

Chapter 1 HW 1. Intro to R, RStudio, and RMarkdown

Adapted from [Nordman et al. \(2020\)](#).

1.1 Learning outcomes

By the end of this HW you should be able to:

- Open RStudio and run R code
- Create a new R Markdown notebook with code chunks
- Create objects by writing and running code in the console and in R Markdown

If you're completely new to programming, you're unlikely to understand everything in these readings the first time you work through it - and that's ok! Some of the beginning information we provide has a bit of a chicken-and-egg problem in that we need to give you it before you begin but until you've had some experience with R, it won't fully make sense. Take what you can from the chapters on the first attempt but also come back to them in a few weeks time and have another read through - you'll be surprised at how much you've learned in a short space of time.

1.2 Activity 1: Install R and RStudio

1.2.1 R and RStudio

R is a programming language that you will write code in. RStudio is an Integrated Development Environment (IDE). Think of it as knowing English and using a plain text editor like NotePad to write a book versus using a word processor like Microsoft Word. You could do it, but it would be much harder without things like spell-checking and formatting and you wouldn't be able to use

some of the advanced features that Word has developed. In a similar way, you can use R without R Studio but we wouldn't recommend it. RStudio serves as a text editor, file manager, spreadsheet viewer, and more.

- There's an [RStudio IDE Cheatsheet](#) you will find helpful to download and keep as a reference.

Your first task is to install R and RStudio, or to open RStudio on a university computer. The installation is fairly straightforward, but if you would like a full walkthrough, here are ones for [Windows](#) and [Macs](#). The basic steps are below. Note that you cannot install R on a Chromebook or a tablet, only a full laptop or desktop computer.

To install R, first go to <https://www.r-project.org> and choose your system. Click the download R link in the middle of the page under "Getting Started." Download and install the installer files (executable, pkg, etc) that correspond to your system.

To install RStudio, please go to <https://www.rstudio.com/products/rstudio/download/#download> and choose your system. The installer is preferred. If you have RStudio installed but not the latest version, just download the latest installer and install.

1.2.2 If you don't have a laptop...

R and RStudio are installed on every computer in the Psychology cluster, as well as on most other CMU computer clusters. You can also access R and RStudio through CMU's Virtual Andrew service.

While you can access Virtual Andrew through a browser and therefore it is technically possible to use it on a phone or tablet, we strongly discourage you from doing so. At minimum you will need a keyboard but in addition, all the instructions in this workbook will assume you are using a desktop computer or laptop and so you may find it more difficult to follow, particularly when it comes to file management.

1.3 Activity 2: Open RStudio

As noted, you have installed two bits of software, R and RStudio. **You only ever need to open RStudio.** If during the installation process you created a shortcut to R, delete the shortcut, it will only cause you trouble.

- Open RStudio
- You may also find it useful to add a shortcut to your desktop or task bar so that it is easy to find.

1.3.1 RStudio panes

RStudio is arranged with four window panes. By default, the upper left pane is the **source pane**, where you view, write, and edit code from files and view data tables in a spreadsheet format.

When you first open RStudio this pane won't display until you open a document or load in some data – don't worry, we'll get to that soon.



The RStudio IDE.

The RStudio IDE.

The lower left pane is the **console pane**, where you can type in commands and view output messages. **When you first open RStudio, the console pane will take up the entire left hand side.** You can write code in the console to test it out. The code will run and can create objects in the environment, but the code itself won't be saved. You need to write your code into a script in the source pane to save it, which we'll cover later in this chapter.

Try using the console as a calculator - type `1 + 1` and press enter.

The right panes have several different tabs that show you information about your code. The most used tab in the upper right pane is the **Environment** tab. The Environment tab lists some information about the objects that you have defined in your code.

In the lower right pane, the most used tabs are the **Files** tab for directory structure, the **Plots** tab for plots, the **Packages** tab for managing add-on packages (we'll explain what these are in the next chapter), and the **Viewer** tab to display reports created by your scripts. You can change the location of panes and what tabs are shown under `Tools > Global Options... > Pane Layout` .

1.4 Activity 3: Settings

Before we start writing code, we're going to change a few settings to make things easier for us in the long-run.

1.4.1 Themes and accessibility

You can customise how RStudio looks to make it work for you.

- Click `Tools > Global Options > Appearance` . You can change the default font, font size, and general appearance of R Studio by applying a theme, including using dark mode.

Play around with the settings and see what you prefer - you're going to spend a lot of time with R, it might as well look how you like it!

1.4.2 Settings for reproducibility

One of the reasons you're going to learn R is so that any quantitative data analysis and research you do is **reproducible**. If your work is reproducible, it means that you (or someone else) can take your data and get the same result you got. This is easier to do with code because you have a written record of everything you did, compared to using point-and-click software where there is no record and you just have to remember what you did.

To help ensure your work is reproducible, there are a few settings you should change in RStudio. If you keep stuff around in your workspace, things can get messy and you can accidentally end up using the wrong dataset or object. You should always start with a clear workspace. This also means that you never want to save your workspace when you exit - the only thing you want to save are your scripts.

