

Introduction to Rstudio

Anders K. Krabberød
a.k.krabberod@ibv.uio.no

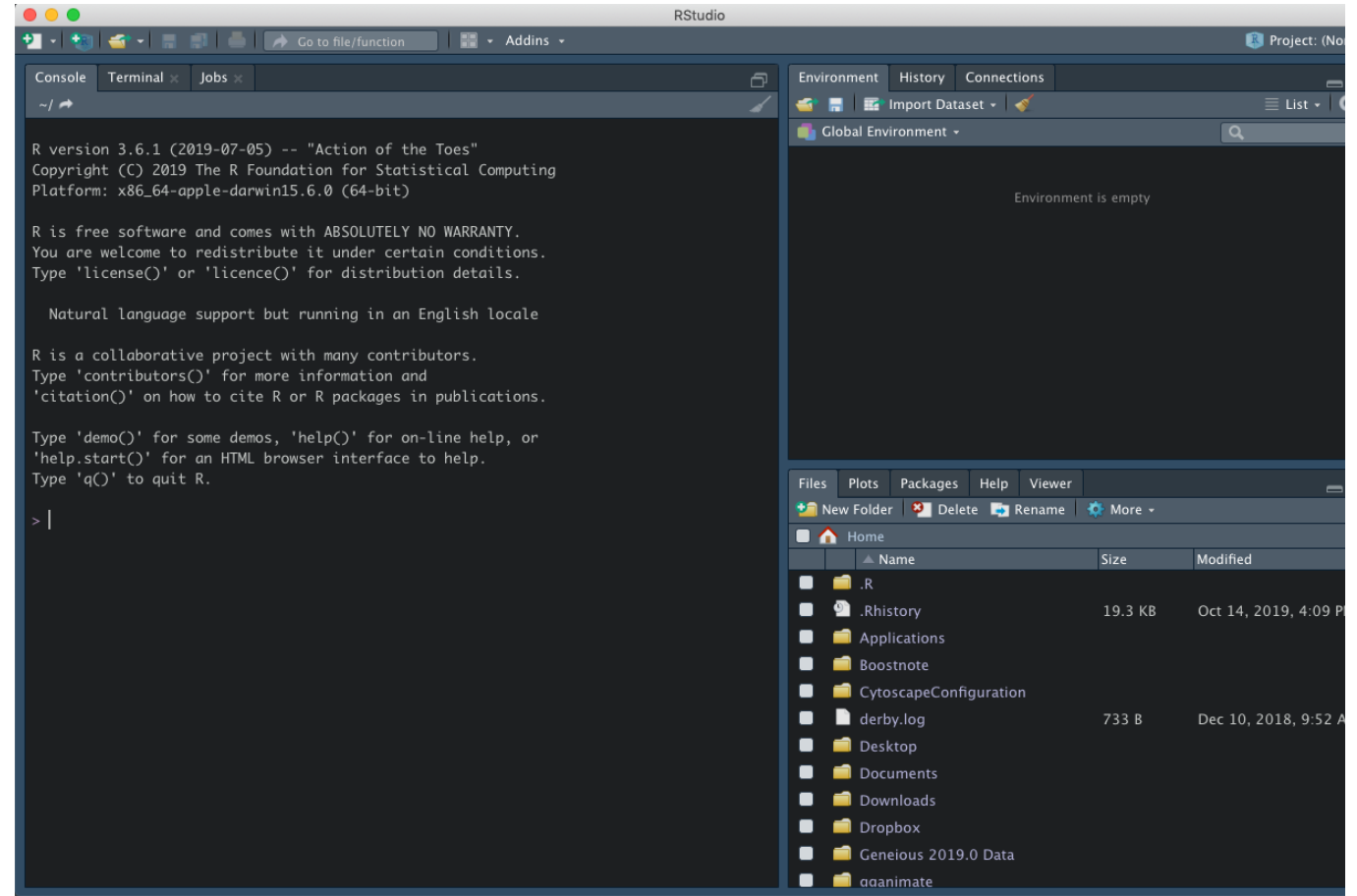
(remember GitHub)



- Github for the workshop with lectures, scripts, and examples:
- https://github.com/krabberod/OMG_bioinformatics_sessions_2022

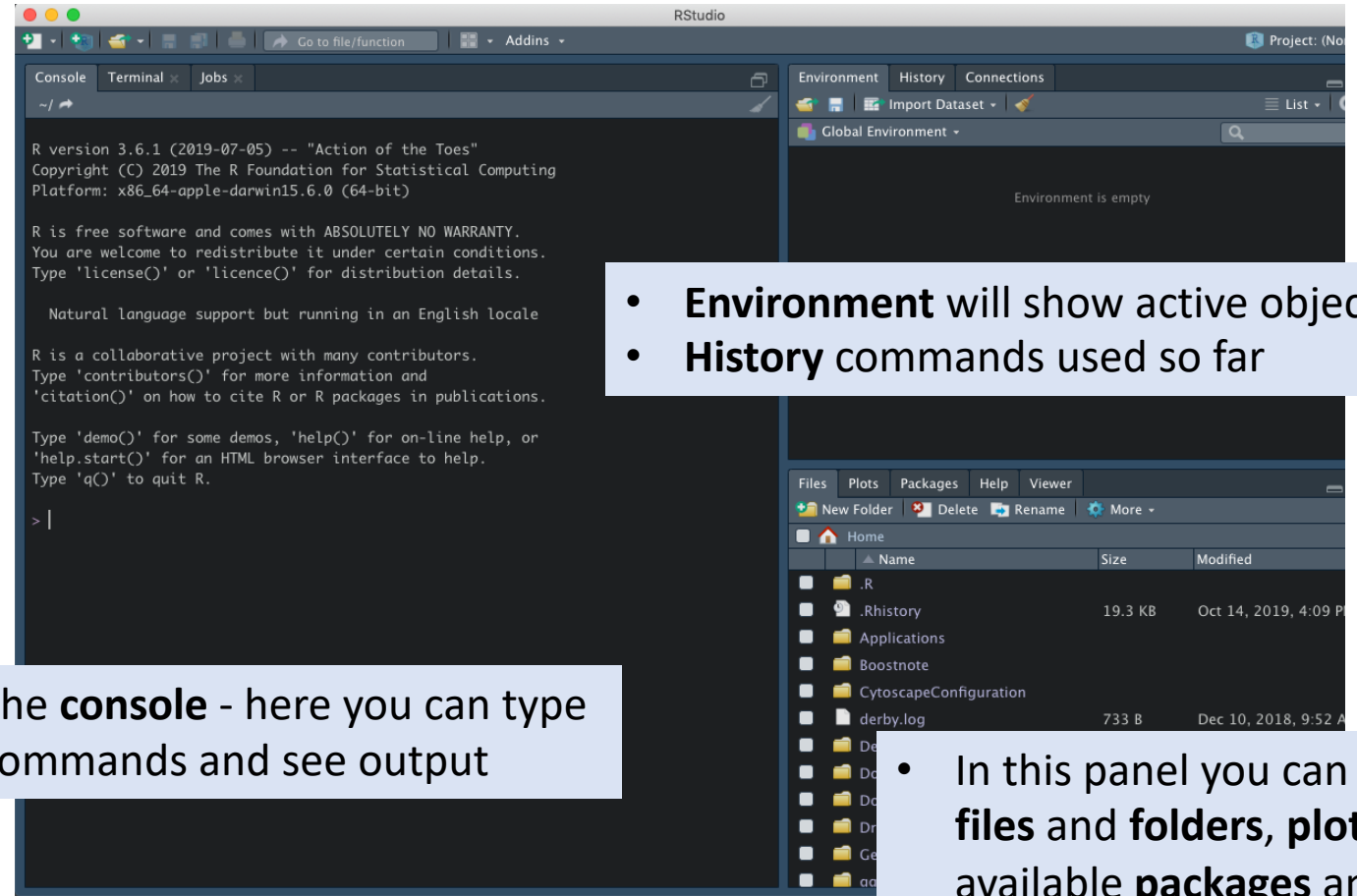
Rstudio

- A graphical user interface for running R (and other programming languages like Python, bash and more).
- Also has a file browser and a window with easy access to objects and variables.



Rstudio

- A graphical user interface for running R (and other programming languages like Python, bash and more).
- Also has a file browser and a window with easy access to objects and variables.

