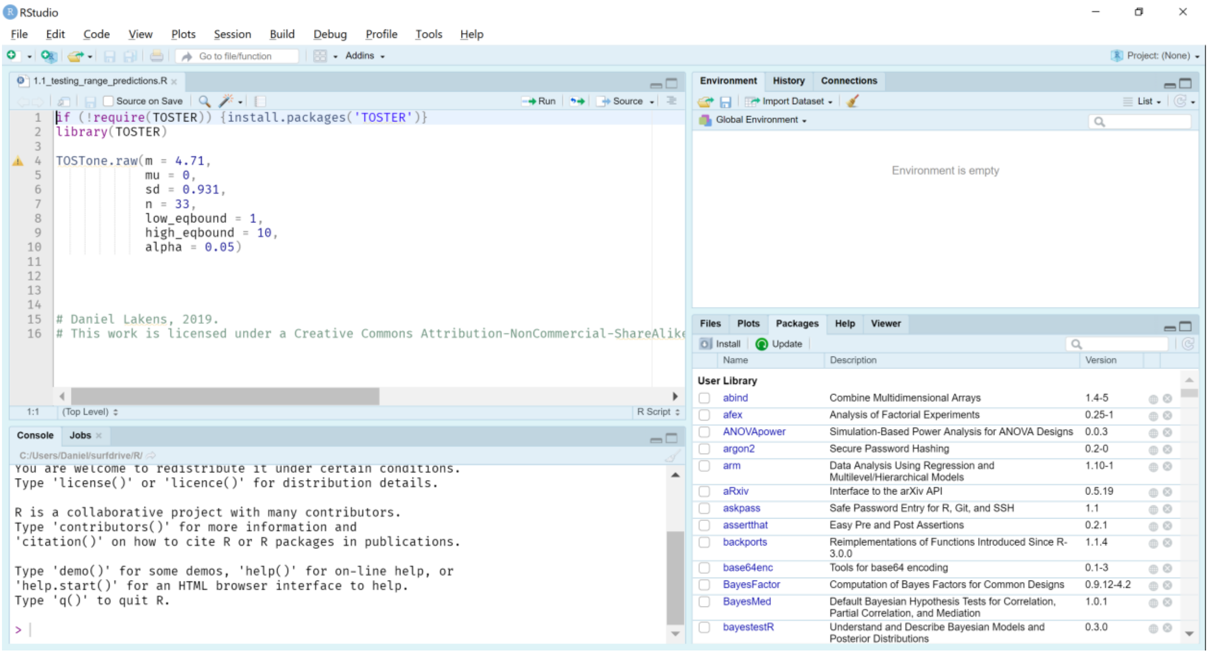
Running R code

In this course we will examine many statistical concepts through simulations or calculations in the free software R. You don’t need to know how to program in R – you just need to know how to run code. Here’s a detailed explanation for people who have never used R.

First install R. Then, **install R studio**.

Then, double-click the R script 1.1_testing_range_predictions.R. It should open in R Studio. If it doesn’t, open Rstudio, go to File>Open File, and open the script. You should see a screen like the screenshot below.



```r
if (!require(TOSTER)) {install.packages('TOSTER')}
library(TOSTER)

TOSTone.raw(m = 4.71,
            mu = 0,
            sd = 0.931,
            n = 33,
            low_eqbound = 1,
            high_eqbound = 10,
            alpha = 0.05)




# Daniel Lakens, 2019.
# This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike
```

```
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 
```

