Statistical Computing with R
Masters in Data Science 503 (S6)
Fourth Batch, SMS, TU, 2025

Shital Bhandary
Associate Professor
Statistics/Bio-statistics, Demography and Medical Education
Patan Academy of Health Sciences, Lalitpur, Nepal
Faculty, Master in Medical Research, NHRC/Kathmandu University
Faculty, FAIMER Fellowship in Health Professions Education, India/USA

Review Preview

- **Markdown**
- **YAML**
- **Text**
- **Code**
- **Authoring/Rendering**
- **HTML**
- **PDF**
- **Word**

Reproducible outputs: Markdown

- **Markdown** is described as: "*Text-to-HTML conversion tool/syntax*".

- Markdown is two things:
  - a plain text formatting syntax; and
  - a software tool, written in Perl, that converts the plain text formatting to HTML.

Reproducible outputs: YAML – “The Title”

- On the other hand, **YAML** is detailed as "*A straightforward machine parsable data serialization format designed for human readability and interaction*".

- **YAML** is a human-readable data-serialization language. It is commonly used for configuration files, but could be used in many applications where data is being stored or transmitted.

- YAML = Yet Another Markup Language in 2001 (YAML Ain’t Markup Language from 2002 onwards = NOT FOR DOCUMENT MARKUP)

R Markdown and knitr in R Studio: Dynamic Report Generation

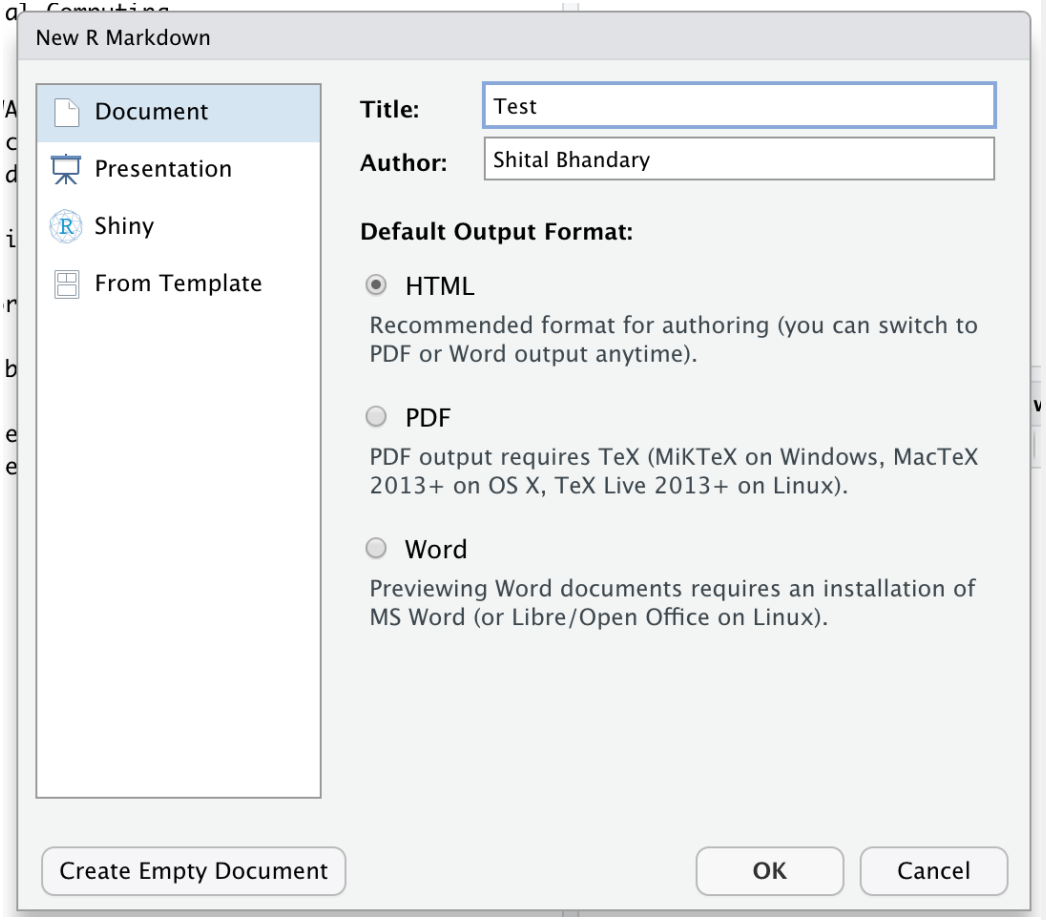
- You cannot execute any R code in a plain Markdown document
- You can embed the R code in plain Markdown using syntax for fenced code block ```r i.e. without curly braces but it will not be executed!
- You can embed R code chunks (```{r}) in an R Markdown document
- More here:
  - https://cran.r-project.org/web/packages/rmarkdown/index.html
  - https://sachsmc.github.io/knit-git-markr-guide/knitr/knit.html
  - https://github.com/rstudio/bookdown