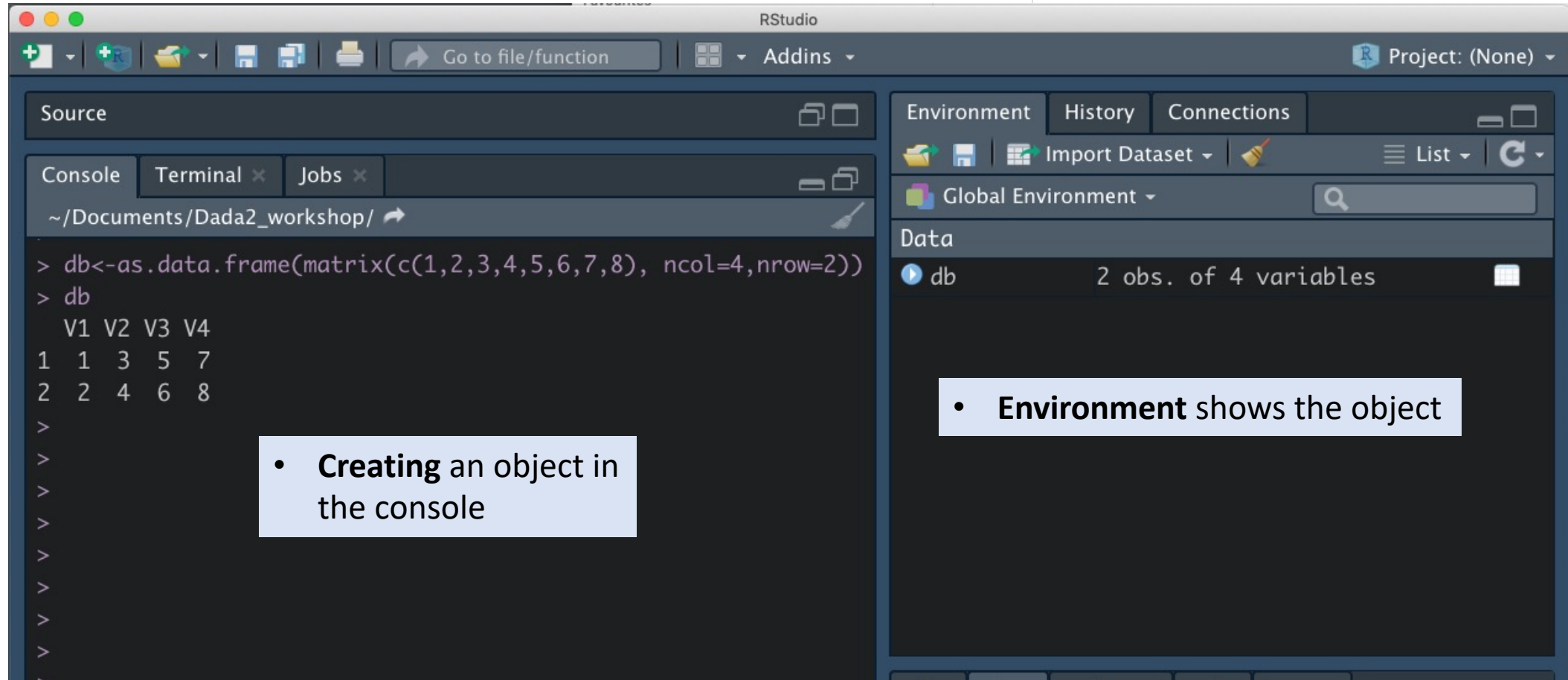


The **console** - here you can type commands and see output

- **Environment** will show active objects
- **History** commands used so far

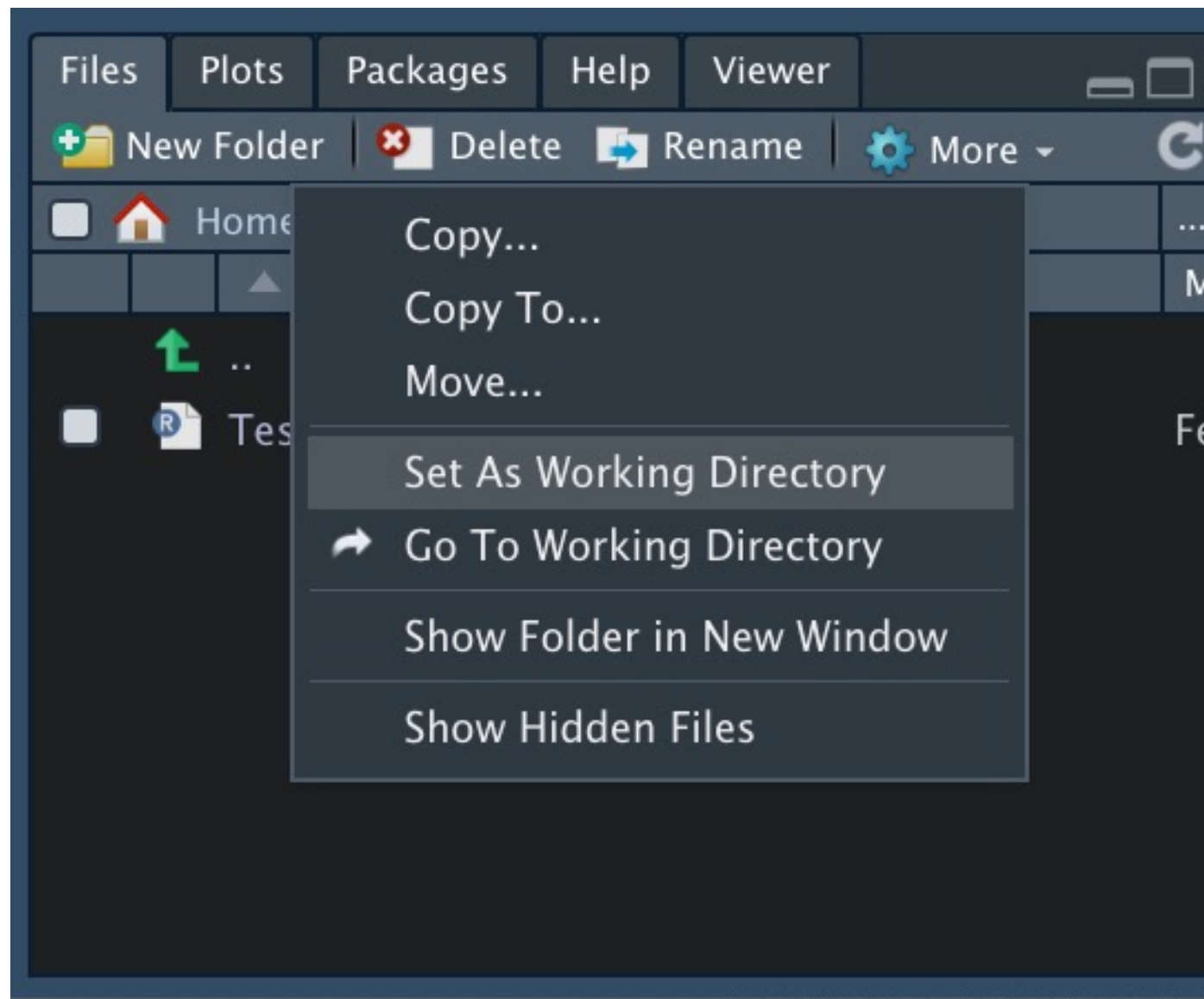
- In this panel you can find **files** and **folders**, **plots**, available **packages** and the **help screen**

Rstudio



Setting working directory

- Setting the working directory will let you determine the folder where *Rstudio* will write output, print graphics and look for files to open.
- Navigate to correct folder under the “files” tab
- Click “Set As Working Directory” (under *More*)



