Data Literacy for the Language Sciences

A very gentle introduction to statistics and data visualisation in R

Elen Le Foll

2024-04-22

Table of contents

Preface

> **Warning**
>
> This textbook draft is very much **work in progress**. I intend to progressively add to it over the course of the summer semester 2024. Note that the PDF version is not optimised in any way. I recommend only looking at the web-book.
> This first draft is intended as complementary materials to my summer semester M.A. class: "More than counting words: Introduction to statistics and data visualisation for linguists" taught at the University of Cologne.
> Student feedback on this first draft is very welcome!

Who is this book for?

This textbook is intended as a very gentle introduction to basic principles of data management, statistics, and data visualisation using the programming language and environment `R`. The target audience are students and researchers in the language sciences, including (applied) linguistics, language teaching, and language education research. The rationale for this textbook is based on my personal observations, in teaching and consulting both students and researcher colleagues, that many so-called 'introductory' textbooks assume previous knowledge and skills that all have or go through contents at too fast a pace for many humanities scholars who often come with little to no experience with programming and/or statistics.

The aim of this textbook is by no means to replace any of the brilliant existing textbooks aimed at imparting statistical literacy for linguistics research, but rather to provide a stepping

stone towards being able to make the most of these wonderful existing resources. A (work-in-progress) list of next-step resources is included in Appendix A.

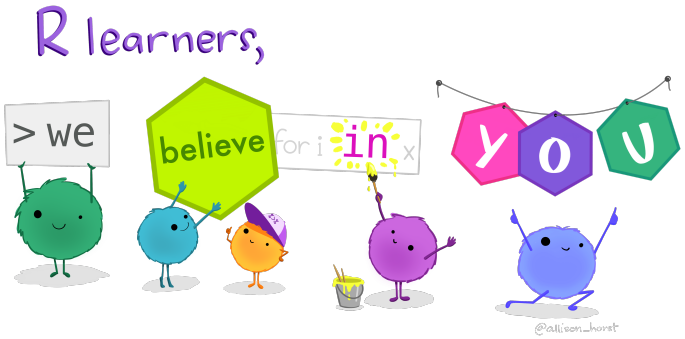


Figure 1: Artwork encouraging beginner `R` learners by @allison_horst

1 Open Scholarship

This book aims to provide a stepping stone for students and scholars of traditionally less quantitative and computational disciplines (such as some branches of linguistics and language education research) to gather first (hopefully positive!) experiences with statistical and computational approaches to working with empirical data[1]. The underlying belief is that these methods ought to be accessible to all, regardless of their academic background or personal circumstances. To this end, this book embraces the principles of Open Scholarship.

Open Scholarship "reflects the idea that knowledge of all kinds should be openly shared, transparent, rigorous, reproducible, replicable, accumulative, and inclusive (allowing for all knowledge systems)" (Parsons et al. 2022). For this to be the case, teaching materials need to shared openly and the tools and software taught in these resources need to be freely accessible, too. In the following, we will briefly consider the role of Open Educational Resources (OERs) and open-source software in our pursuit of Open Scholarship.

1.1 Open Source

In line with its aim to provide an accessible introduction to statistics and data visualisation, this textbook relies exclusively on open-source software and programming languages, foremost `LibreOffice Calc`, `R` and `RStudio`. Open source refers to software whose source code is available under a license that grants anyone the rights to study, modify, and distribute the software to anyone and for any purpose. If we think of a software application as a cake, the source code is like its recipe. It contains

[1] Emprical data is based on what is experienced or observed rather than on theory alone.

the list of ingredients and the steps to bake the cake. Open source means that the recipe is publicly available. You can access it, read it, and use it to bake the cake. You can also modify it to add your own twist, such as adding a new ingredient or making it vegan, and share it with others. In the context of software, this allows many people to collaborate, make improvements, and share their versions, resulting in better and more diverse software.

Using open-source software in this introductory textbook means that anyone[2] can download, install and use the required software at no cost. However, it is very important to note that not all free software (*freeware*) is open source. Let us illustrate the difference by comparing different spreadsheet programmes as, in the following chapter, we will begin exploring tabular data structures in a spreadsheet programme.

The most most widely used spreadsheet programme to date is undoubtedly `Microsoft Excel`. Excel is a commercial, proprietary spreadsheet editor which forms part of the Microsoft 365 package. As such, to use Excel on your personal computer, you need to buy a license or be a member of an organisation (e.g., your university or company) that pays for such a license. It is true that Microsoft now also offers a free (functionally limited) web-based version of Excel, yet this still does not make it open source. This is because Microsoft does not share the source code of any Excel version, which means that, even if they are giving away free cake, we do not have the recipe to bake the cake ourselves should the company decide to start charging money for the cake or to no longer distribute it at all! Similarly, you may be familiar with a popular, web-based alternative to Excel: `Google Sheets`. Whilst it is (currently) free to use, as the name suggests, Google Sheets is owned by Google and is therefore not open source, either. By contrast, `LibreOffice Calc` is a project of The Document Foundation (TDF) that provides a popular, free, open-source office productivity software suite comparable to Microsoft 365 called `LibreOffice`. LibreOffice is developed collaboratively by very many different people across the world who all do so on a volunteer basis.