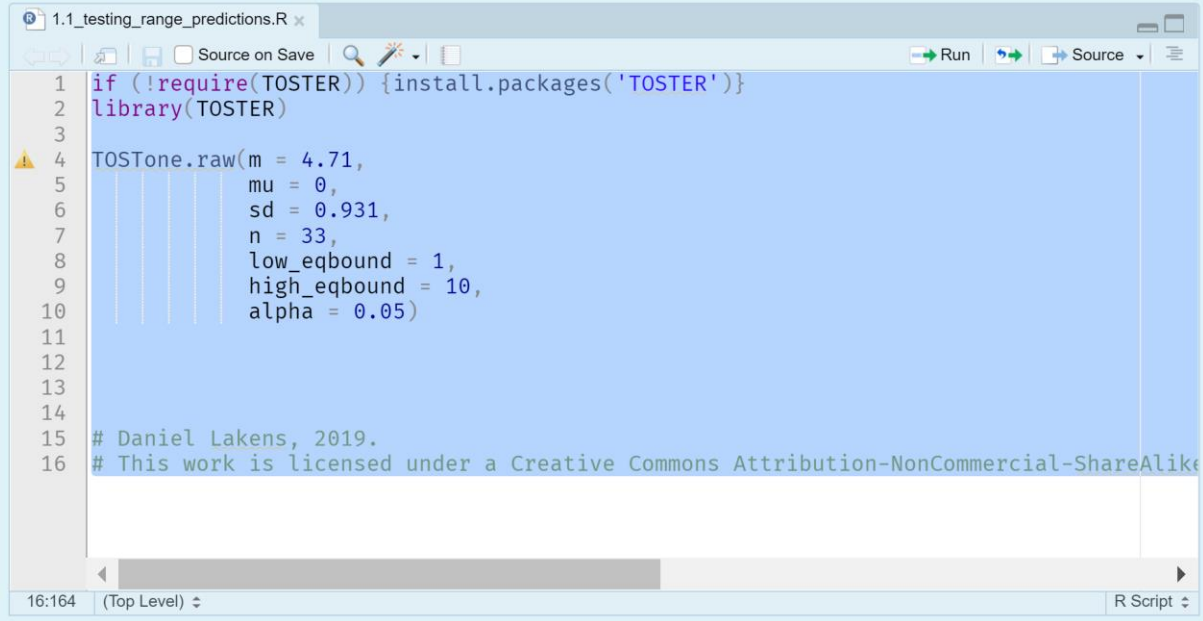
Environment is empty

	Name	Description	Version
	User Library		
□	abind	Combine Multidimensional Arrays	1.4-5
□	afex	Analysis of Factorial Experiments	0.25-1
□	ANOVApower	Simulation-Based Power Analysis for ANOVA Designs	0.0.3
□	argon2	Secure Password Hashing	0.2-0
□	arm	Data Analysis Using Regression and Multilevel/Hierarchical Models	1.10-1
□	aRxiv	Interface to the arXiv API	0.5.19
□	askpass	Safe Password Entry for R, Git, and SSH	1.1
□	assertthat	Easy Pre and Post Assertions	0.2.1
□	backports	Reimplementations of Functions Introduced Since R-3.0.0	1.1.4
□	base64enc	Tools for base64 encoding	0.1-3
□	BayesFactor	Computation of Bayes Factors for Common Designs	0.9.12-4.2
□	BayesMed	Default Bayesian Hypothesis Tests for Correlation, Partial Correlation, and Mediation	1.0.1
□	bayestestR	Understand and Describe Bayesian Models and Posterior Distributions	0.3.0

You see four sections in the four corners. Top left, you see the R script. In this window, you type new code, or change existing code. In the bottom left you see the console window, which gives output related to the code you are running, and where the results from calculations will appear. In the top right there is a window which will inform you about variables that have been defined – it’s useful for coding, but you won’t need it just to run code. In the bottom right, there is a window with 5 tabs. The only tab we need is the Plots tab, where plots that are created by R scripts will appear.

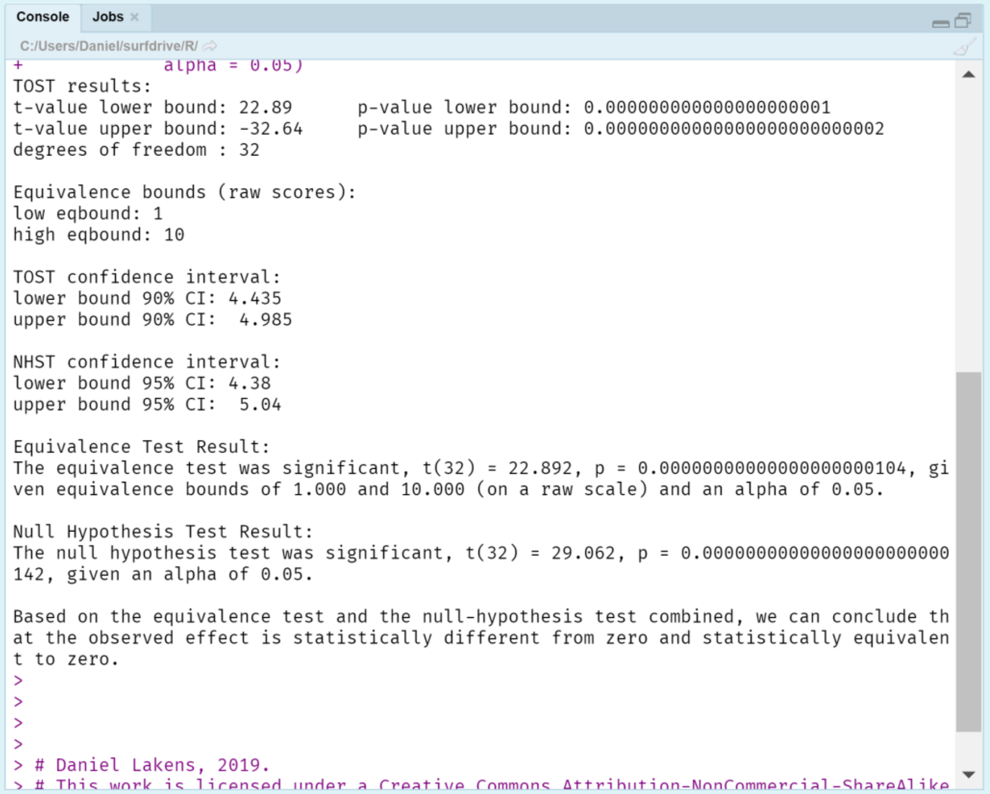
To interact with the R script, you need to click inside the top-left window. A blinking cursor should be visible.

