Installation of the necessary software

Claudius Gräbner-Radkowitsch

2025-09-19

Table of contents

1 Introduction

During this course we will use the following software and services:

- R
- R-Studio
- Git
- Github
- Netlify

You will need to install the software and register for these services on your own. While I offer an optional session for joint troubleshooting, it is absolutely necessary that you do your best to install the software on your own before that date. To this end, this document is meant to provide you with all the information needed. If you have questions, *please use the Moodle forum*. It is very unlikely that you are the only person having a particular problem. Maybe others can already help you out, and if not, all should benefit from the solution we find for your problem together.

Please also note that there is a separate tutorial on how to install the packages (a.k.a. R extensions) that we are going to use over the semester. Its best to continue with this tutorial on installing the required R packages directly after you have complete this one.

2 An overview over the apps and services we will be using - and why

R is the programming language we are using. Installing `R` basically gives your computer the ability to understand commands you will issue to it using the language `R`. R-Studio is an application that facilitates the development of `R` code. It is a so called *Integrated Development Environment* (IDE). You may think of it as a fancy editor, which not only allows you to write programs in `R` using a more elaborated text editor, but simultaneously lets you preview the graphs you are designing, or the reports you are writing. Because R-Studio facilitates the use of `R`, it is important to install R *first*, and then install R-Studio *second*.

Git is a so called *version control system* (VCS). It allows you to keep track of different versions of your scripts without saving them under different names. In other words, if you use Git, you can dispense of saving `Script-v1.R`, `Script-v2.R` and `Script-v3.R`, but instead only keep `Script.R` and save the different versions in the background, together with some comments about what you have changed. Moreover, it facilitates the joint work on the same document, something that is tremendously helpful for group works. While we will not learn how to use Git (I will provide some optional tutorials, though), it is important to install it since many user-written features of R are distributed in a way that relies on Git.

Github can be thought of as a server for Git. `R` is an open source programming language with a very active community. Many people are constantly developing new features for `R`. For instance, if a new prediction algorithm get developed, some people will soon write a so called R-package that implements this algorithm in `R`. Then you as an R user can download this package and use the new algorithm. This way, `R` is unlikely to ever become outdated. Since most developers use Git when developing the packages, they often distribute the programs in a way that aligns well with the use of Git. And the most prominent way is Git*hub*. Thus, you can view the source code for almost all `R` packages you will be using via Github. Github is free, and all you need is to register!

The final service we will be using is Netlify. It is an easy way to publish reports and host small websites. We will use it such that you can create reproducible reports and present them in a visually appealing way. Given the ease of use, it is a great tool if you want to polish your results up visually and send them to friends or colleagues who might not even use R themselves.

3 Installation guidelines

The information in this tutorial is provided for Mac OS, Windows, and Linux. Given the variety of different Linux distributions the comments on the latter are rather short.

There are, however, plenty explanations to be found on the web for many different Linux distributions.

After you have completed the steps outlined below you are almost done: all what remains is to install the R packages we require over the course of the seminar. To this end, please consult the respective Tutorial.

3.1 Install R

The installation of `R` is very similar across operating systems (OS). The easiest way is to visit the R Homepage and to download the most recent version for your OS. In case you are using Mac OS and want to use Homebrew, its best to use this formula.

Important for Mac user: There are different versions of `R` for Intel chips, and Apple chips (M1, M2, etc.). It is very important that you install the correct version. If you are not sure whether your Mac contains a chip from Intel or Apple, click on the Apple symbol in the upper left of the screen, then click on `About this Mac` and you can see which processor your Mac is using in the new window. If you have an Apple chip, always install R for the so called `arm64` architecture. Intel chip users must use the `x86_64` architecture instead.

3.1.1 Only Windows: Install RTools

If you are using Windows, it is necessary to install `RTools`, which is required if you want to use packages written by others that are not officially released. To do so, simply visit the following website, download the installer, and install the software:

When asked during the installation process, do *not* select the box for `Add rtools to system PATH`, but *do* select the box for `Save version information to registry`.

3.1.2 Only Mac: Command Line Developer Tools

The Command Line Developer Tools could be thought of as the Mac pendant to RTools. These allow you to build R packages from source (meaning, basically, you can use packages that are in early stages of distribution, or packages that are not released on the official R servers).

The easiest way to install them is to open the App `Terminal`, and then to type

```
xcode-select --install
```

and press `Enter`. Then a pop up window will open and allow you to install the software.

