How to install R and RStudio

R (www.r-project.org) is a free statistical programming language.

Installing R

You first need to install the R program on your computer.

Installing R on a Windows PC

1. Go to https://cran.r-project.org/
2. Click on “Download R for Windows”
3. Under “Subdirectories”, click on the “base” link.
4. On the next page, you should see a link saying something like “Download R 3.6.2 for Windows”. Click on this link.
5. Save the file on the Desktop. Then double-click on the icon to run the installation file.
6. You will be asked what language to install it in - choose English
7. Confirm the following dialogues with “Next” at the bottom of the R Setup wizard window.
8. When R has finished, you will see “Completing the R for Windows Setup Wizard” appear. Click on “Finish”.

How to install R on Apple or Linux computers

To install R on a computer with Mac OS X or Linux download the appropriate R installer for that operating system at https://cran.r-project.org/ and follow the R installation instructions for the appropriate operating system at https://cran.r-project.org/bin/macosx/ (Apple) or https://cran.r-project.org/bin/linux/ (Linux).

Installing RStudio

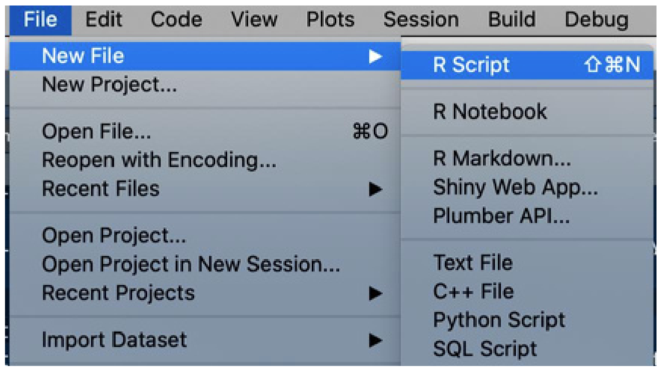
RStudio offers a user-friendly integrative interface to interact with R.
According to https://rstudio.com/products/rstudio/download/ “RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, and a variety of robust tools for plotting, viewing history, debugging and managing your workspace”. Download the *free* version of RStudio from https://rstudio.com/products/rstudio/download/#download and choose the appropriate installer depending on your operation system (Windows, OS X or Linux) and install RStudio. Leave all default settings in the installation options.
1. Open RStudio.

How to run R in RStudio (first baby steps)