If you want to run all the code in a script, you have to select all lines of the script, and run the code. The easiest way to do this is through keyboard shortcuts. Press **CTRL+A** to select all the code. All lines in the script window should be light blue:



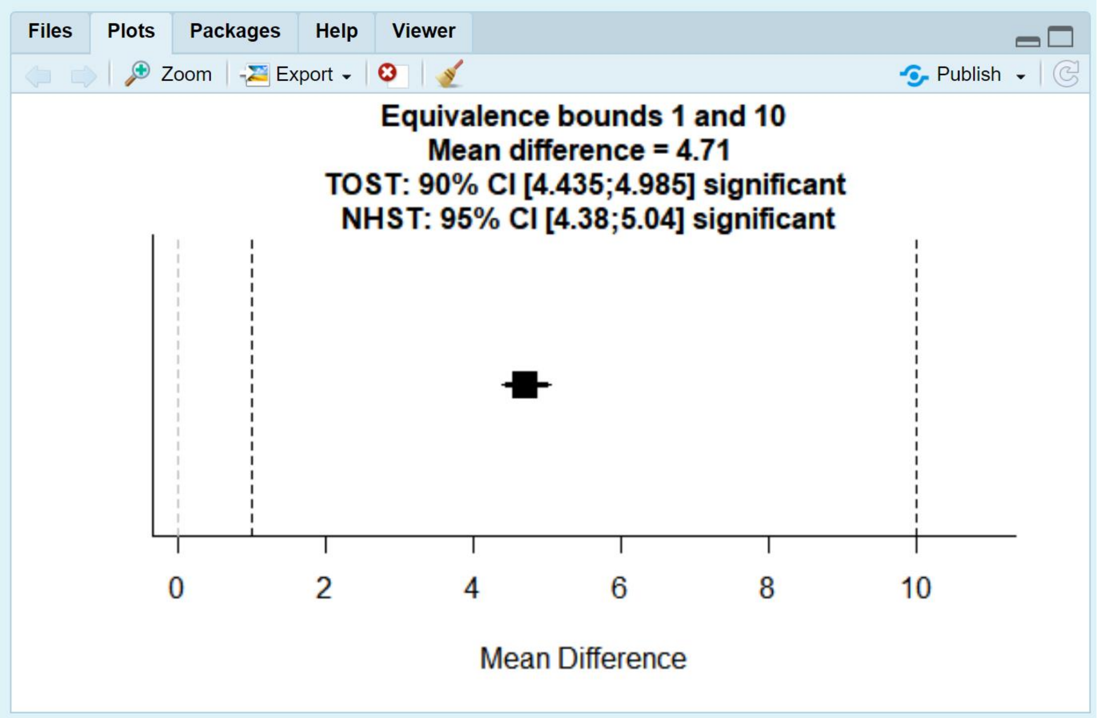
```
if (!require(TOSTER)) {install.packages('TOSTER')}
library(TOSTER)

TOSTone.raw(m = 4.71,
            mu = 0,
            sd = 0.931,
            n = 33,
            low_eqbound = 1,
            high_eqbound = 10,
            alpha = 0.05)





# Daniel Lakens, 2019.
# This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlik
```