- Go to `Tools > Global Options`
- In the General tab, uncheck the box that says `Restore .RData into workspace at startup` .
- Additionally, set the "Save workspace to .RData on exit" to `Never` .

 RStudio settings.

RStudio settings.

1.4.3 Sessions and projects

A “session” refers to a single period of use with R. During this session, you can load data, write code, and create visualizations. All of the objects that you create and data that you import into R exist within the environment of this session.

For example, if you create a variable in R, that variable is stored in memory for the duration of the session. If you end the session and then start a new one, that variable will no longer exist in your R environment unless you’ve saved your workspace or the object to a file.

A “project” is a specific working directory (or folder) designated to contain all the files relevant to a particular set of analyses or a specific research study. Projects are a way of organizing your work in RStudio that makes it easier to keep track of all the different files associated with a particular line of work.

We’re going to use projects to help organise your data skills work for Research Methods.

1.5 Activity 4: Create a new project

First, if you don’t have one already, make a folder on your computer named “Research-Methods” and then within this folder, make another folder named “Data-Skills-HW”.

Then do the following, in this order, in RStudio:

- Click “File” then click “New project”
- Then, click on the first option in the list “New Directory”
- Then, click “New Project”
- Then you are given the opportunity to name your project and select which folder it should be stored in. First in the “Directory name” box, type “Intro to R”.
- Then, click “Browse” and find the “R-Intro-HW” folder in the “Research-Methods” folder you created and click “Open”.
- Finally, click “Create project”.

1.6 Activity 5: Writing code

Ok, let's write some code! We're going to open a script soon but first let's show you how to write code and create objects in the console.

1.6.1 Objects

A large part of your coding will involve creating and manipulating objects. Objects contain stuff. That stuff can be numbers, words, or the result of operations and analyses. You assign content to an object using `<-`

Run the following code in the console, but change the values of `name`, `age` and `birthday` to your own details (`birthday` should be the date of your next birthday, year-month-day). Press enter after you type each line.

```
name <- "Abby"
age <- 40
today <- Sys.Date()
birthday <- as.Date("2025-07-11")
```

You'll see that four objects now appear in the environment pane:

- `name` is **character** (text) data. In order for R to recognise it as text, it **must** be enclosed in double quotation marks `" "`.
- `age` is **numeric** data. In order for R to recognise this as a number, it **must not** be enclosed in quotation marks. Every year I have to update my age in this book and I am confronted with my own mortality drawing ever closer.
- `today` stores the result of the **function** `Sys.Date()`. This function returns your computer system's date. Unlike `name` and `age`, which are hard-coded (i.e., they will always return the values you enter), the contents of the object `today` will change dynamically with the date. That is, if you run that function tomorrow, it will update the date to tomorrow's date.
- `birthday` is also a date but it's hard-coded as a specific date. It's wrapped within the `as.Date()` function that tells R to interpret the character string you provide as a date rather than text.

To print the contents of an object, type the object's name in the console and press enter. Try printing all four objects now.

1.6.2 Removing objects

To clear objects from the environment you can remove them individually:

```
rm(name)
```

Or you can remove them all at once by clicking on the brush icon at the top of the environment pane.



Removing objects.

Removing objects.

1.7 Activity 6: R Markdown Notebooks

The problem with writing code in the console is that it doesn't save it anywhere, and the whole point of writing code is so that you can save it and use it again. We're going to write our code in a document format called **R Markdown** (abbreviated as Rmd).

R Markdown documents are a great way to create dynamic documents with embedded chunks of code and we'll use R Markdown to do all your data skills work for this class.

For more information about R Markdown feel free to have a look at their main webpage <http://rmarkdown.rstudio.com>. The key advantage of R Markdown is that it allows you to write code into a document, along with regular text, and then **knit** it using the package `knitr` to create your document as either a webpage (HTML), a PDF, or Word document (.docx).

1.7.1 Create a new R Markdown Notebook

To open a new R Markdown document click the 'New File' icon (it's a white document with a green plus sign just under the "File" menu) and then click 'R Notebook'. You'll see a new file appear in the source pane.

Once you've opened a new document be sure to save it by clicking `File -> Save`. You should also name this file "HW-1_intro-to-R" (or a meaningful name that you prefer). If you've set the working directory correctly, you should now see this file appear in your file viewer pane.

One of the great things about using projects in RStudio is that your files will automatically save into your project working directory by default. This makes it easy to keep everything organised so make sure you have followed the steps to set up your project carefully!

1.7.2 Creating code chunks

When you first open a new R Notebook you will see a bunch of welcome text and code. This will appear every time you open a new R Markdown document.

Do the following steps in this order:

- Delete **everything** from line 6 onwards
- On line 6, type "# About me" (the # indicates that this is a top-level heading) and press enter to move the cursor down to line 9
- Click the code chunk button at the top of your Markdown document (it's a green square with a C and a plus sign) then click `R` to insert a new R code chunk.