Setting working directory

- Alternatively write

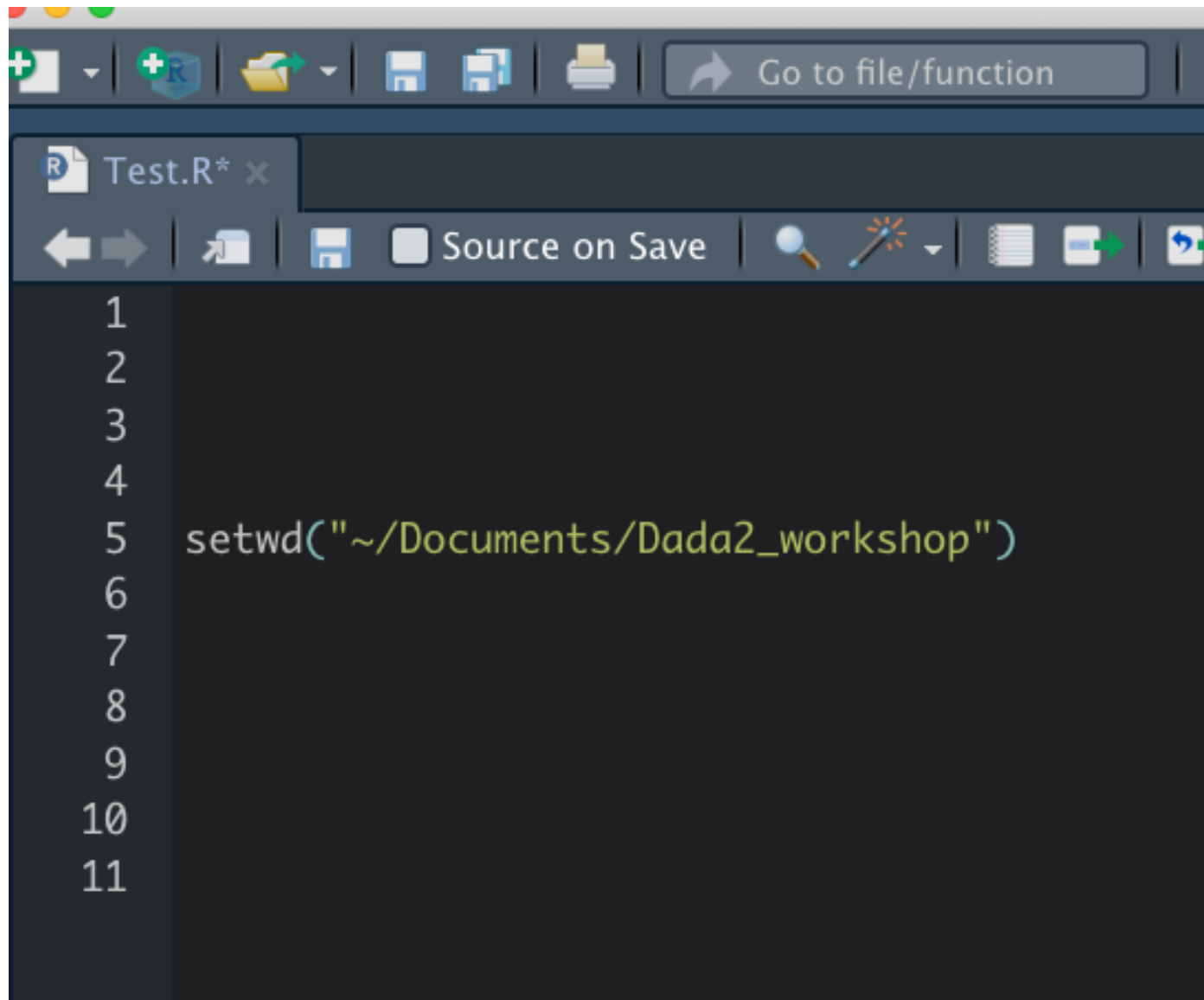
MAC:

```
setwd("~/path/to/my/folder")
```

WINDOWS

```
setwd("C:/path/to/my/folder")
```

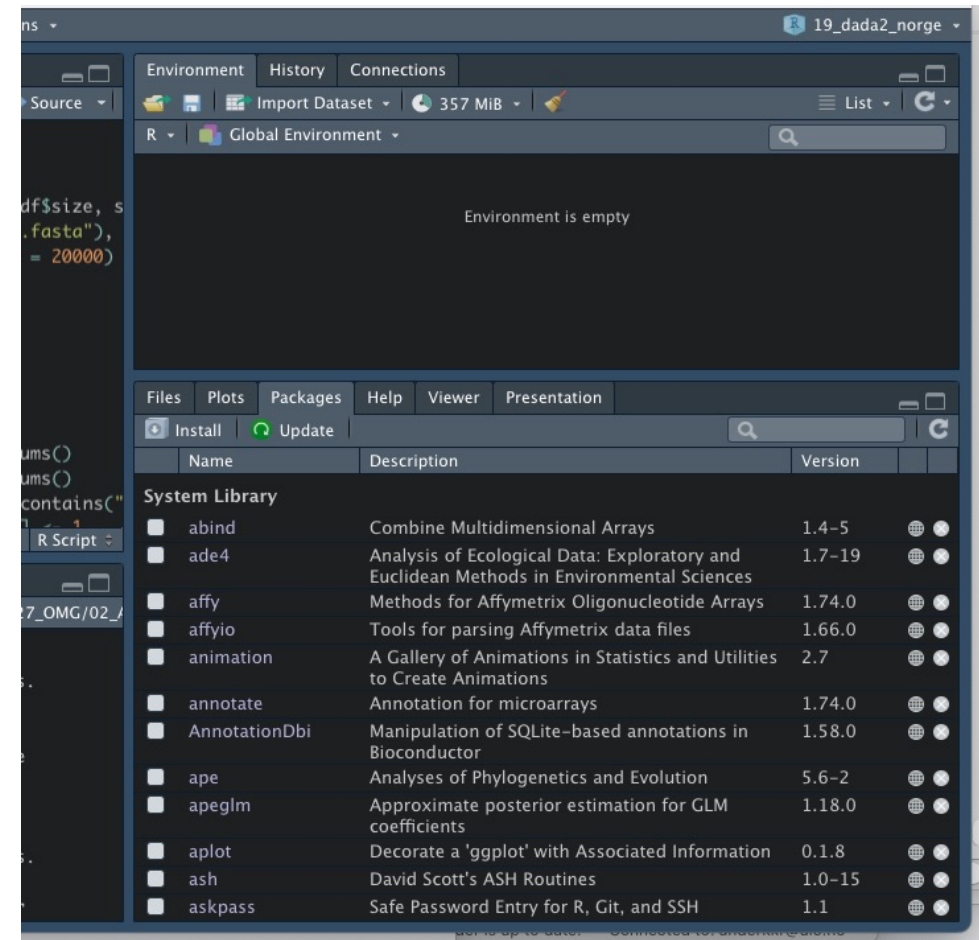
- This way you can include the path to the working directory as part of your script.

A screenshot of the RStudio IDE interface. The top toolbar includes icons for file operations and a search bar labeled "Go to file/function". Below the toolbar, a tab labeled "Test.R*" is active. The editor pane shows a script with line numbers 1 through 11 on the left. Line 5 contains the command `setwd("~/Documents/Dada2_workshop")` in a light green font. The rest of the script is empty.

```
1  
2  
3  
4  
5 setwd("~/Documents/Dada2_workshop")  
6  
7  
8  
9  
10  
11
```

Installing packages

- Packages are a set of functions, compiled code, and sample data.
- When you start R only the default packages are loaded
- Other packages need to be installed and called explicitly to be utilized
- Packages can be found in three main repositories:
 - **CRAN**: Comprehensive R Archive Network(CRAN) is the official repository; it is a network of ftp and web servers maintained by the R community around the world. Reviewed and curated.
 - **Bioconductor** is a topic-specific repository, intended for open-source software for bioinformatics. Reviewed and curated.
 - **GitHub** is the most popular repository for open-source projects. *Not* reviewed and curated.



Installing packages

- Point-and-click available for CRAN (example for the package cowplot)