[2] Provided that they have access to the internet and a functioning personal computer.

The Document Foundation estimates that there are 200 million active LibreOffice users worldwide, about 25% of whom are thought to be students (figures from 2018, see LibreOffice 2024). Its popularity is likely due to the fact that it not only uses open formats (e.g., `.odt` and `.ods`), but can also open and save to a range of popular formats including those used by Microsoft (e.g., `.docx` and `.xlsx`).

Quiz time!

1) Which of these is an open-source alternative to Microsoft Word?

2) Which of these is an open-source alternative to Microsoft Powerpoint?

3) Not only can software be open source, programming languages can, too. In fact, most modern programming languages are open source. In this book, we will focus on the open-source programming language `R`. Which of these is *not* an open-source programming language?

4) There are also many open-source operating systems. Which of these is an open-source alternative to the operating system Windows?

Task 1

Your first task is to **download** and **install LibreOffice** as we will use its spreadsheet editor, **LibreOffice Calc**, in the next few chapters.

- LibreOffice is available for Windows, Mac and Linux. You can download it from here: https://www.

libreoffice.org/download/download-libreoffice/.

- You will find detailed installation instructions here: https://www.libreoffice.org/get-help/install-howto/.
- Detailed documentation is also available in many different languages: https://documentation.libreoffice.org/en/english-documentation/

1.2 Open Education

The web-based version of this textbook is published as an Open Educational Resource (OER; see Figure 1.1) under the Creative Commons license: `CC BY-NC-SA`. This means that it is free to read and use, as well as edit, remix, and expand upon, provided that a) the original author and source is mentioned (as indicated by `BY`), b) any derived version is not made into a commercial product (`NC` stands for non-commercial), and c) that any derived versions of this textbook (e.g., a translated version or a version adapted for historians) are also shared with this same license (`SA` stands for share alike).

In line with the principles of Open Education, all of the datasets that we will work with in this textbook have been published in Open Access, which means that we can freely use them to learn about statistics and data visualisation using real datasets from published research studies in applied linguistics and language education.

Tips to go further

This chapter has simplified things considerably. To be considered open source, software distributions actually have to comply with ten criteria. You can read up on them here:

- https://opensource.org/osd

To find out more about the benefits of open-source soft-

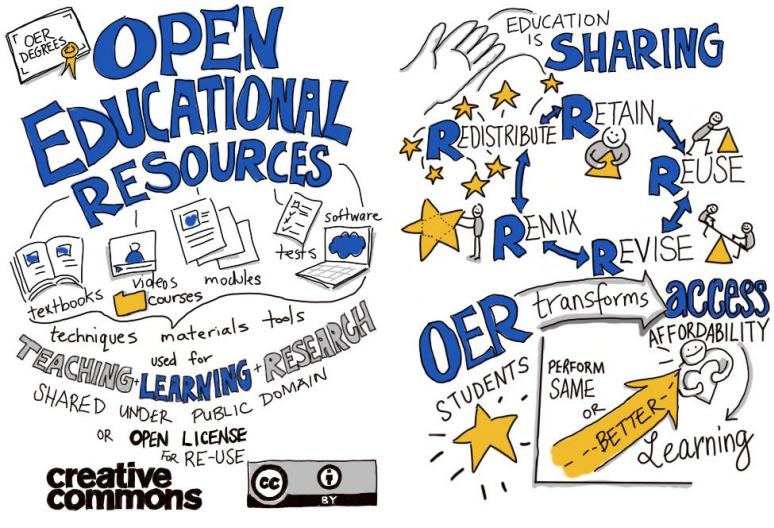


Figure 1.1: OER sketch note by Yvonne Stry

ware in the context of research, I recommend reading:

- https://book.the-turing-way.org/reproducible-research/open/open-source

To find out more about Open Educational Resources (OERs), I recommend exploring the following OER databases:

- https://oercommons.org/
- https://www.twillo.de/oer/web/

2 Installing R and RStudio

Chapter overview

This chapter is designed to help you to get started using `R` and `RStudio`, assuming no prior use of either. We will be covering the following topics:

- Advantages of learning `R`
- Downloading `R` and `RStudio`
- Setting up `RStudio`
- Using the console in `RStudio`
- Installing and loading `R` packages
- Accessing help files
- Citing packages

If you already have some experience of using `R` and `RStudio`, please ensure that both are up-to-date. Whilst parts of this chapter will likely be revision, others may be the opportunity to learn some new tips about setting up and using `R` in `RStudio`, installing and citing packages. Once you've skimmed through this chapter, feel free to swiftly move on to the next chapter.

2.1 Why learn R?

In short, because `R` can do it all! This statement is only a slight exaggeration: `R` is indeed a highly versatile programming language and environment that allows you to do a multitude of tasks relevant to the language sciences. These include data handling and processing, statistical analysis, creating effective and appealing (including interactive!) data visualisations, web scraping, text analysis, generating reports in various formats, designing web pages and interactive apps, and much, much more!