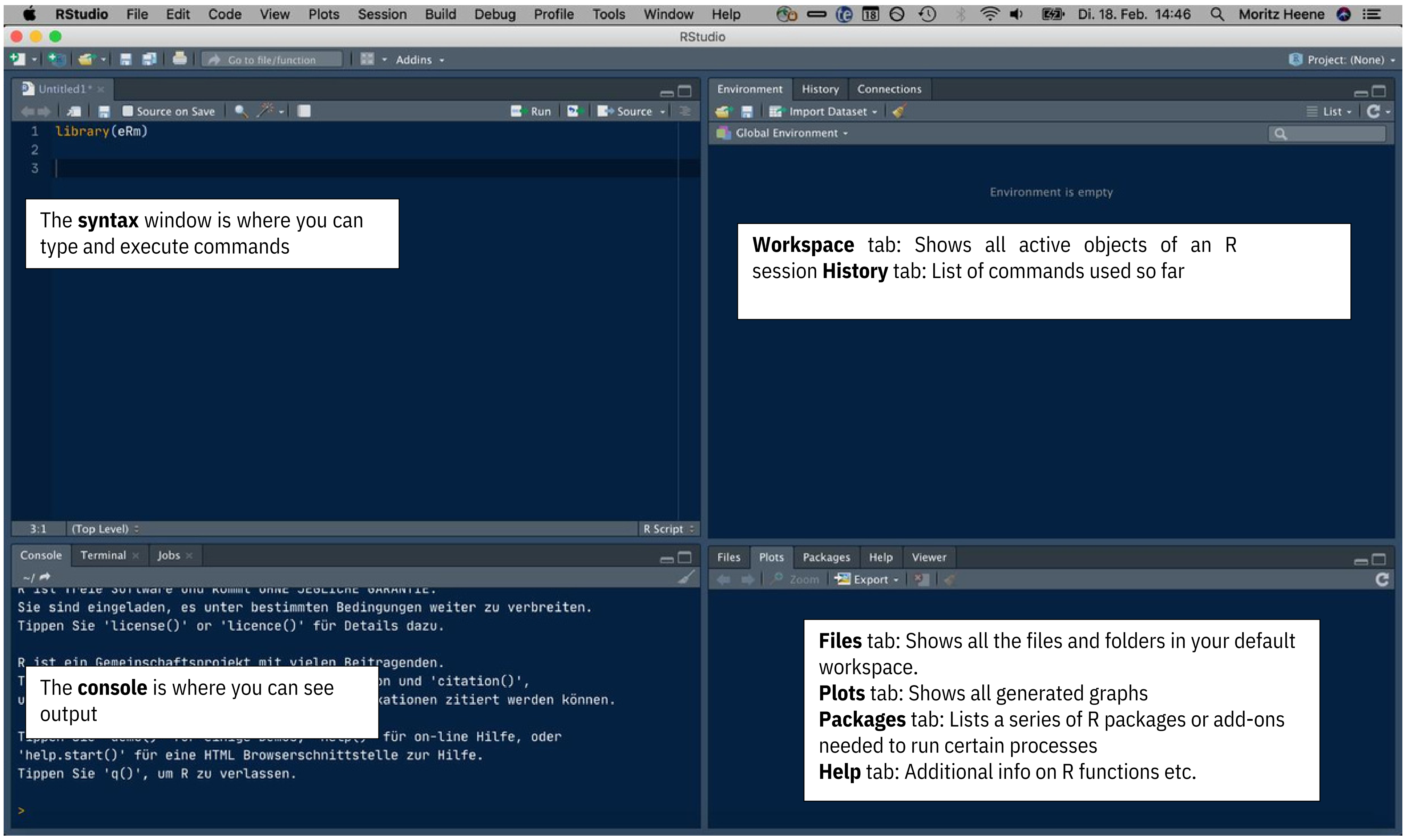
R is not a user-friendly, point-and-click software, but a programming language. That means every calculation or data manipulation, every variable/item selection or reading of data sets needs to be coded in lines of R programming language. On the one hand, this increases flexibility of data manipulation and analysis enormously, and this is one of the reasons why the most advanced and state-of-the-art Rasch model applications (and other analytical procedures) have been implemented into various R packages. On the other hand, that added flexibility and sophistication, also adds a layer of complexity which usually poses a major hurdle for those applied researchers who have been unfamiliar with coding. Fortunately, there are innumerable online sources about how to get started with R. The following introduction therefore just shows the absolute minimum of how to *execute* code in RStudio, so you can follow and execute the R language tutorials for ARM4; it is not aimed at providing a general introduction to R, that would be beyond the scope of these tutorials.

You have downloaded the R and RStudio resources appropriate for your computer's operating system, so let's take our first baby steps towards using R.

1. To familiarize yourself with RStudio's interface, first create a new syntax file as follows:



2. Give the file a name **with the extension .R** (for instance myfirstRscript.R) and save it on your desktop or somewhere else where you can easily find it. To do so, go to "File" "Save As…" and chose a location.
Your interface should then look like the one below:



```
library(eRm)
```

The **syntax** window is where you can type and execute commands

Workspace tab: Shows all active objects of an R session **History** tab: List of commands used so far

```
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.
T                                        on und 'citation()',
u                                        kationen zitiert werden können.

T                                        für on-line Hilfe, oder
'help.start()' für eine HTML Browserschnittstelle zur Hilfe.
Tippen Sie 'q()', um R zu verlassen.

>
```