- Or for CRAN use the command :

```
install.packages("cowplot")
```

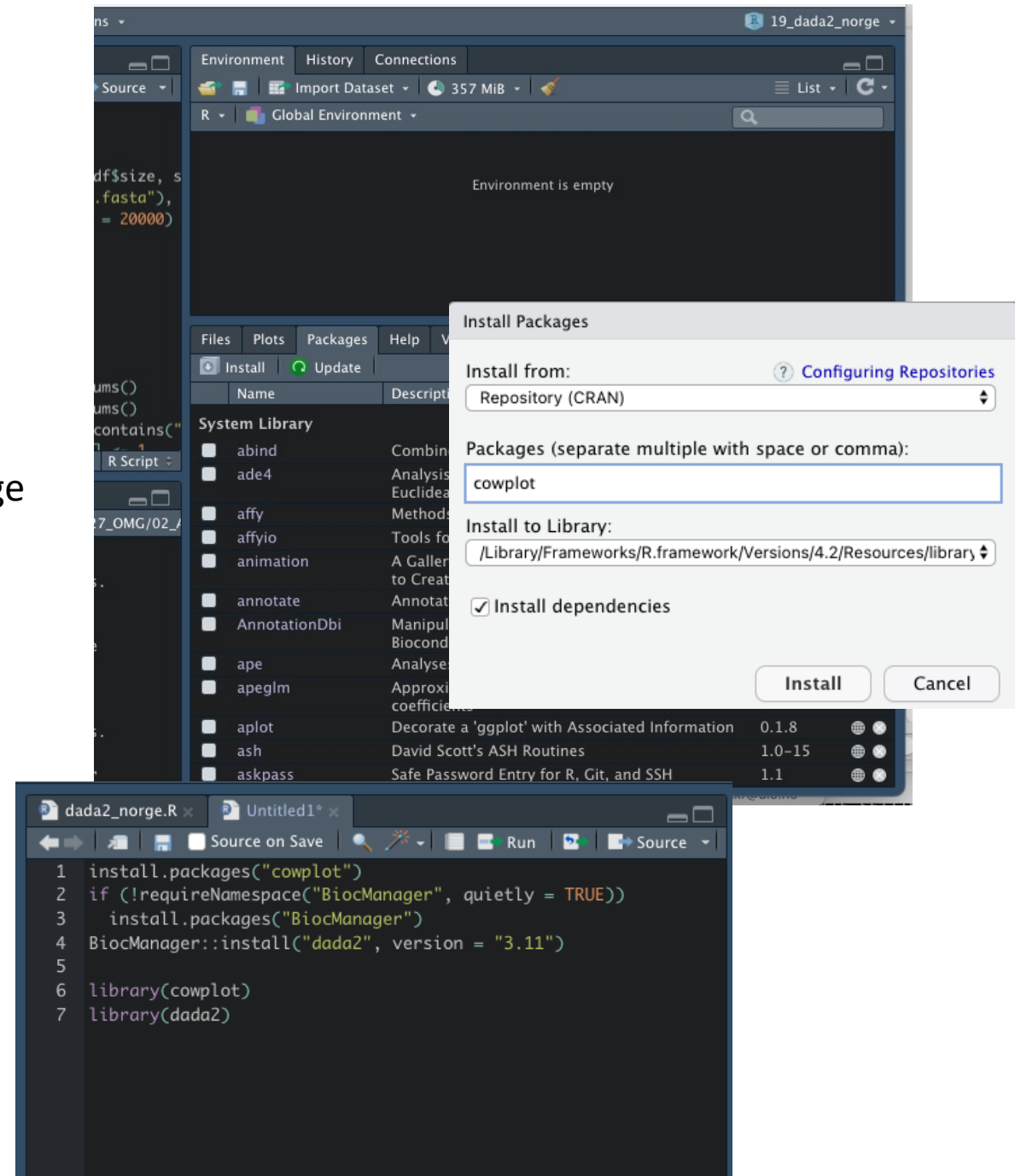
- For Bioconductor packages use (with dada2 as example):

```
if (!requireNamespace("BiocManager", quietly = TRUE))
```

```
install.packages("BiocManager")
```

```
BiocManager::install("dada2", version = "3.11")
```

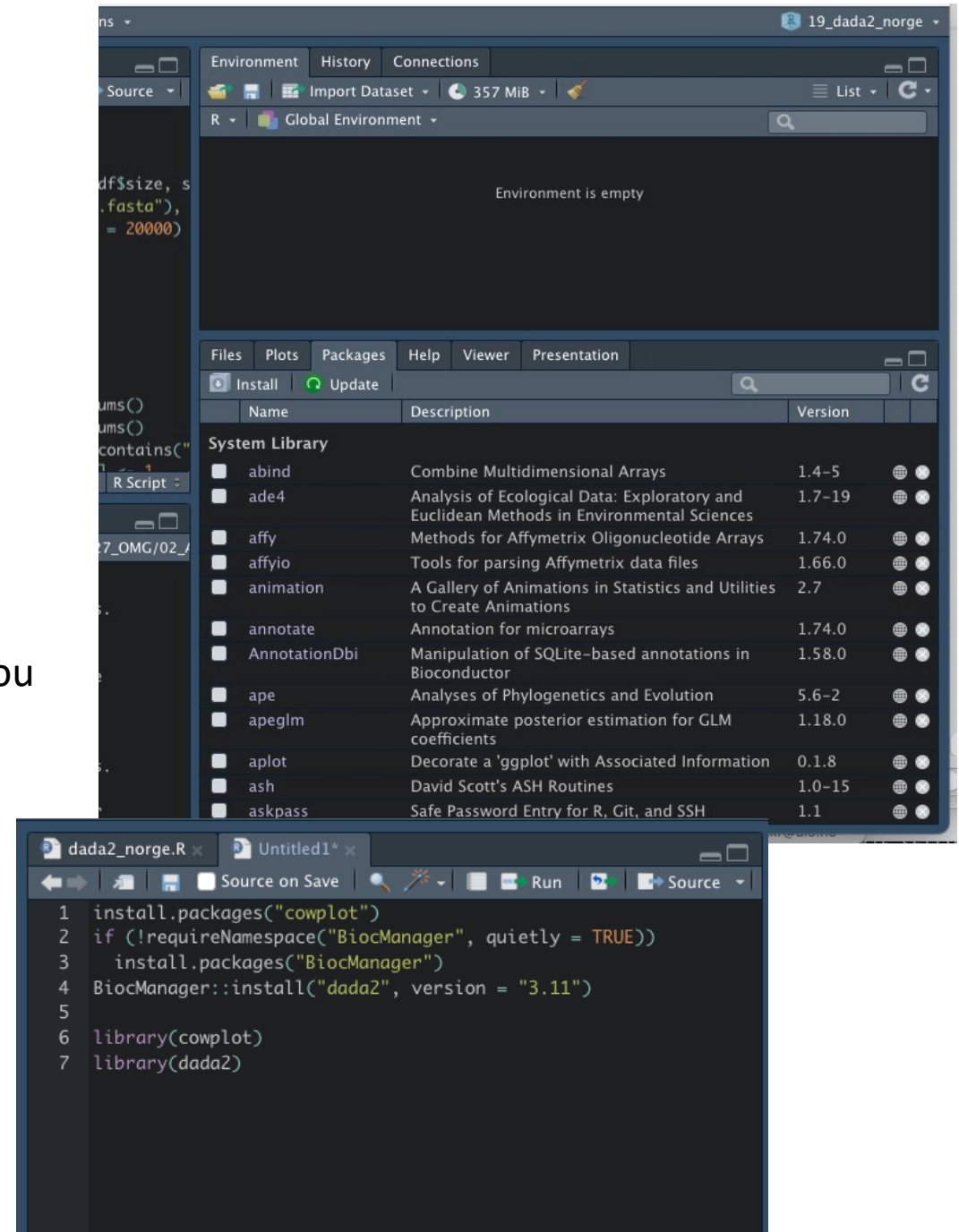
- The installation will take care of installing dependencies if necessary



Loading packages

- Once a package is installed you need to load it into the environment for the functions to be available for the *current session*
- Point-and-click is available (but not recommended since you want to have the libraries in your script)
- The command for loading a package is `library()`:

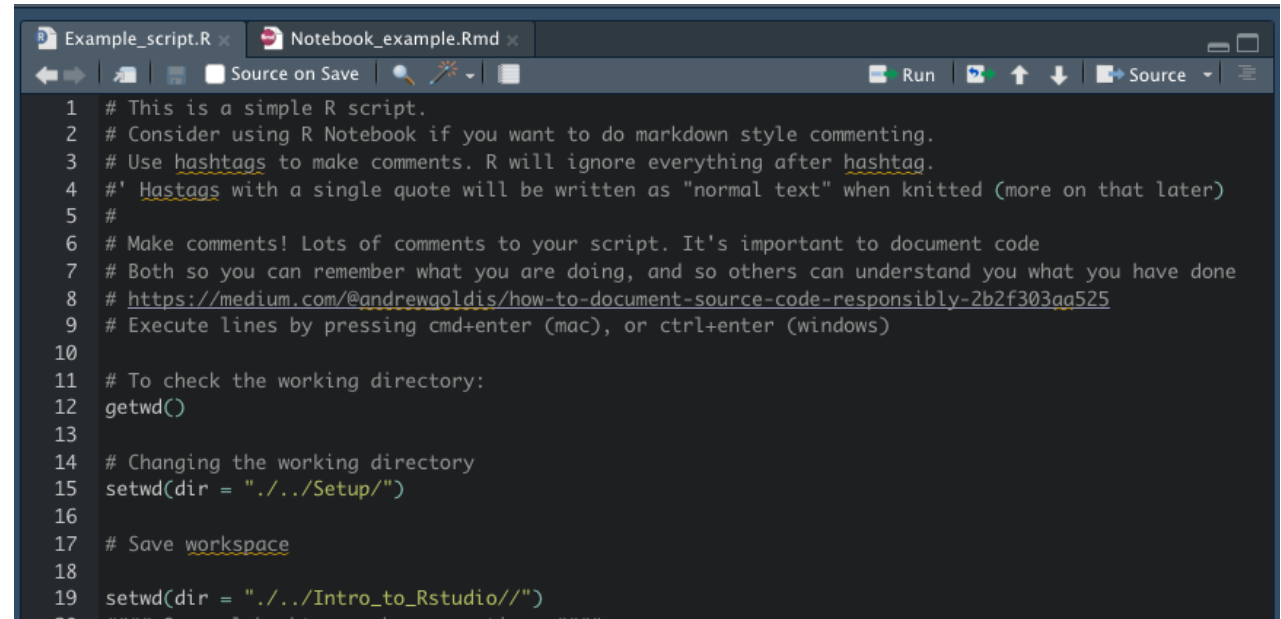
```
library(cowplot)  
library(dada2)
```
- When you close Rstudio the package will be unloaded and you have to reload the package when you start your next session



