

Data analysis using R for Psychology

Alexander (Sasha) Pastukhov

2020-10-29

Contents

Introduction	5
About the seminar	5
Note on exercises	6
Why R?	6
Tidyverse versus base R	7
Getting Started	9
Installing R	9
Installing R-Studio	9
Installing RTools	9
Installing packages	10
Keeping R and packages up-to-date	11
Reproducible Research and RMarkdown Notebooks	13

Introduction

About the seminar

This is a material for *Applied data analysis for psychology using the open-source software R* seminar. Each chapter covers a single seminar, introducing necessary ideas and is accompanied by a notebook with exercises, which you need to complete and submit. The material assumes no foreknowledge of R or programming in general from the reader. Its purpose is to gradually build up your knowledge and introduce to a typical analysis pipeline. It is based on a data that is typical for the field (repeated measures, appearance, accuracy and response time measurements, Likert scale reports, etc.), you are welcome to suggest your own data set for analysis. Even if you already performed the analysis using some other program, it would still be insightful to compare the different ways and, perhaps, you might gain a new insight. Plus, it is more engaging to work on your data.

Remember that throughout the seminar you can and should(!) always ask me whenever something is unclear, you do not understand a concept or logic behind certain code. Do not hesitate to write me in the team or (better) directly to me in the chat (in the latter case, the notifications are harder miss and we don't spam others with our conversation).

You will need to submit your assignment one day before the next seminar (Tuesday before noon at the latest), so I would have time evaluate it and provide feedback.

As a final assignment, you will need to program a full analysis pipeline for a given data set (or, if you want, for your data set). All the necessary steps will be covered by the seminar material. Please inform me, If you require a grade, as then I will create a more specific description for you to have a clear understanding of how the program will be graded.

Note on exercises

In many exercises you will be not writing the code but understanding it. Your job in this case is “to think like a computer”. Your advantage is that computers are very dumb, so instructions for them must be written in very simple, clear, and unambiguous way. This means that, with practice, reading code is easy for a human (well, reading a well-written code is easy, you will eventually encounter “spaghetti-code” which is easier to rewrite from scratch than to understand). In each case, you simply go line-by-line, doing all computations by hand and writing down values stored in the variables (if there are too many to keep track of). Once you go through code in this manner, it will be completely transparent for you. No mysteries should remain, you should have no doubts or uncertainty about any(!) line. Moreover, then you can run the code and check that the values you are getting from computer match yours. Any difference means you made a mistake and code is working differently from how you think it does. In any case, **if you not 100% sure about any line of code, ask me, so we can go through it together!**

In a sense, this is the most important programming skill. It is impossible to learn how to write, if you cannot read first! Moreover, when programming you will probably spend more time reading the code and making sure that it works correctly than writing the new code. Thus, use this opportunity to practice and never use the code that you do not understand completely. So do use stackoverflow but do make sure you understand the code you copied!

Why R?

There are many software tools that allow you preprocess, plot, and analyse your data. Some cost money (SPSS, Matlab), some are free (Python, Julia) just like R. Moreover, you can replicate all the analysis that we will perform using Python in combination with Jupyter notebooks (for reproducible analysis), Pandas (for Excel-style table) and statmodels (for statistical analysis). However, R in combination with Tidyverse family of packages is optimized for data analysis pipeline, making it easy to write simple, powerful and expressive code that is very easy to understand (a huge plus, as you will discover). I will run circles around myself trying to replicate the same analysis in Python or Matlab. In addition, R is loved by mathematicians and statisticians, so it tends to have implementations for all cutting edge methods (my impression is that even Python is lagging behind it in that respect).

Tidyverse versus base R

I will be teaching what one might call a “dialect” of R, based on Tidyverse family of packages. R is extremely flexible, making it possible to redefine its own syntax. Because of that Tidyverse-based code will look very different from the base R code to the point that it might look like written in two completely different languages (which, in a sense, it is). Although Tidyverse, at least in my opinion, is a better way, we will start with *base R*, so that you will be able to read and understand code written outside of Tidyverse, as it is also very common.