Whilst some will claim that `R` has a steep learning curve, this textbook aims to prove that the opposite is true! Whilst it's fair to say that, as with all new things, it will take you a while to get the hang of it, once you've got started, you will see that your possibilities are endless and that learning how to do new things in `R` is fun and very rewarding. This textbook introduces the {tidyverse} approach to programming in R`,` which is very accessible to beginners and we will use `RStudio` to work in `R`, which also greatly facilitates the learning process.

What's more, both `R` and the `RStudio Desktop` version that we will be using are free and open source (see Chapter 1), which means that they are accessible to all, regardless of their institutional affiliation or professional status. All you will need is access to the internet, a computer, and the intrinsic motivation to work your way through the basic skills taught in this textbook.

> "[U]sing R - it's like the green and environment-friendly gardening alternative to buying plastic wrapped tomatoes in the supermarket that have no taste anyway." (Martin Schweinberger 2022)

Figure 2.1: "Tomato Harvest, Yellow & Red" by OakleyOriginals is licensed under CC BY 2.0.

Last but not least, in choosing to learn `R`, you are entering a vibrant community of users. As an open-source programming environment, `R` is the product of many different people's contributions. Everyday, new packages, functions, and resources are being developed, improved, and shared with the community. Given that `R` has evolved into one of the most popular languages for scientific programming (and is particularly popular among linguists, see e.g., *REF*), many of these have been created by scientists and are particularly well-suited to research workflows. Moreover, the `R` community is known for being welcoming, supportive, and inclusive (in contrast to quite a few other communities in the computing world, sadly). This is reflected in the strong presence of many community-led initiatives such as RLadies and RainbowR, which encourage under-represented groups to participate in and contribute to the `R` community.

Figure 2.2: Logo of the RLadies Ribeirão Preto meet-up group, one of many RLadies chapters.

"Look, I am studying languages so why should I learn to code?"

As we will see in a future chapter, `R` code is very easy to export and share in various formats (including `.html` that can be

opened in any browser and `.pdf`). And because many other language scientists use `R` too, you will also be able to share your scripts with others, thus making your research more more accessible, transparent, and sustainable and facilitating collaborations. Being open-source, there are no restrictions as to who can run `R` code and older versions are available ensuring that exact reproduction is possible, even years later.

Learning to code in `R` is an excellent way to understand the basics of data literacy and statistical reasoning. These are skills that are highly valued among employers, both in academia and the industry. Many companies, public institutions (e.g., ministries, hospitals, and national agencies) and NGOs hire data scientists who often work in `R`. And, even if you end up doing little to no coding yourself later on, understanding the basic principles of programming is undoubtedly a very useful skill in the modern world.

Some of you may be wondering whether you should be learning `Python` rather than `R`. Both are widely used programming languages in scientific programming and data science. However, there are more resources specifically aimed at linguists and education researchers in `R` than there are in Python simply because it is currently the most widely used language in these disciplines. Should you wish to learn `Python` at a later stage, many of the same principles that you will have learned in `R` will apply: it should feel somewhat like learning Italian when you already speak Spanish or French fluently.

2.2 Installing R and RStudio

2.2.1 What are `R` and `RStudio`? And why do I need both?

As a beginner, it's easy to confuse `R` and `RStudio`, but it's important to understand that they are two very different things. `R` is a programming environment for statistical computing and graphics that uses the programming language `R`. Think of it as the engine with which we will learn to perform lots of different

tasks. `RStudio`, by contrast, is a set of tools, a so-called 'integrated development environment' (IDE). It makes working in `R` much more intuitive and efficient. If `R` is the engine of our car, you can imagine `RStudio` as our dashboard. Hence, even though we will later on appear to only be working in `RStudio`, `R` will actually be doing the heavy-lifting, under the hood.

(a) Logo of the programming language and environment `R`

(b) Logo of the IDE `RStudio` (Note that RStudio® is a trademark of Posit Software, PBC)

Figure 2.3: Even the two logos are easy to confuse, but remember that `R` and `RStudio` are two very different things!

Using other IDEs to work in R

At the time of writing, `RStudio` is the most widely used Integrated Development Environment (IDE) to work in `R`. However, it is worth noting that other IDEs that can be used to access `R`. Among others, these include:

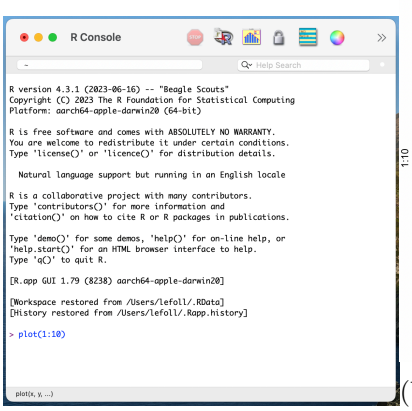
- Jupyter notebook
- Visual Studio Code
- PyCharm
- Eclipse

Whilst this textbook will assume that everyone is working in `RStudio`, if you are already familiar with another

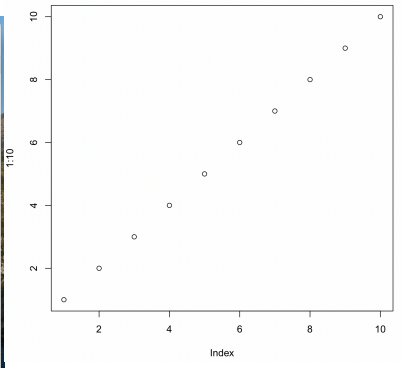
IDE that works well with `R`, you are welcome to continue working in that IDE. Each IDE has a different feel to it and offers different functions so, ultimately, it'll be up to you to find the one that suits you best!