Use scripts or R Notebook

- Scripts and R notebooks allows you to save your process and makes it easy to share the work, or re-run any analysis.
- Scripts are simple text files that contains R commands executed line by line. All lines not preceded by a # will be executed
- R Notebooks are more complex, with chunks of code and running text together in the same document. Only the chunks will be executed (more on that later).

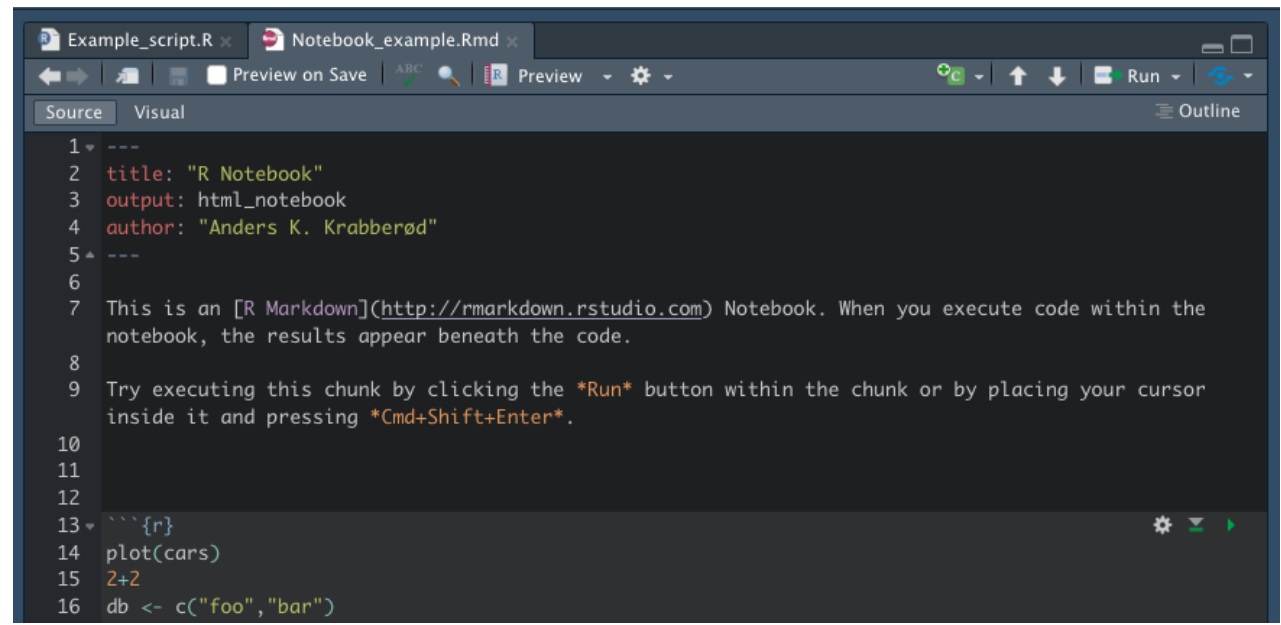
R script example:



The screenshot shows an R script file named 'Example_script.R' open in RStudio. The script contains 19 lines of code, primarily consisting of comments starting with '#'. The comments provide instructions on how to use R scripts, including how to comment, how to execute lines, and how to change the working directory. The code includes the following lines:

```
1 # This is a simple R script.
2 # Consider using R Notebook if you want to do markdown style commenting.
3 # Use hashtags to make comments. R will ignore everything after hashtag.
4 #' Hastags with a single quote will be written as "normal text" when knitted (more on that later)
5 #
6 # Make comments! Lots of comments to your script. It's important to document code
7 # Both so you can remember what you are doing, and so others can understand you what you have done
8 # https://medium.com/@andrewgoldis/how-to-document-source-code-responsibly-2b2f303aa525
9 # Execute lines by pressing cmd+enter (mac), or ctrl+enter (windows)
10
11 # To check the working directory:
12 getwd()
13
14 # Changing the working directory
15 setwd(dir = "../Setup/")
16
17 # Save workspace
18
19 setwd(dir = "../Intro_to_Rstudio/")
```

R Notebook example:



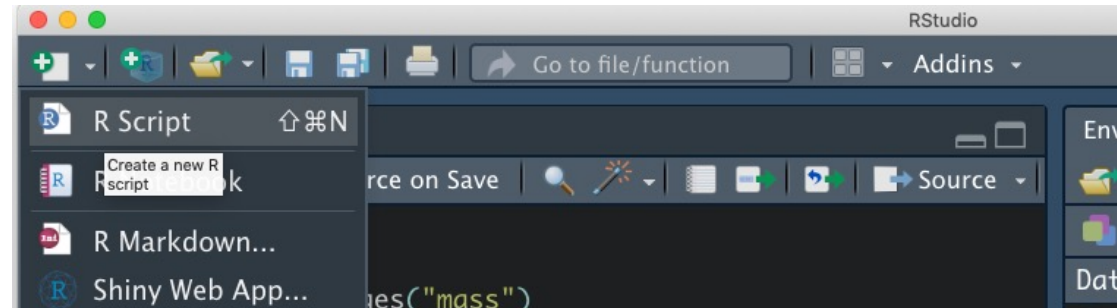
The screenshot shows an R Notebook file named 'Notebook_example.Rmd' open in RStudio. The notebook is displayed in the 'Source' view, showing 16 lines of code. The code includes a title, author, and a chunk of R code. The code includes the following lines:

```
1 ---
2 title: "R Notebook"
3 output: html_notebook
4 author: "Anders K. Krabberød"
5 ---
6
7 This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code within the
8 notebook, the results appear beneath the code.
9
10 Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor
11 inside it and pressing *Cmd+Shift+Enter*.
12
13 ```{r}
14 plot(cars)
15 2+2
16 db <- c("foo", "bar")
```

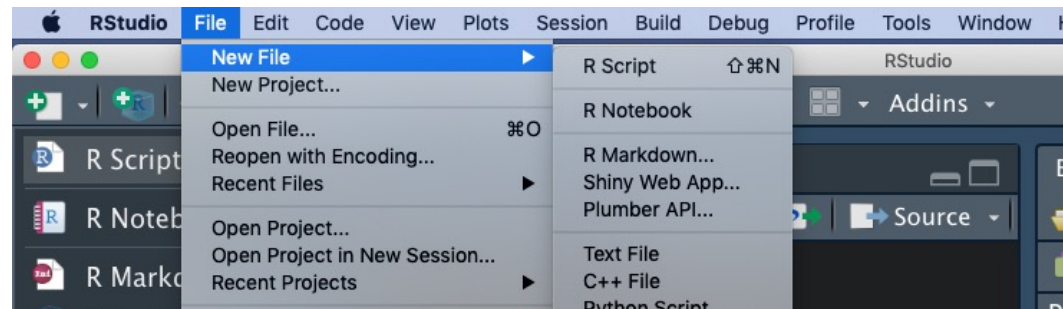
Create an R script

- Scripts are simple text files that contains R commands executed line by line.