Getting Started

Installing R

Go to r-project.org and download the current stable version of R for your platform. Run the installer, accepting all defaults. The installer will ask you whether you also want the 32-bit version to be installed alongside 64-bit. You probably won't need 32-bit, so if space is at premium you can skip it. Otherwise, it will make very little difference.

Installing R-Studio

Go to rstudio.com and download *RStudio Desktop Free* edition for your platform. Install it using defaults. The *R-Studio* is an integrated development environment for R but you need to install R separately first! The R-Studio will automatically detect latest R that you have and, in case you have several versions of R installed, you will be able to alter that choice.

I will explain the necessary details on using R-Studio throughout the seminar but the official cheatsheet is an excellent, compact, and up-to-date source of information. In fact, R Studio has numerous cheatsheets that describe individual packages in a compact form.

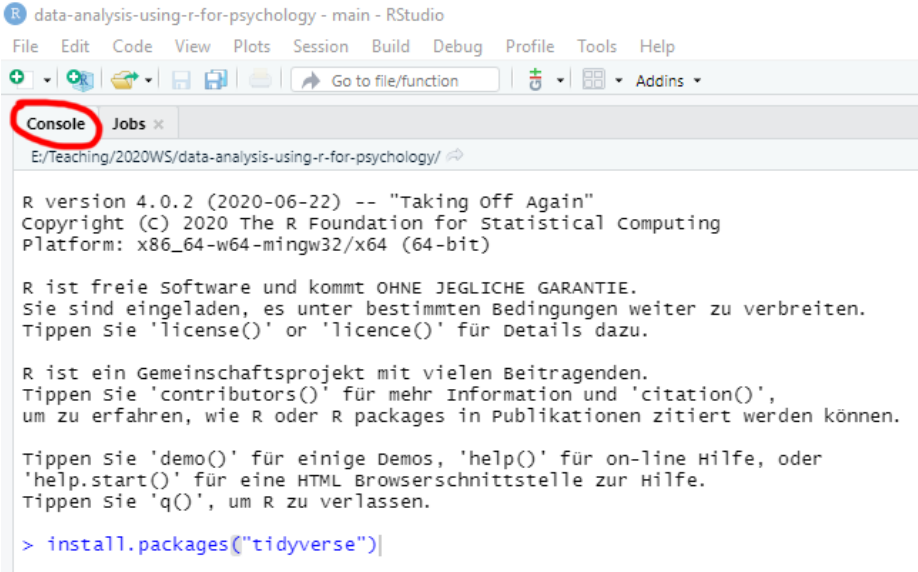
Installing RTools

If you are using Windows, we might need Rtools for building and running some packages. You do not need to install it at the beginning, but when we will need it later, just following the link above, download the latest *Rtools* version, run the installer using the defaults and **follow the instructions on that page to put Rtools on the PATH!** (I do not repeat them here, because they might change).

Installing packages

The real power of R lies in a community-driven vast family of packages suitable for any occasion. The default repository used by R and R-Studio is The Comprehensive R Archive Network (a.k.a. *CRAN*). It has very strict requirements for submitted packages, which makes it very annoying for the authors but ensures high quality of most packages, as they tend to be well-documented and come with example data and code. We will use CRAN as a sole source for packages, but there are alternatives, such as Bioconductor that might have a package that is missing at CRAN. The Bioconductor relies on its own package manager, so you will need to consult latest manual on their website.

To install CRAN package you have two alternatives: via command line function or via R-Studio package manager interface. In the former case, go to **Console** tab and type `install.packages("package-name")`, for example `install.packages("tidyverse")`, and press **Enter**.