Then, you can use the keyboard shortcut **CTRL+ENTER** to run the selected code. If you prefer to use the mouse, you can click on the Run icon in the upper bar of the script window. Look at the console window:

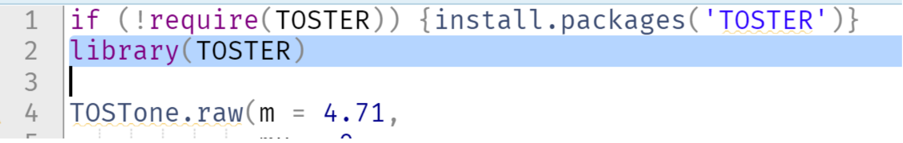


```
+             alpha = 0.05)
TOST results:
t-value lower bound: 22.89      p-value lower bound: 0.000000000000000000001
t-value upper bound: -32.64     p-value upper bound: 0.00000000000000000000000002
degrees of freedom : 32

Equivalence bounds (raw scores):
low eqbound: 1
high eqbound: 10

TOST confidence interval:
lower bound 90% CI: 4.435
upper bound 90% CI:  4.985

NHST confidence interval:
lower bound 95% CI: 4.38
upper bound 95% CI:  5.04

Equivalence Test Result:
The equivalence test was significant, t(32) = 22.892, p = 0.00000000000000000000104, gi
ven equivalence bounds of 1.000 and 10.000 (on a raw scale) and an alpha of 0.05.

Null Hypothesis Test Result:
The null hypothesis test was significant, t(32) = 29.062, p = 0.00000000000000000000000
142, given an alpha of 0.05.

Based on the equivalence test and the null-hypothesis test combined, we can conclude th
at the observed effect is statistically different from zero and statistically equivalen
t to zero.
>
>
>
>
> # Daniel Lakens, 2019.
> # This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike
```

If you scroll up you can see the code you have run. And output from calculations. Furthermore, a plot has appeared in the bottom right window:


Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
Equivalence bounds 1 and 10
Mean difference = 4.71
TOST: 90% CI [4.435;4.985] significant
NHST: 95% CI [4.38;5.04] significant
0
2
4
6
8
10
Mean Difference

Let's go back to the script window. Sometimes, you might want to select only a few lines of code. Notice how each line in the script window has a number on the left of the code, starting at 1. This makes it easy to refer to lines in the code. Scroll down to line 2 of the code. Put your cursor at the beginning of line 2, and select only this line. I like to use the keyboard shortcut **SHIFT+DOWNARROW**. Now, a single line is selected.

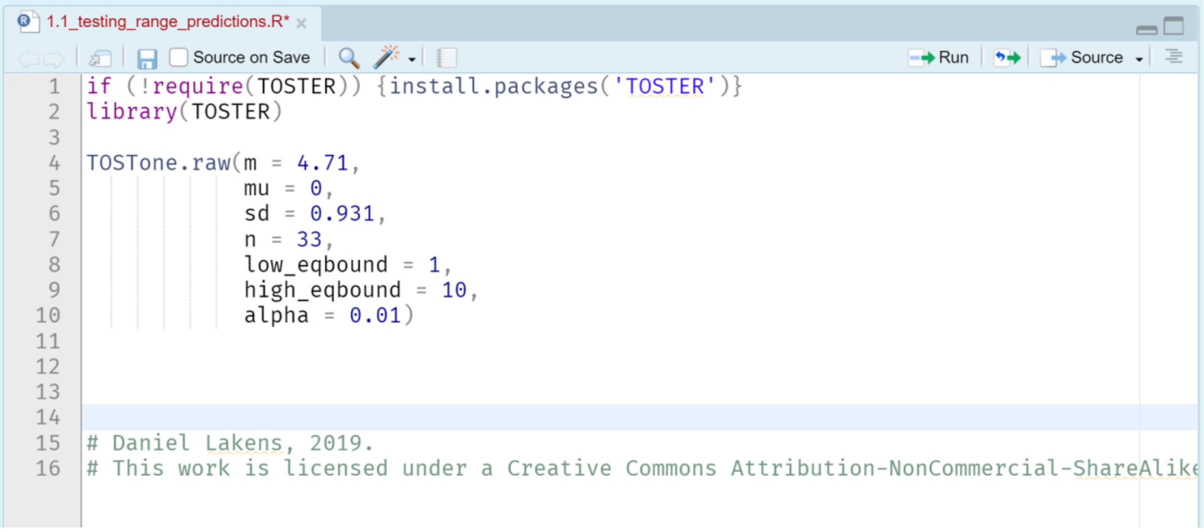


```
1  if (!require(TOSTER)) {install.packages('TOSTER')}
2  library(TOSTER)
3  
4  TOSTone.raw(m = 4.71,
```

Run this single line by CTRL+ENTER. In this single line, the TOSTER package is loaded. R requires you to load packages you need (there are a huge number of packages, and they are not all included in the default R installation).