Click icon with a document and a + sign



OR click File -> New File -> R Script



Using Scripts

The screenshot shows the RStudio interface with a script file named 'Test.R' open. The script contains the following R code:

```
1 db<-as.data.frame(matrix(c(1,2,3,4,5,6,7,8),
2                             ncol=4,nrow=2))
3 db
4 |
5
6
7
8
9 4:1
```

The Environment pane on the right shows the object 'db' with 2 observations and 4 variables. The console at the bottom shows the output of the script:

```
>
>
> db<-as.data.frame(matrix(c(1,2,3,4,5,6,7,8), ncol=4,nrow=2))
> db
  V1 V2 V3 V4
1  1  3  5  7
2  2  4  6  8
>
```

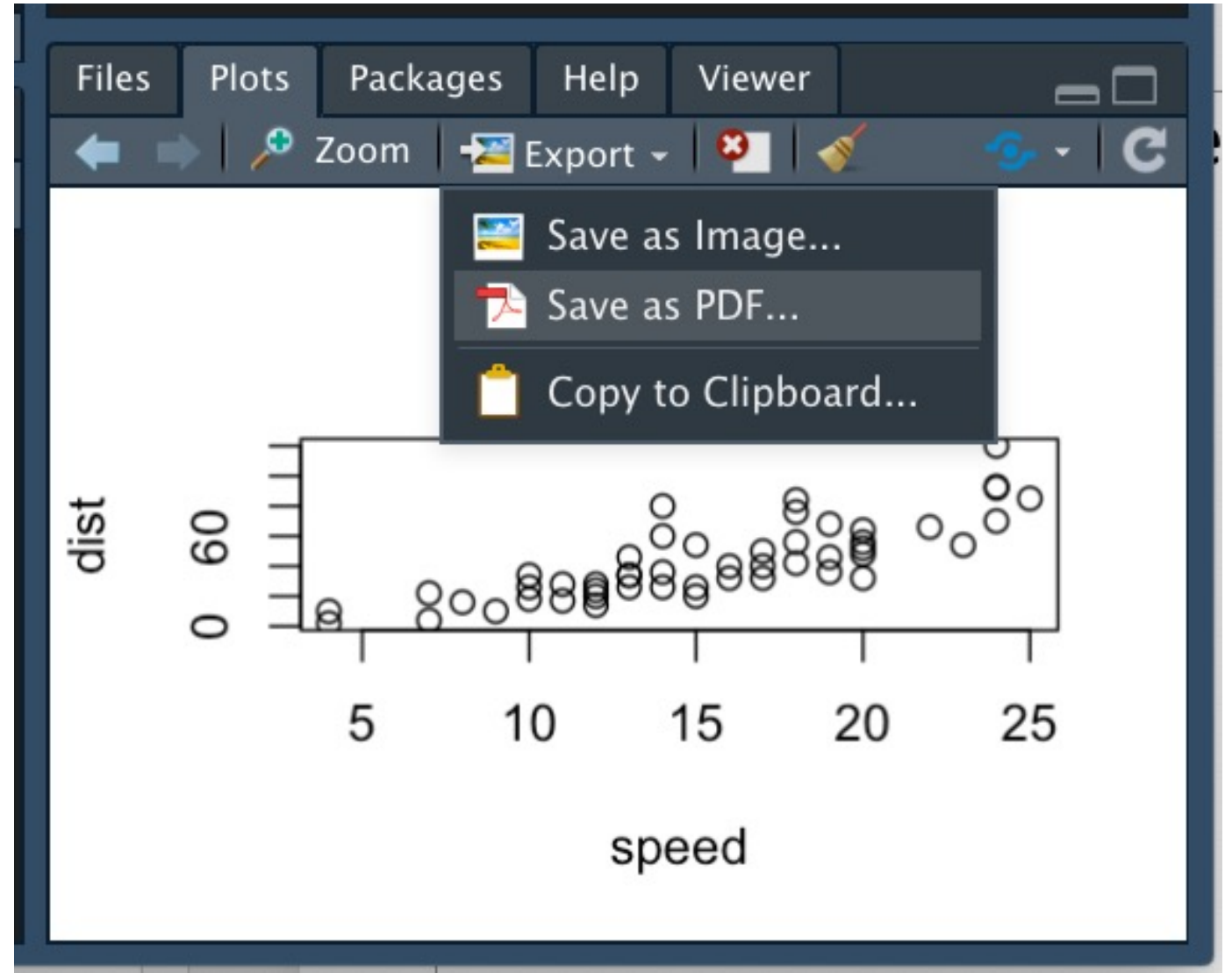
- Write in the script, execute lines by
- cmd+R (mac)
- Ctrl+R (win)
- Remember to annotate!!!

- **Environment** shows the object

- Output in the console

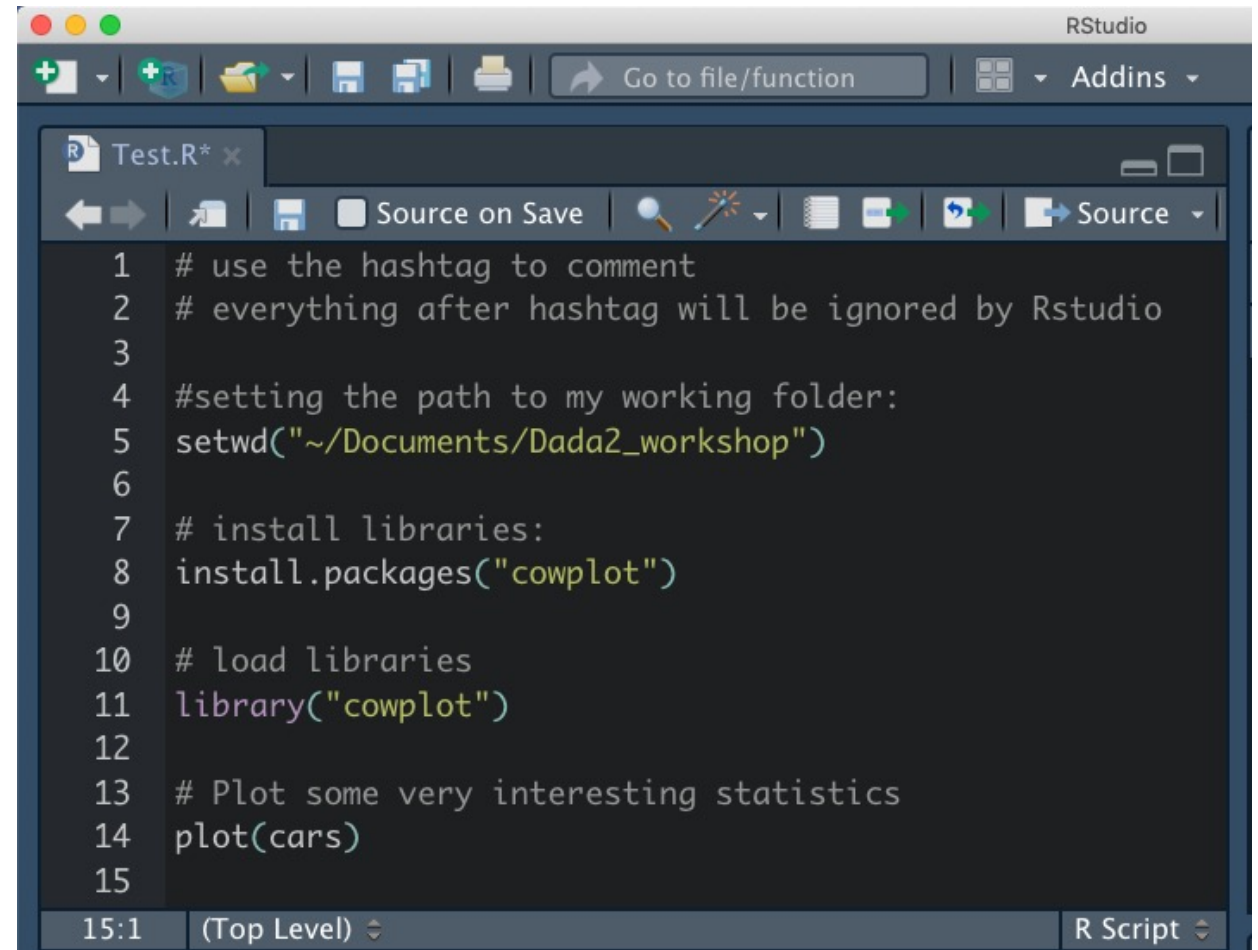
Plotting plots and other dots

- Plots will appear in the *plots* tab and can be exported in various formats.
- Additionally it is possible to give plotting commands in the script that will export the graphics to your working directory.
- The package *ggplot2* is very powerful and contains several plotting functions.



Comment and annotate your script!!!