2.2.2 How to install `R`:

1. Go to the website of the Comprehensive R Archive Network (CRAN): https://cran.r-project.org.
2. Click on the "Download R for …" link that matches your operating system (Linux, macOS or Windows), then:
    - For Windows, click on the top 'base' link, also marked as "install R for the first time" (Note that you should also use this link if you are updating your R version). On the next page, click on the top "Download R" link.
    - For MacOS, click on either the top `.pkg` link if you have an Apple silicon Mac (e.g., M1, M2, M3) or the second `.pkg` link, if you have an older Intel Mac.[^gettingstarted-1]
    - For Linux, click on your Linux distribution and then follow the instructions on the following pages.
3. Once you have downloaded one of these `R` versions, navigate to the folder where you have saved it (by default, this will be your Downloads folder), and double click on the executable file to install `R`.
4. Follow the on-screen instructions to install `R`.
5. Test that `R` is correctly installed. On Windows and MacOS, navigate to your Applications folder and double click on the `R` icon. On Linux, open up `R` by typing `R` in your terminal. This should open up an R Console. You can type R commands into the Console after the command prompt `>`. Type the following R code after the command prompt and then press enter: `plot(1:10)`.



(a) Test command in the R Console



(b) Resulting plot (note that the proportions of your plot may be different depending on the size of your window)

Figure 2.4: Testing R

If you see the plot above, you have successfully installed and tested `R` and you can go on to installing `RStudio`.

If that's not the case, make a note of the errors produced (copy and paste them into a text document or take a screenshot) and search for solutions on the Internet. It is very likely that many other people have already encountered the same problem as you and that someone from the `R` community has posted a solution online.

What to do if you cannot get R and/or RStudio working on your computer

The aim of this chapter is to install both `R` and `R Studio` on your own computer so that you can write and run your own scripts locally (i.e., on your own computer without the need for an internet connection). In some cases, however, this might not be possible. For example, because the programmes are not available for your operating system, or because you do not have admin rights on your computer, or because your disk is full and you cannot delete anything. None of these situations are ideal to do research, but don't give up on learning `R`: there is an alternative!

You can sign up to Posit Cloud. Posit Cloud will allow you to run `R` in `RStudio` in a browser (e.g., Firefox or Chrome) without having to install anything on your computer. Although Posit Cloud's free plan is limited, it will suffice to learn the contents of this textbook. You will be able to follow the textbook in exactly the same way as everyone else. However, you will need a stable internet connection and you may find that you need to be a bit more patient as things are likely to run a little slower. If you decide to opt for the Posit Cloud solution, create a free account and then go straight to Setting up RStudio.

2.2.3 How to install `RStudio`:

When you head over to their website, it may be confusing to you that the company that provides `RStudio`, Posit, also offers paid-for versions of `RStudio` and other paying services. Do not worry, we will not need any of these: These are products designed for companies and large organisations. The version of `RStudio Desktop` that we will be using, however, is completely free and, given that it is open source, even if Posit decided to stop working on this product one day, others in the `R` community would take over. Such is the beauty of open-source software!

1. Head over to this page: https://posit.co/download/rstudio-desktop/ to download the latest version of `RStudio Desktop`.
2. As you have already installed `R`, you can jump straight to step "2: Install RStudio". The website should have detected which operating system your computer is running on, so that you can most likely simply click on the "Download RStudio Desktop..." button. Your download should start straight away.
    - If an incorrect operating system is detected, simply scroll down the page to find your operating system and download the corresponding version of `RStudio`.

3. Once you have downloaded `RStudio`, navigate to the folder where the downloaded file has been saved (by default, this will be your Downloads folder), and double click on the executable file to install `RStudio`.

4. Follow the on-screen instructions to install `RStudio`.

If you run into any issues that you cannot solve with existing online posts, the Posit Community forums are a good place to ask for help.