The **console** is where you can see output

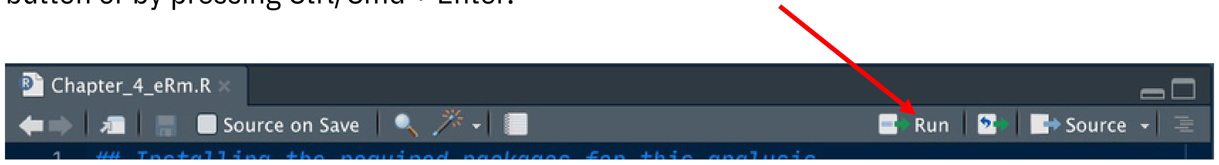
Files tab: Shows all the files and folders in your default workspace.
Plots tab: Shows all generated graphs
Packages tab: Lists a series of R packages or add-ons needed to run certain processes
Help tab: Additional info on R functions etc.

To get used to RStudio and R, type in the following commands into the syntax window:

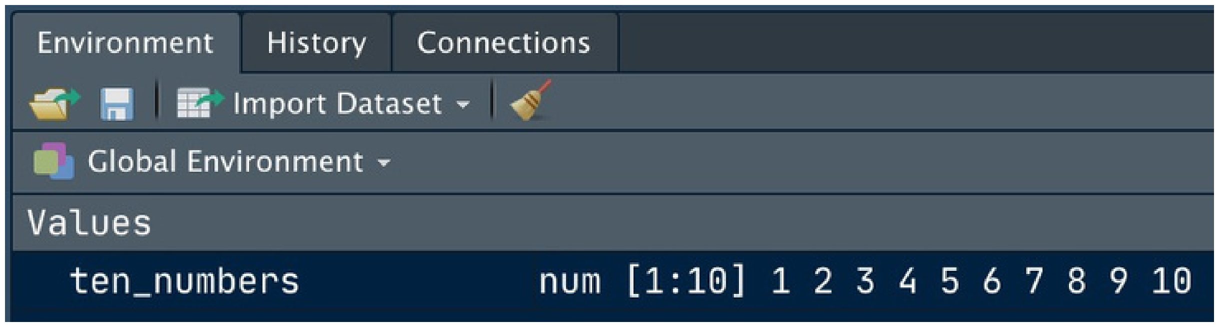
```
ten_numbers <- c(1,2, 3, 4, 5, 6, 7, 8, 9, 10)
```

(Hint: To insert the arrow as the assignment operator, users of Windows or Linux can press "Alt + -" and Mac users "Option + -").
This code combines the numbers 1 to 10 into an object called ten_numbers", and the arrow serves as the assignment operator. However, these numbers are not assigned yet because the code has not been executed. To do so, you need to run the code.

You can run a line from your script by putting the cursor next to it and clicking the "Run" button or by pressing Ctrl/Cmd + Enter.



The result will then appear in the "Environment" tab:

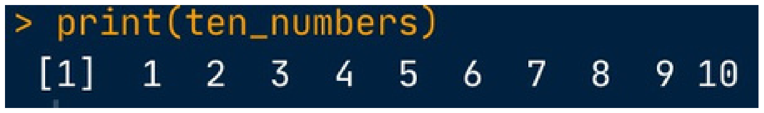


```
ten_numbers          num [1:10] 1 2 3 4 5 6 7 8 9 10
```

Alternatively, you can also execute a code selection by highlighting it and clicking "Run" or pressing Ctrl/Cmd + Enter.
Now, type in

```
    print(ten_numbers)
```

This will print the object "ten_numbers". The result appears in the "Console" tab:



```
> print(ten_numbers)
 [1]  1  2  3  4  5  6  7  8  9 10
```

Let us now say you want to know the sum, the median, and the standard deviation of the object called "ten_numbers", and to save these values because you want to reuse them for some later calculations. The corresponding R functions for the sum, the median, and the standard deviation are

```
sum()
```

median()
sd()