- What the code does
- How the code does it
- How to use the code



The screenshot shows the RStudio interface with a script editor open. The script contains the following R code with line numbers 1 through 15:

```
1 # use the hashtag to comment
2 # everything after hashtag will be ignored by Rstudio
3
4 #setting the path to my working folder:
5 setwd("~/Documents/Dada2_workshop")
6
7 # install libraries:
8 install.packages("cowplot")
9
10 # load libraries
11 library("cowplot")
12
13 # Plot some very interesting statistics
14 plot(cars)
15
```

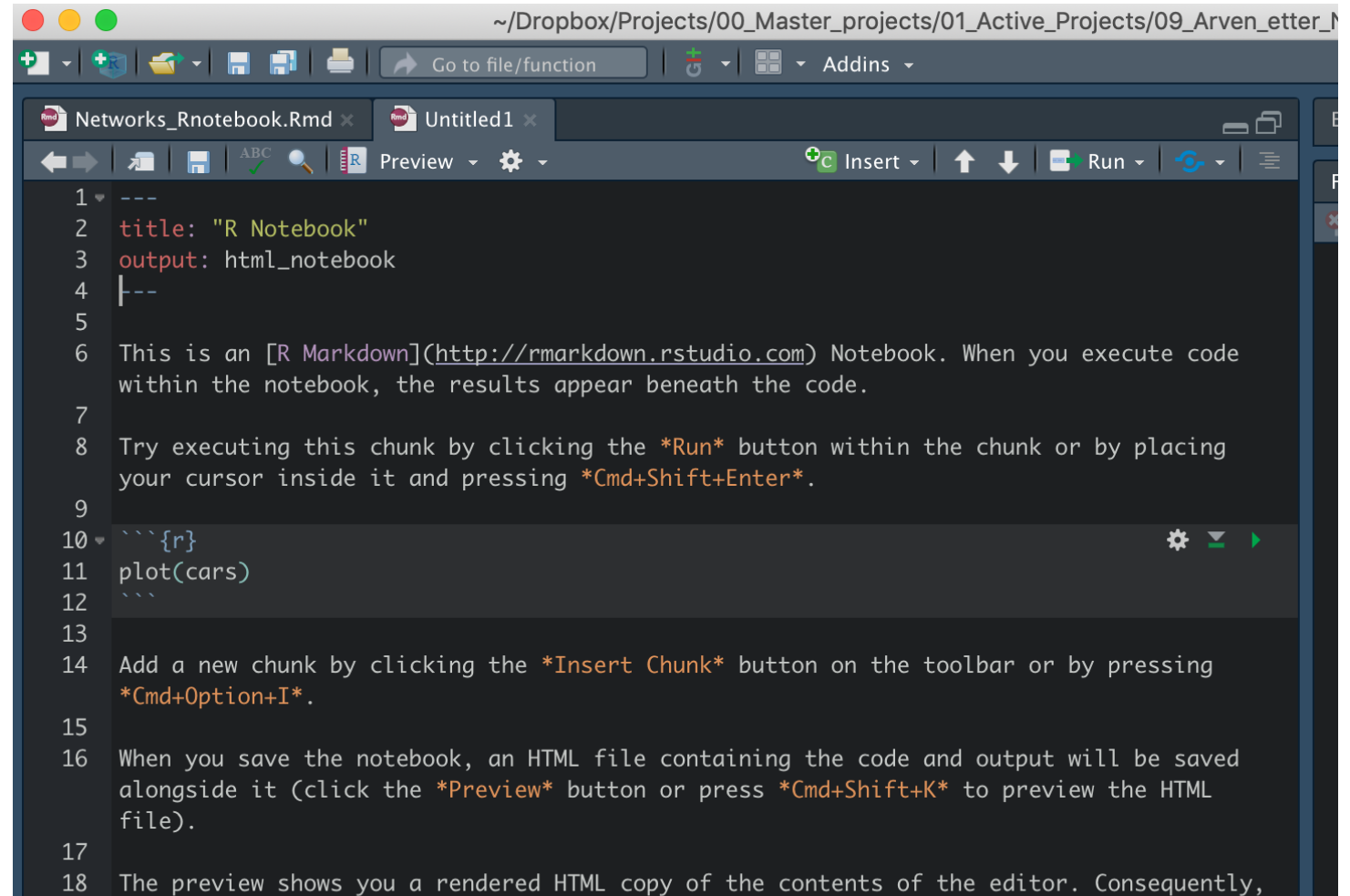
The status bar at the bottom indicates the cursor is at line 15, column 1, at the top level of the script.

R notebooks and markdown

- R Notebook is an alternative to “simple” script in Rstudio.
- **Advantage:** easy to export in other easy-to-read formats (i.e. html, pdf, word, presentations).
- Markdown language is an easy way of formatting using plain text
- R Notebook is somewhat more powerful with additional options for formatting.
- Can run chunks of code from other languages *within* Rstudio
- **Disadvantage:** Not compatible with (standalone) R, which is often used on clusters and servers. For instance running R on Saga can be done with a simple script.

R Notebook

- The first lines are called the YAML (YAML Ain't Markup Language) and contains the metadata and options for the entire document such as the author name, date, output format, etc. The beginning and the end of the YAML is marked by three dashes (---).
- Code is run in chunks placed between three *opening single quotation marks*, in the curly bracket the type of code is defined. For R it is {r}.
- Inserting a chunk of code is done by pressing **Cmd+Option+I** on macs or **Ctrl+Alt+I** on Windows