What you have created is a **code chunk**. In R Markdown, anything written in the white space is regarded as normal text, and anything written in a grey code chunk is assumed to be code. This makes it easy to combine both text and code in one document.

When you create a new code chunk you should notice that the grey box starts and ends with three back ticks ```. One common mistake is to accidentally delete these back ticks. Remember, code chunks are grey and text entry is white - if the colour of certain parts of your Markdown doesn't look right, check that you haven't deleted the back ticks.

1.7.3 Adding code

Now we're going to add in the code we used earlier to our R Markdown document so that we can save it.

- In your code chunk write the below code (but again, replace the values of name/age/next birthday with your own details).

- Remember that text values and dates need to be contained in quotation marks but numerical values do not. Missing and/or unnecessary quotation marks are a common cause of code not working - remember this!

```
name <- "Abby"
age <- 40
today <- Sys.Date()
birthday <- as.Date("2025-07-11")
```

1.7.4 Running code

When you're working in an R Markdown document, there are several ways to run your lines of code.

The best way to run a single line of code is to use a keyboard shortcut. Make sure that the cursor is in the line of code you want to run (it can be anywhere) and press `Ctrl + enter` or `Cmd + enter` if you're on a Mac. This will run the line of code and move your cursor to the next line.

- Run each line of code to create the objects using the above method.
- Then, clear all the objects from the environment either using `rm()` on each object or click the brush icon to get rid of them all at once.

Now, let's try running all the code in the chunk at once rather than one line at a time. You have two options:

- Put your cursor somewhere in the code chunk and press `Ctrl + shift + enter`
- Or, press the green "play" button at the top-right of the code chunk and this will run **all** lines of code in that chunk.

Whilst you're working through these assignments, use the white space to add notes. For example, if you figure something out or you manage to fix an error, make a note of it in your Markdown document.

1.8 Activity 7: Inline code

An incredibly useful feature of R Markdown is that R can insert values into your writing using **inline code**. If you've ever had to copy and paste a value or text from one file to another, you'll know how easy it can be to make mistakes. Inline code avoids this. It's easier to show you what inline code does rather than to explain it so let's have a go.

- First, copy and paste this text exactly (do not change anything) to the **white space** underneath your code chunk.

```
My name is `r name` and I am `r age` years old. It is `r birthday` - today` days unt
```

- Now, we're going to "knit" our file. This simply means that we're going to compile our code into a document that is more presentable. To do this click `Preview`. R Markdown will create a new HTML document and it will automatically save this file in your project working directory.

As if by magic, that slightly odd bit of text you copied and pasted now appears as a normal sentence with the values pulled in from the objects you created.

My name is Abby and I am 40 years old. It is 151 days until my birthday.

We're not going to use this function too often in the rest of the course but hopefully you can see just how useful this would be when writing up a report with lots of numbers. R Markdown is an incredibly powerful and flexible format - these chapters were written using it!

There are a few final things to note about knitting that will be useful for the homework:

- R Markdown will only knit if your code works, otherwise it will throw an error. Knitting your file is therefore a good way of checking for the assignments whether you've written legal code!
- The order of what's in your Markdown document matters. When the file knits, it starts at the beginning. This means that if you want to use any values in inline code, the code that creates these values must come before the inline code that calls upon it.

- You can choose to knit to a Word or PDF document rather than HTML. This can be useful for e.g., sharing the document with others, however, it may lose some functionality and it probably won't look as good so we'd recommend always knitting to HTML.
- R will automatically open the knitted file in the viewer, however, you can also navigate to the Files tab and open the HTML file in your web browser (e.g., Chrome or Firefox).

1.9 Activity 8: Saving and quitting the session

We're done with all the coding activities for this chapter and whilst it might be tempting to just ~~run away from R~~ close your laptop, ensuring you save and quit the session properly can save you a lot of time and problems in the long-run.

- First, make sure your R Markdown is saved. If there are any unsaved changes then the save icon will be in blue, if it's greyed out it means there are no unsaved changes. But just to be safe, always hit `Ctrl + s` or click `File - Save` which will save your file.
- Then, exit RStudio completely by clicking "File - Quit session" or using the shortcut "`Ctrl + Q`" (Windows) or "`Cmd + Q`" (Mac).

1.10 Finished!

And you're done! You've not only written functioning code but you've written a reproducible output! You could send someone else your R Markdown document and they would be able to produce exactly the same HTML document as you, just by pressing knit.

The key thing we want you to take away from this HW is that R isn't scary. It might still be very new, but we're going to take you through it step-by-step. You'll be amazed at how quickly you can start producing professional-looking data analysis and visualizations.

1.10.1 Check for completeness

For this HW, upload your R notebook (.Rmd file), and your Preview html file (.nb.html). Both should include:

- An "About Me" heading

- A code chunk setting the four variables described above
- A sentence including those values as inline code
- Any notes for yourself about things you figured out along the way

All code in your R Notebook should run successfully.