2.3 Setting up RStudio

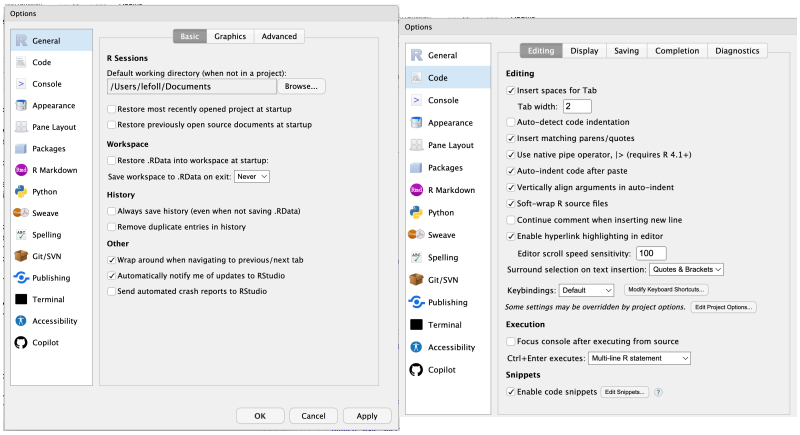
From now on, we will only be accessing `R` through `RStudio`. When you open up `RStudio` for the first time, you might find the layout rather intimidating The application window is divided into several sections, which we call 'panes'. Each pane also has several tabs. Although it may seem overwhelming at first, you will soon see that these different panes and tabs will actually make life much easier.

2.3.1 Global options

Before we get staRted properly, however, we need to change some of the default settings of `RStudio`. The first set of changes that we are going to make ensure that, each time we launch a new `R` session in `RStudio`, we are starting afresh.

To do so, head over to the 'Tools' dropdown menu and click on 'Global Options'. Make sure that the first three boxes are unticked (see Figure 2.5a). Under "Save workspace to .RData on exit", select the option "Never". Always starting afresh is good programming practice. It avoids any problems being carried over from previous `R` sessions. You can think of it like cooking in a freshly cleaned, tidy kitchen. It's much safer than preparing a meal in a messy, possibly even contaminated kitchen!

Or use the keyboard shortcut `Ctrl/Command + ,`



(a) General tab

(b) Code tab

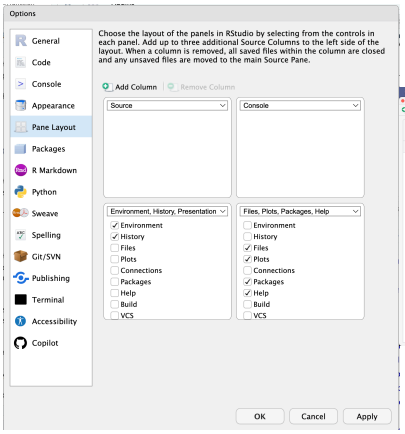
Figure 2.5: RStudio's Global Options

Next, under the 'Global Options' tab 'Panes' of the 'Global Options' window, ensure that the option "Use native pipe operator" is ticked (see Figure 2.5b). This is a new feature in `R` that is very useful so we will make use of it in this textbook.

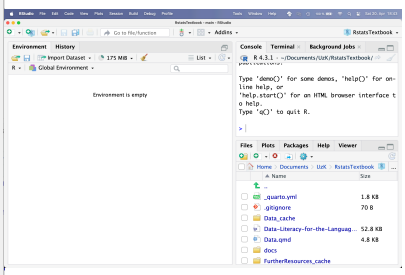
Finally, head over to the 'Pane Layout' tab. From here, you can rearrange the panes of your `RStudio` window. You can also select which tabs you would like to see in each pane. If you are already familiar with `RStudio`, feel free to stick to your favourite set-up. Personally, I use the panes layout below and, if you are new to `R`, I recommend that you select this layout, too, at least for now. You can go back to these 'Global Options' to change this setup at any stage. Don't forget to click on 'OK' to save your settings. Now, the panes in your `RStudio` window should be ordered as in Figure 2.6b.

2.3.2 Testing RStudio

It is now time we tested whether `RStudio` is communicating well with `R`. To do so, let's run the same test as in the `R Console`. This time, head over to the Console tab in the top right pane of your `RStudio` window and, after the command prompt `>`, type: `plot(1:10)` and then press enter. You should see the same



(a) Panes Layout tab



(b) Customised panes layout

Figure 2.6: Recommended `RStudio` panes layout

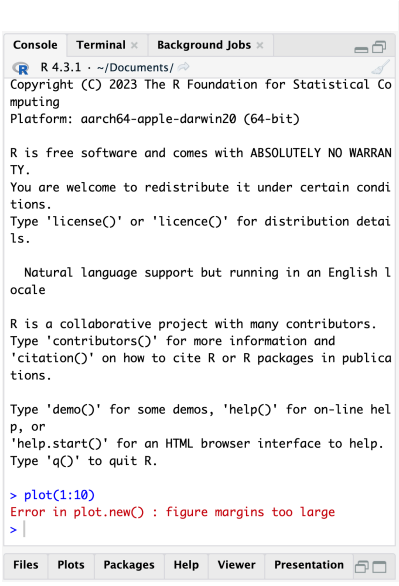
plot as earlier on (see Figure 2.4b), appearing in the Plots tab of the bottom right pane of your `RStudio` window.

If you get the following error message `Error in plot.new() : figure margins too large`, this is because your bottom right pane is hidden from view or too small for the plot to be printed there. Click on the small two-window icon in the bottom right corner to unhide the bottom right pane, if it is hidden (see Figure 2.7a). Or, if it too small, click on the dividing line between the two right-hand side panes and, whilst still holding down the mouse button, drag up the line until it is about halfway up. Then, re-type the command `plot(1:10)` in the Console pane and press enter again. The plot should appear as in Figure 2.7b.