It's easy to change a small part of the code by just typing and replacing. In line 10, you will see alpha = 0.05. You can read the "=" combination as 'is assigned the value' or 'becomes'. R also often used the combination "<-" for this, a left-pointing arrow. So the variable alpha (which is used by the TOSTER function) is assigned the numerical value 0.05.

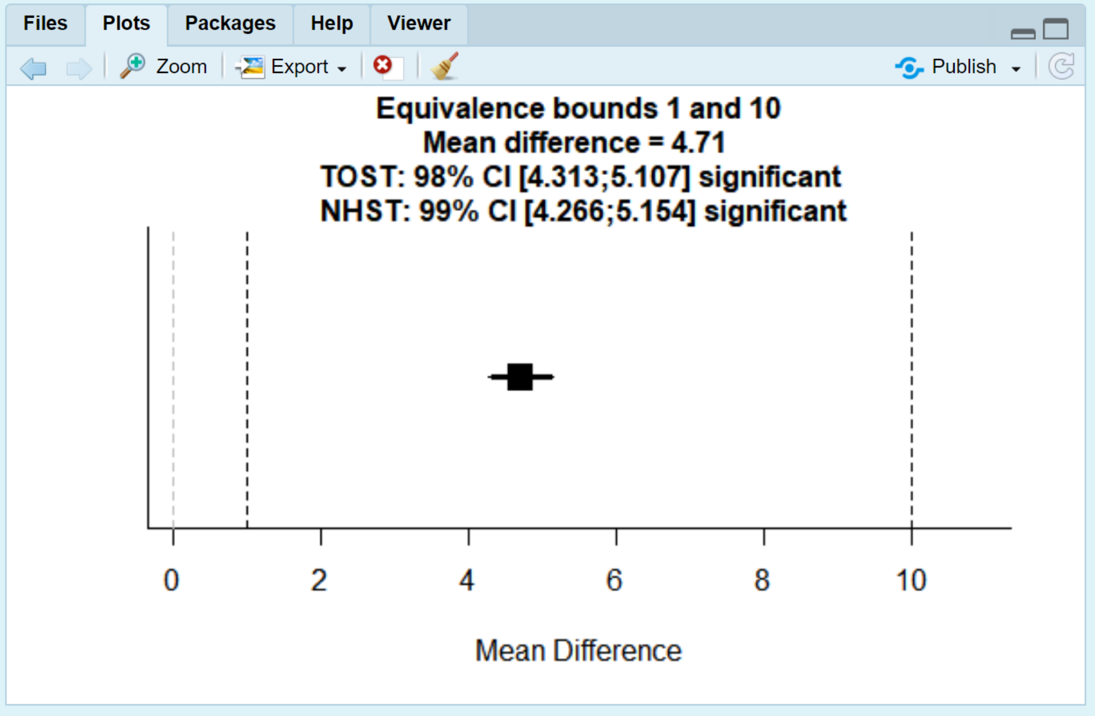
Change the alpha = 0.05 to alpha = 0.01.



```
if (!require(TOSTER)) {install.packages('TOSTER')}
library(TOSTER)

TOSTone.raw(m = 4.71,
            mu = 0,
            sd = 0.931,
            n = 33,
            low_eqbound = 1,
            high_eqbound = 10,
            alpha = 0.01)




# Daniel Lakens, 2019.
# This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike
```

If you look in the script window, the name of the script in the top has colored red to indicate the code has changed. If you save the code, you will need to decide if you want to save or don't save these changes.

Run the entire code again. You will see a new plot:



If you want to see the old plot, in the upper bar of the plot tab, you can click on back and forward arrows (⇦ ⇨) to scroll through the plots you have generated.

That’s all you need to know to run the code for all assignments in this course. You might still get stuck now and then. It happens when programming. Remember my favorite quote from the book ‘Zen and the Art of Motorcycle Maintenance’: *Stuckness shouldn’t be avoided. It’s the psychic predecessor of all real understanding.*

Good luck!