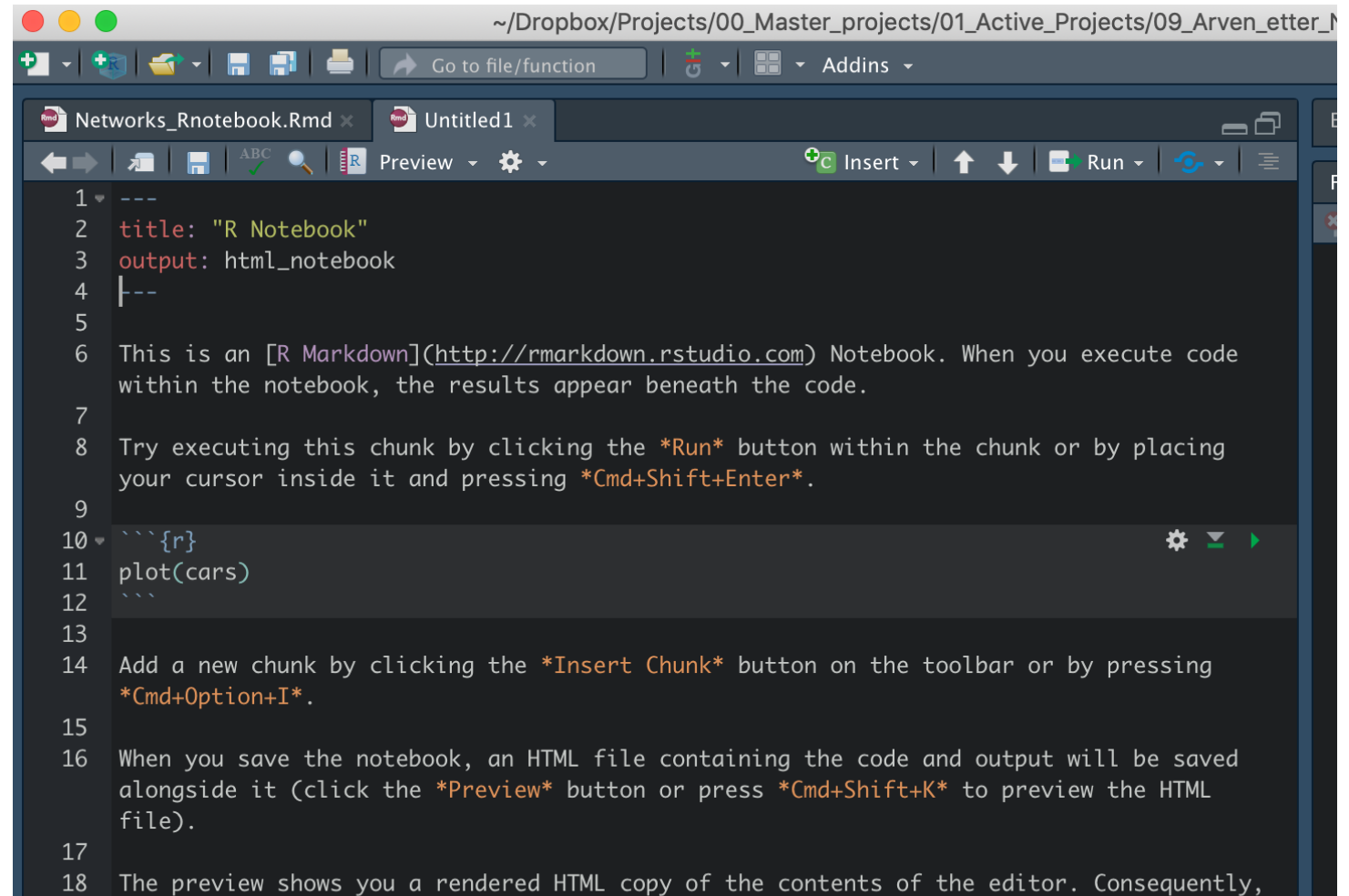


The screenshot shows the RStudio interface with an R Notebook. The title bar indicates the file path: ~/Dropbox/Projects/00_Master_projects/01_Active_Projects/09_Arven_etter_I. The notebook has two tabs: 'Networks_Rnotebook.Rmd' and 'Untitled1'. The editor shows the following content:

```
1 ---
2 title: "R Notebook"
3 output: html_notebook
4 ---
5
6 This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code
7 within the notebook, the results appear beneath the code.
8
9 Try executing this chunk by clicking the *Run* button within the chunk or by placing
10 your cursor inside it and pressing *Cmd+Shift+Enter*.
11
12 ```{r}
13 plot(cars)
14 ```
15
16 Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing
17 *Cmd+Option+I*.
18
19 When you save the notebook, an HTML file containing the code and output will be saved
20 alongside it (click the *Preview* button or press *Cmd+Shift+K* to preview the HTML
21 file).
22
23 The preview shows you a rendered HTML copy of the contents of the editor. Consequently,
```

R Notebook

- Text between the chunks will not be executed, and can be formatted with the markdown syntax.
- When you save the R Notebook a html file containing the output will be saved.
- The format can be changed to pdf, word etc.
- If you want to learn more about R Notebook and R Markdown this is a good source:
https://intro2r.com/rmarkdown_r.html



The screenshot shows the RStudio interface with a file named "Networks_Rnotebook.Rmd" open. The editor displays a code chunk with the following content:

```
1 ---
2 title: "R Notebook"
3 output: html_notebook
4 |---
5
6 This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code
7 within the notebook, the results appear beneath the code.
8
9 Try executing this chunk by clicking the *Run* button within the chunk or by placing
10 your cursor inside it and pressing *Cmd+Shift+Enter*.
11
12 ```{r}
13 plot(cars)
14 ```
15
16 Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing
17 *Cmd+Option+I*.
18
19 When you save the notebook, an HTML file containing the code and output will be saved
20 alongside it (click the *Preview* button or press *Cmd+Shift+K* to preview the HTML
21 file).
22
23 The preview shows you a rendered HTML copy of the contents of the editor. Consequently,
```

The interface includes a toolbar with buttons for navigation, editing, and execution (Run, Insert, etc.). The status bar at the bottom shows the file path: ~/Dropbox/Projects/00_Master_projects/01_Active_Projects/09_Arven_etter_I

Save and load your work

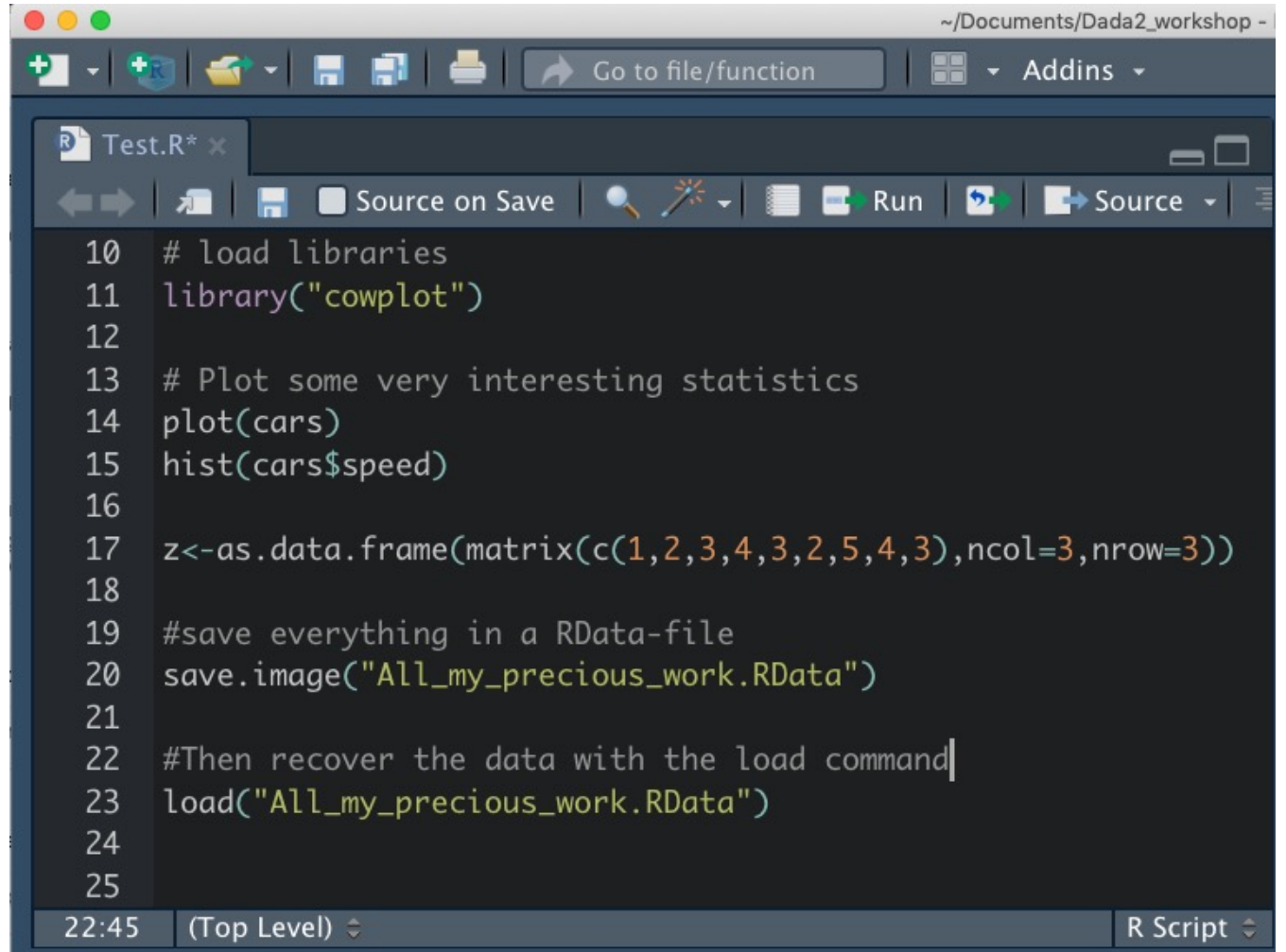
- You can save your “workspace” this will save all created objects and variables (but **not** plots and packages).

```
save.image("filename.Rdata")
```

- Retrieve with

```
load("filename.Rdata")
```

- Easy way to keep all analyses without having to rerun everything.
- You can also save single objects or a selection of objects with `saveRDS()`

A screenshot of the RStudio interface. The title bar shows the file path '~ / Documents / Dada2_workshop'. The menu bar includes 'Go to file/function' and 'Addins'. The toolbar has icons for file operations and a 'Run' button. The source editor shows a script named 'Test.R*' with the following R code:

```
10 # load libraries
11 library("cowplot")
12
13 # Plot some very interesting statistics
14 plot(cars)
15 hist(cars$speed)
16
17 z<-as.data.frame(matrix(c(1,2,3,4,3,2,5,4,3),ncol=3,nrow=3))
18
19 #save everything in a RData-file
20 save.image("All_my_precious_work.RData")
21
22 #Then recover the data with the load command
23 load("All_my_precious_work.RData")
24
25
```

The status bar at the bottom shows the time '22:45', the environment '(Top Level)', and the file type 'R Script'.

Use R-projects

- This will set the default working directory for the particular project, and makes it easy to save everything in the same folder.
- Very helpful when working on several different projects
- Also very easy to integrate with *github* and version control with the option to push and pull repositories (not covered in this workshop)
- Or for sharing all data with somebody else using Rstudio