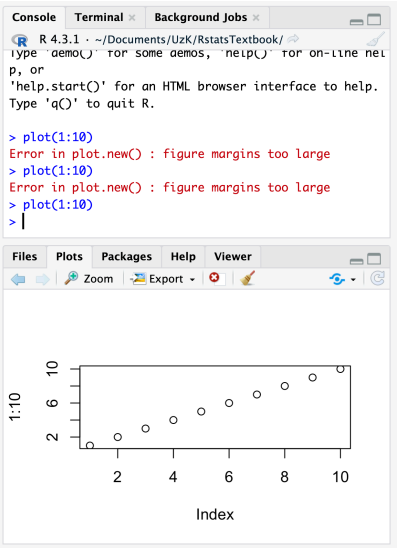
2.4 Installing `R` packages

2.4.1 What are packages?

You now have a base installation of `R`. Base `R` is very powerful and comes with many standard packages and functions that `R` users use on a daily basis. If you click on the Packages tab in the bottom right pane and scroll down, you will see that there



(a) Hidden (minimised) bottom right pane



(b) Now the dividing line between the two panes is halfway up and the plot has been successfully output in the Plots pane

Figure 2.7: Testing that `RStudio` is communicating well with your `R` installation.

are many packages available. Only a few are selected. These are part of the base R installation.

At the same time, thousands of `R` users have developed and shared additional `R` packages that enable us to vastly increase the capacities of base `R`. Packages are a very helpful way to bundle together a set of functions, data, and documentation files so that other `R` users can easily download these bundles and add them to their local `R` installation.

Throughout this textbook, the names of packages will be enclosed in curly brackets, e.g.: {ggplot2}.

Quiz time!

1) Which of these packages is not part of base `R`?

2) Is it possible to create an `R` package that provides access to the full texts of Jane Austen's six completed, published novels for computational text analysis in `R`?

3) Is the {janeaustenr} package installed as part of base `R`?

2.4.1 Installing packages

To install a package, you will first need to download it from the internet. Packages are typically stored on different websites (online repositories), but the most trustworthy one and easiest to work with is CRAN (Comprehensive R Archive Network). To install the {janeaustenr} package from CRAN, simply type the following command in the Console pane and then type enter: `install.packages("janeaustenr")`.

This command will take a few seconds to run (or longer depending on how slow your internet connection is). You should then

see a message in red in the console indicating that the package has been downloaded and its size (here: 1.5 megabyte), as well as the path to where the package's content has been saved on your computer (see Figure 2.8). You do not need to worry about any of this.

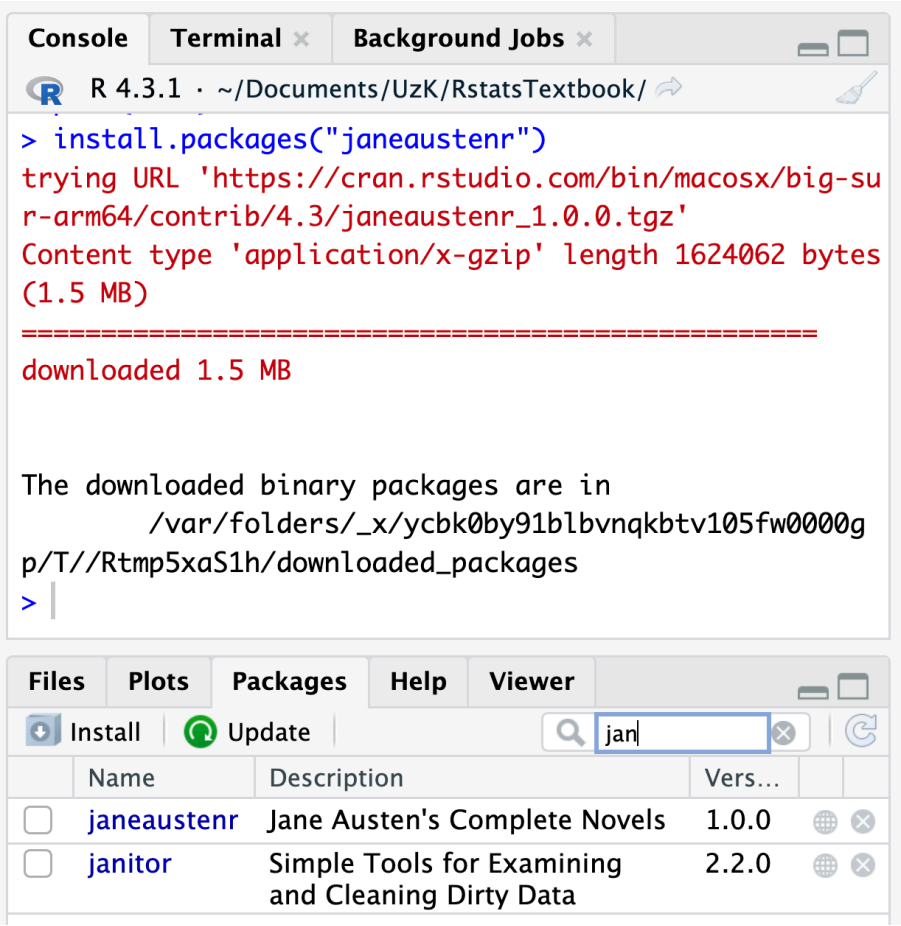


Figure 2.8: Screenshot showing that the package has been correctly installed.