The image is a screenshot of the RStudio application window. The title bar reads "data-analysis-using-r-for-psychology - main - RStudio". The menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu bar is a toolbar with various icons. The "Console" tab is selected and highlighted with a red circle. The console output shows the R version (4.0.2), copyright information, and a list of helpful commands like 'demo()', 'help()', and 'help.start()'. At the bottom of the console, the command `> install.packages("tidyverse")` has been entered, with the cursor at the end of the line.

```
R version 4.0.2 (2020-06-22) -- "Taking off Again"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

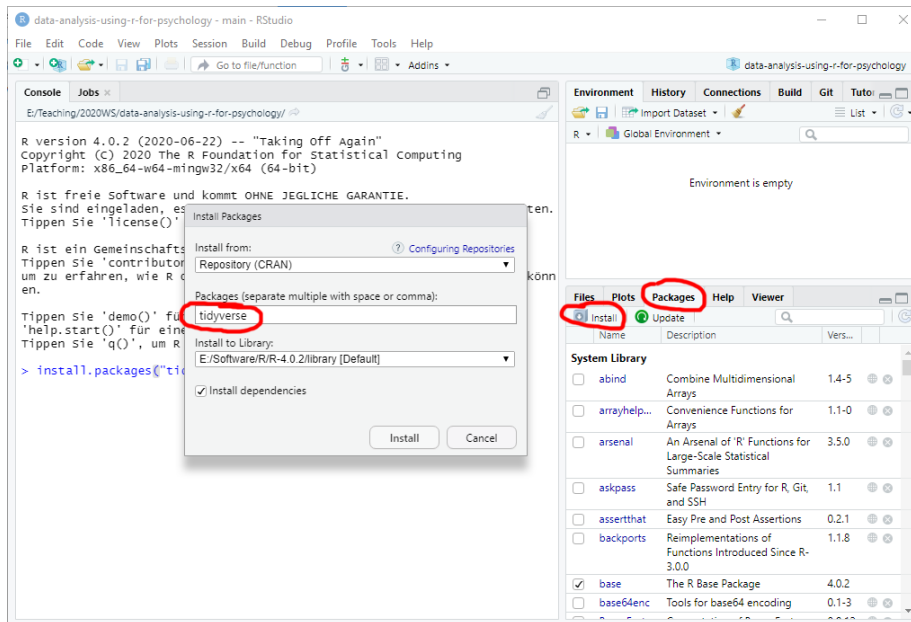
R ist freie Software und kommt OHNE JEGLICHE GARANTIE.
Sie sind eingeladen, es unter bestimmten Bedingungen weiter zu verbreiten.
Tippen Sie 'license()' or 'licence()' für Details dazu.

R ist ein Gemeinschaftsprojekt mit vielen Beitragenden.
Tippen Sie 'contributors()' für mehr Information und 'citation()',
um zu erfahren, wie R oder R packages in Publikationen zitiert werden können.

Tippen Sie 'demo()' für einige Demos, 'help()' für on-line Hilfe, oder
'help.start()' für eine HTML Browserschnittstelle zur Hilfe.
Tippen Sie 'q()', um R zu verlassen.

> install.packages("tidyverse")|
```

Alternatively, go to **Packages** tab, click on **Install** button, enter the package name in the window, which has autocomplete to help you, and press **Install**.



In some cases, R will ask whether you want to install packages from source. In this case, it will grab the source code and compile the package, which takes time and requires RTools. In most cases, you can say “No” to install a pre-build binary version. The binary version will be slightly outdated but the emphasis is on *slightly*.

Please install the following packages:

- **tidyverse** : includes packages from data creation (**tibble**), reading (**readr**), wrangling (**dplyr**, **tidyr**), plotting (**ggplot2**). Plus type specific packages (**stringr** for strings, **forcats** for factors) and functional programming (**purrr**).
- **rmarkdown** : package for working with RMarkdown notebooks, which will we use to create reproducible analysis.
- **fs** : file system utilities.

Keeping R and packages up-to-date

R and packages are getting constantly improved, so it is a good idea to regularly update them. For packages, you can use **Tools / Check for Packages Updates...** menu in R-Studio. To update R and, optionally, packages, you can use **installr** package that can install newest R (but it keeps old version!) optionally copying your entire library of packages, update packages, etc. For

R-Studio itself, use **Help / Check for Updates** menu and install a newer version, if it is available (it is generally a good idea to keep your R-Studio in the newest state).

Reproducible Research and RMarkdown Notebooks