The objects for which you want to get these statistics must be entered within the brackets. To save the results to an object, we also need to assign them to an object using the assignment operator (<-) and provide an object name. You can name an R object anything but it is highly advisable to give it a meaningful name. For example, an object called sum_ten_numbers is more meaningful than just x. Furthermore, blank spaces within an object names are not allowed because R would not recognize it as a single object name. So, sum_ten_numbers would work, whereas sum ten numbers would not. To avoid that, use either an underscore (_) or a dot (.) to separate words of an object name.
So, as for our example, the corresponding R code could be

```
sum_ten_numbers <- sum(ten_numbers)
median_ten_numbers <- median(ten_numbers)
sd_ten_numbers <- sd(ten_numbers)
```

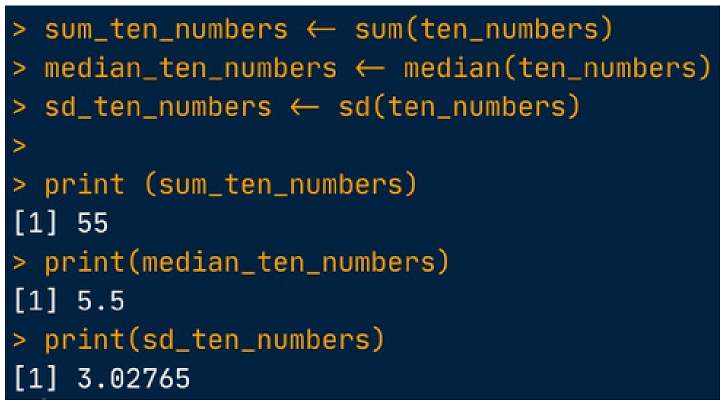
To print the results after we executed this code we use the function print() as follows:

```
print(sum_ten_numbers)
print(median_ten_numbers)
print(sd_ten_numbers)
```

Please finally notice that R is case-sensitive (see https://cran.r-project.org/doc/manuals/r-release/R-intro.html#R-commands_003b-case-sensitivity-etc for details), that is, it treats uppercase and lowercase letters as distinct. Relating this to our example, print (sum_ten_numbers) would correctly print the result but, for instance, print (Sum_ten_numbers) or print(sum_ten_Numbers) would not (i.e., would give an error message).

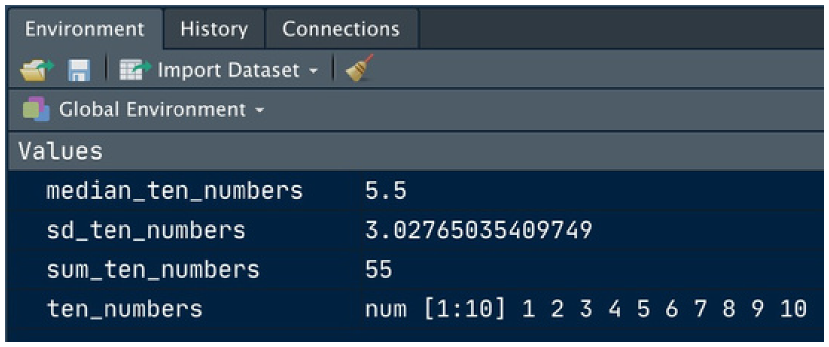
Now, rewrite the code given above and run it.

The results should then show up in the console:



```
> sum_ten_numbers <- sum(ten_numbers)
> median_ten_numbers <- median(ten_numbers)
> sd_ten_numbers <- sd(ten_numbers)
>
> print (sum_ten_numbers)
[1] 55
> print(median_ten_numbers)
[1] 5.5
> print(sd_ten_numbers)
[1] 3.02765
```

Note that the values will also show up in the “Environment” panel:



```
Values
  median_ten_numbers     5.5
  sd_ten_numbers         3.02765035409749
  sum_ten_numbers        55
  ten_numbers            num [1:10] 1 2 3 4 5 6 7 8 9 10
```