To check that the package has been successfully downloaded and installed, head over to the Packages tab of the bottom-right pane and scroll down to the {janeaustenr} package, or search for it using the search window within this same tab. The {janeaustenr} package should now be visible, which tells us that the package is installed on your computer. Note, however, that

the checkbox next to it is currently empty. This means that the package hasn't been loaded in our current R session and therefore cannot be used yet. Note that whilst you only need to install each package once, you will need to load it every time we want to use it in a new R session. This is because, when we start a new R session, the kitchen is perfectly clean and tidy and everything is back in storage. And the good news is that we don't even need to do the washing-up!

2.4.2 Loading packages

You can think of base R as a fully functional student kitchen. It is rather small and only has the most essential ingredients and equipment, but it still has everything you need to cook simple, delicious meals. Downloading and installing additional packages is like buying fancier ingredients (these are packages that include datasets) or more sophisticated and specialised kitchen devices (these are packages that include additional functions).

Once you have downloaded and installed a new package, it is put in storage (either in the fridge or in a kitchen cupboard). In this case, the package appears in your Packages tab, but is not yet selected. If you want to use the new ingredient or the piece of equipment that was delivered in the new package, you need to get it out of the fridge or the cupboard and place it on the kitchen counter. This is the equivalent to loading a package. Once they are unpacked (i.e., installed), packages are referred to as libraries.

So the first thing we now need to do to be able to analyse Jane Austen's novels in R is to load the {janeaustenr} library. The command to do is `library(janeaustenr)`. Type this command in the Console and press enter. Ideally, it should look like nothing has happened. Indeed, if you do not get an error message (in red), then the library has been successfully loaded. We can check that this is actually the case by looking again for the {janeaustenr} in the Packages tab. The checkbox should now be ticked.

2.5 Package documentation

To find out more about any package or function, simply use the command `help()` or its shortcut `?()`. For example, to find out more about the {janeaustenr} package, enter the command `help(janeaustenr)` or `?(janeaustenr)` in the Console. The help file will open up in the Help tab of the bottom-right pane. It contains the name of the package and a short description, as well as the name of the package maintainer, Julia Silge, and some additional links.

One of these links takes us to the package creator's GitHub repository. This is where we can find a source code for the package, should we want to check how it works under the hood, or amend it in any ways. Click on this link and scroll down the package's GitHub page to read its README file. This document informs us that the package includes plain text versions of Jane Austen's six novels and tells us under what name they are stored within the library. For example, to access *Pride and Prejudice,* we need to load the library object `pridepredjudice`.

Note that the names of objects in `R` cannot contain spaces or hyphens.

Pick your favourite Jane Austen novel and enter its corresponding object name in the Console, e.g., `emma`. The entire novel will be printed in the Console output! You can print only a few lines by selecting them within square brackets, e.g., the command `emma[20:25]` will only print lines 20 to 25 of the text object `emma` (see Figure 2.9).

To find out more about a dataset or function within a package, use the functions `help()` or `?()`, e.g., `help(emma)` or `?emma`. In this case, the help file provides us with a short description of this object and a link to the original source from which the package creator obtained the novel (which is in the public domain, otherwise it would not be possible to share it in this way).

2.6 Citing packages

When we use a package that is not part of base `R`, it is very important to reference the package adequately. There are two

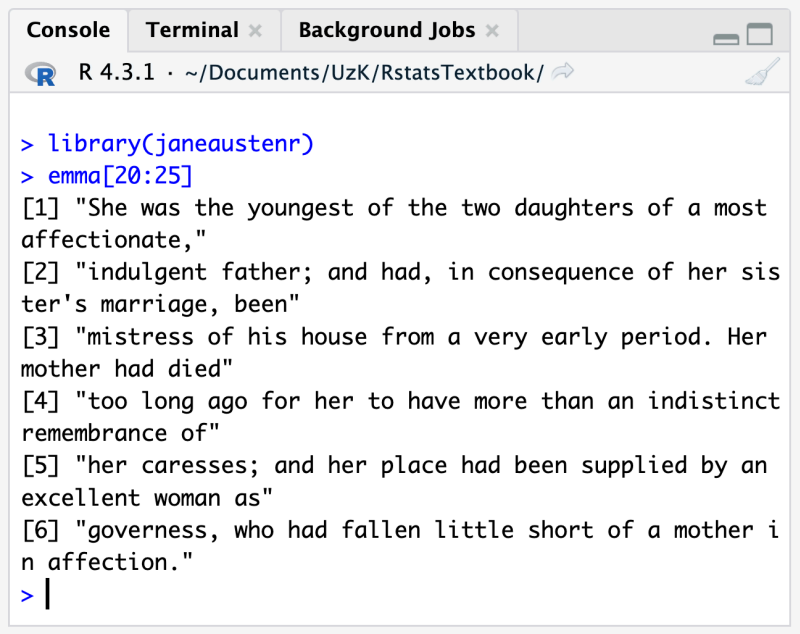


Figure 2.9: Screenshot showing a selection of lines from the text object `emma` (note that you can adjust the size of the Console pane to see more or less of the text at any one time).

main reasons for doing this. For a start, the people who create and maintain these packages largely do so in their free time and they deserve full credit for their incredibly valuable work and contribution to science. Hence, whenever you use a package for your research, you should cite it, just like you would other sources.