3.2 Update R

In case R is already installed on your computer you should make sure that your version is more of less up to date. For our seminar you should use at least R version `R 4.4.2`. The version you are currently using is shown as soon as you start `R`.

Please note: if you installed R anew in the previous step, you do *not* need to update it. The information on updating R is mainly relevant for people who have installed R already some time ago.

3.2.1 MacOS users

For **MacOS users**, the easiest route to update `R` is to just re-install the most current version from the R Homepage. Keep in mind that in this case you might need to re-install all previously installed packages. If you have a lot of packages installed that you want to keep, the following steps facilitates the re-installation process. First, save a list with all the packages you installed yourself. To this end type the following into the R console:[1]

```
package_overview <- installed.packages()
package_names <- as.vector(
  package_overview[is.na(package_overview[,"Priority"]), 1])
save(package_names, file="r_packages.rda")
```

After re-installing R, you then need to load the file you previously saved and identify the missing packages. You can use the following code to do so if you are in the working directory in which you saved the file `"r_packages.rda"`:

```
load("r_packages.rda")
packages_new <- installed.packages()
packages_new_ <- as.vector(packages_new[is.na(packages_new[,"Priority"]), 1])
missing_packages <- setdiff(package_names, packages_new_)
install.packages(missing_packages)
update.packages()
```

3.2.2 Windows users

Windows users have a slightly more convenient route available to them: the installr package. It does not require you to re-install your packages. Just type the following code into your R console:[2]

```
install.packages("installr")
library(installr)
updateR(TRUE)
```

[1] If you do not yet know what the R console is don't worry. You will learn this during the course. But for now it would then be better to update R by just re-installing it.

[2] If you do not yet know what the R console is don't worry. You will learn this during the course. But for now it would then be better to update R by just re-installing it.

For more information see the package website.

3.2.3 Linux users

Linux users simply install `R` via their package manager. A quick search on Google should provide you with the information that are relevant for your particular Linux distribution. Updating is usually straightforward as well: just run the respective command from your package manager.

3.3 R-Studio

Installing R-Studio is easy. The only thing you should keep in mind that you should **install R first**, and **R-Studio second**. So, after installing `R` got to the R-Studio download page and download the *RStudio Desktop* version for your OS according to the installation instructions provided.

If you are on Mac and you are using Homebrew you may use this formula.

If you want to update R-Studio, you just install it again. Please note that the **minimal version** for this seminar should be RStudio `2023.12.1+402`, which is from late January 2024. You can check your version by clicking on `RStudio` in the upper left part of your screen when R-Studio is open. Then click on `About RStudio`.

3.4 Git

Installing Git is straightforward, but the right approach depends on your OS.

3.4.1 MacOS

On **MacOS** you should install Git as part of the Command Line Developer Tools, which themselves are part of XCode (see above). Its easiest to run the following command from your Terminal:[3]

```
git --version
```

If you get an output such as `git version 2.34.1` you already installed you need. If not, you will be asked to install the respective software packages (see above).

3.4.2 Windows

On **Windows** you download `Git for Windows` from the official Webpage, which also provides you with all the relevant instructions.

[3] By this I mean that you first open the app `Terminal` and then enter the commend into the window that has opened, and then press `Enter`.

3.4.3 Linux

On **Linux** use you package manager. In most cases the name of the relevant package is `git-all`, so on Ubuntu, for instance, you would install Git via `sudo apt install git-all`.

3.5 Install Quarto

Quarto allows you to write text and R code within one document. This is very useful in many instances, and allows you to create a wide variety of nicely looking and practically appealing outputs, including apps, websites, statistical reports, and much more. To install Quarto just follow the instructions from this webpage.

4 Register for the necessary services

4.1 Github

This is easy. Just visit https://github.com/ and sign up using your email account.

4.2 Netlify

This is easy as well. Visit https://www.netlify.com/ and click on `Sign up` in the upper right of the webpage. You can now either create a Netlify account by clicking on `Email` and register a new email address, or you can link Netlify to one of the other accounts you might already have. I personally, for instance, linked Netlify to my Github account.

5 Next steps

Once you finished with installing all the relevant software described in this post I recommend you continue directly with the second (and last) installation tutorial that assists you in installing the `R` packages required over the course of the semester.

And, by the way, if this installation process seems frustrating to you: don't worry, its the only time during the semester you need to install anything;)