The help page of the {janeaustenr} package already informed us that the maintainer of the package is Julia Silge. To get a full citation, however, we should use the `citation()` function. Enter `citation("janeaustenr")` in the Console to find out how to cite this package.

Note that the recommended bibliographic reference also includes the package version, which is important for reproducibility as the package may evolve and someone wanted to reproduce your analysis will need to know which version you used. This is the second main reason why we should cite the packages that we use properly. In a research report, thesis, or academic article, you would cite the {janeaustenr} package like this:

> We used the janeaustenr package (Silge 2022) to access Jane Austen's six, published novels in R (R Core Team 2024).

You can see the full references by hovering on the in-text citation links or by going to the References section of this book.

References

2024. LibreOffice. *Wikipedia*. https://en.wikipedia.org/w/index.php?title=LibreOffice&oldid=1218520104.

Parsons, Sam, Flávio Azevedo, Mahmoud M. Elsherif, Samuel Guay, Owen N. Shahim, Gisela H. Govaart, Emma Norris, et al. 2022. A community-sourced glossary of open scholarship terms. *Nature Human Behaviour*. Nature 6(3). 312–318. https://doi.org/10.1038/s41562-021-01269-4.

Silge, Julia. 2022. *Janeaustenr: Jane austen's complete novels*. https://CRAN.R-project.org/package=janeaustenr.

Team, R Core. 2024. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/.

A Next-step resources

In the hope that this textbook has inspired you to dive deeper into the wonderful world of quantitative data analysis, statistics, data visualisation, and coding in R, here is a (work-in-progress) curated list of further resources to continue your learning journey!

A.1 Recommended resources specific to the language sciences

- Brezina, Vaclav. 2018. Statistics in Corpus Linguistics: A Practical Guide. Cambridge: Cambridge University Press. https://doi.org/10.1017/9781316410899.
- Desagulier, Guillaume. 2017. Corpus Linguistics and Statistics with R: Introduction to Quantitative Methods in Linguistics (Quantitative Methods in the Humanities and Social Sciences). Cham: Springer International Publishing.
- Gries, Stefan Thomas. 2013. Statistics for linguistics with R: a practical introduction. 2nd revised edition. Berlin: De Gruyter Mouton.
- LADAL contributors. Tutorials of the Language Technology and Data Analysis Laboratory. https://ladal.edu.au/tutorials.html `Open Educational Resource`.
- Levshina, Natalia. 2015. How to do linguistics with R: Data exploration and statistical analysis. Amsterdam: John Benjamins.
- Schneider, Dr Gerold & Max Lauber. 2020. Statistics for Linguists. https://dlf.uzh.ch/openbooks/statisticsforlinguists/ `Open Educational Resource`.

- Winter, Bodo. 2019. Statistics for Linguists: An Introduction Using R. New York: Routledge. https://doi.org/10.4324/9781315165547.

A.2 Further Open Educational Resources (in no particular order)

- Diez, David, Mine Cetinkaya-Rundel, Christopher Barr & OpenIntro. 2015. OpenIntro Statistics. Leanpub. https://leanpub.next/os.
- Guide to Effect Sizes and Confidence Intervals: https://matthewbjane.quarto.pub/guide-to-effect-sizes-and-confidence-intervals/
- Happy Git and GitHub for the useR: https://happygitwithr.com/
- Quarto & reproducibility: https://ucsbcarpentry.github.io/Reproducible-Publications-with-RStudio-Quarto/index.html
- Modern Data Visualization with R: https://rkabacoff.github.io/datavis
- Building reproducible analytical pipelines with R: https://raps-with-r.dev/
- Modern Plain Text Computing: https://mptc.io/content/01-content.html
- https://www.data-to-viz.com/
- Interpreting data visualisation: https://pressbooks.library.torontomu.ca/criticaldataliteracy/
- Improve your statistical inferences: https://lakens.github.io/statistical_inferences/
- What they forgot to teach you about R: https://rstats.wtf/
- Introduction to Data Science: https://florian-huber.github.io/data_science_course/book/cover.html
- Data Science in Education Using R: https://datascienceineducation.com/
- Models Demystified: A Practical Guide from t-tests to Deep Learning https://m-clark.github.io/book-of-models/

- Data Visualization in R https://datavizf23.classes.andrewheiss.com/
- R for Data Science https://r4ds.hadley.nz/intro

A.2.1