R Studio: File → New File → R Markdown

- New R Markdown → Document → Title → Test → OK
- What do you get?
- Click the “knit” button → “Test” → Save
- It will save “Test.html” in your working directory

Recommended reading: https://rmarkdown.rstudio.com/lesson-2.html

You will get this:



Untitled1

Knit | Run

```
title: "Test"
author: "Shital Bhandary"
date: "4/1/2023"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```{r cars}
summary(cars)
```

Including Plots

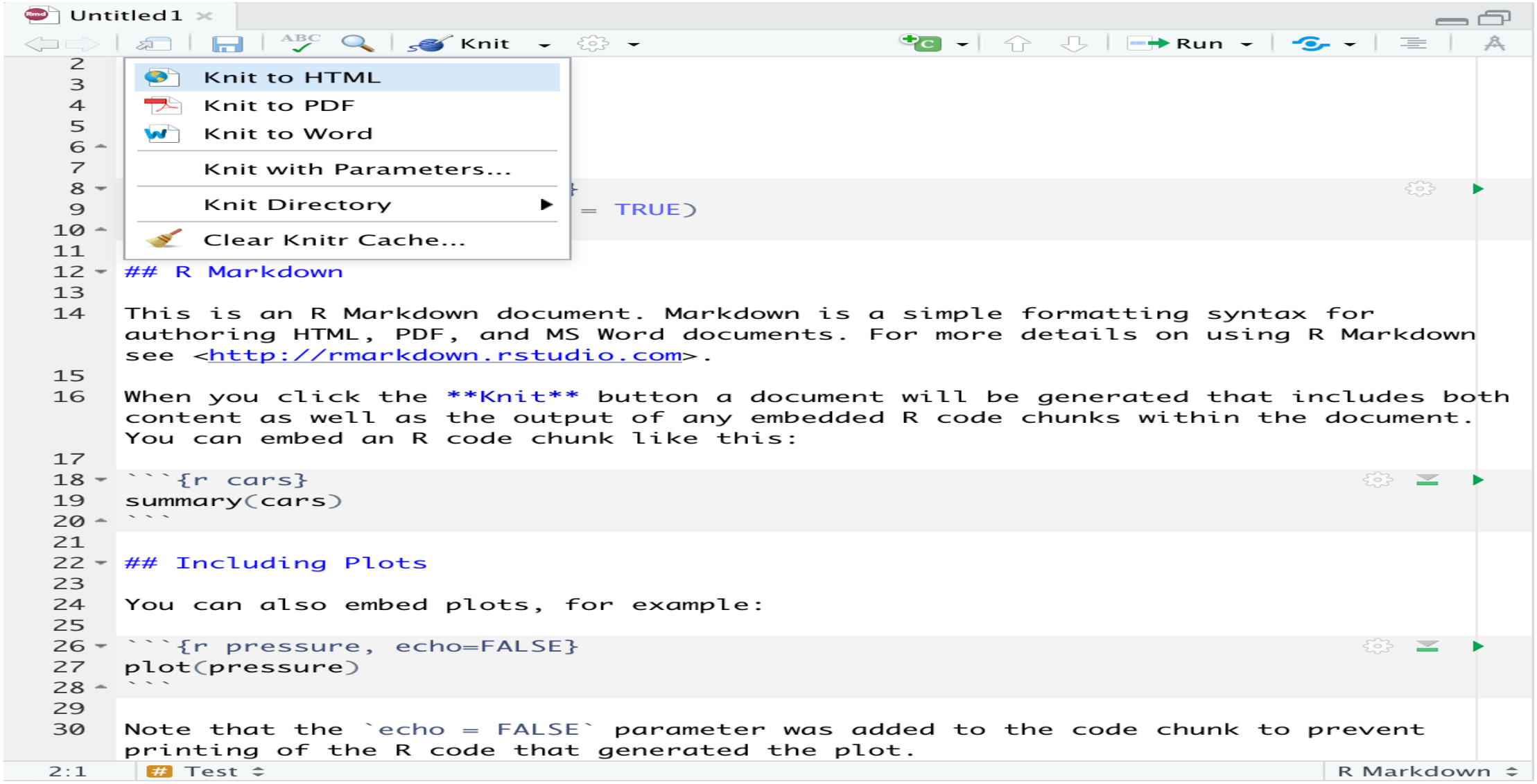
You can also embed plots, for example:

```{r pressure, echo=FALSE}
plot(pressure)
```

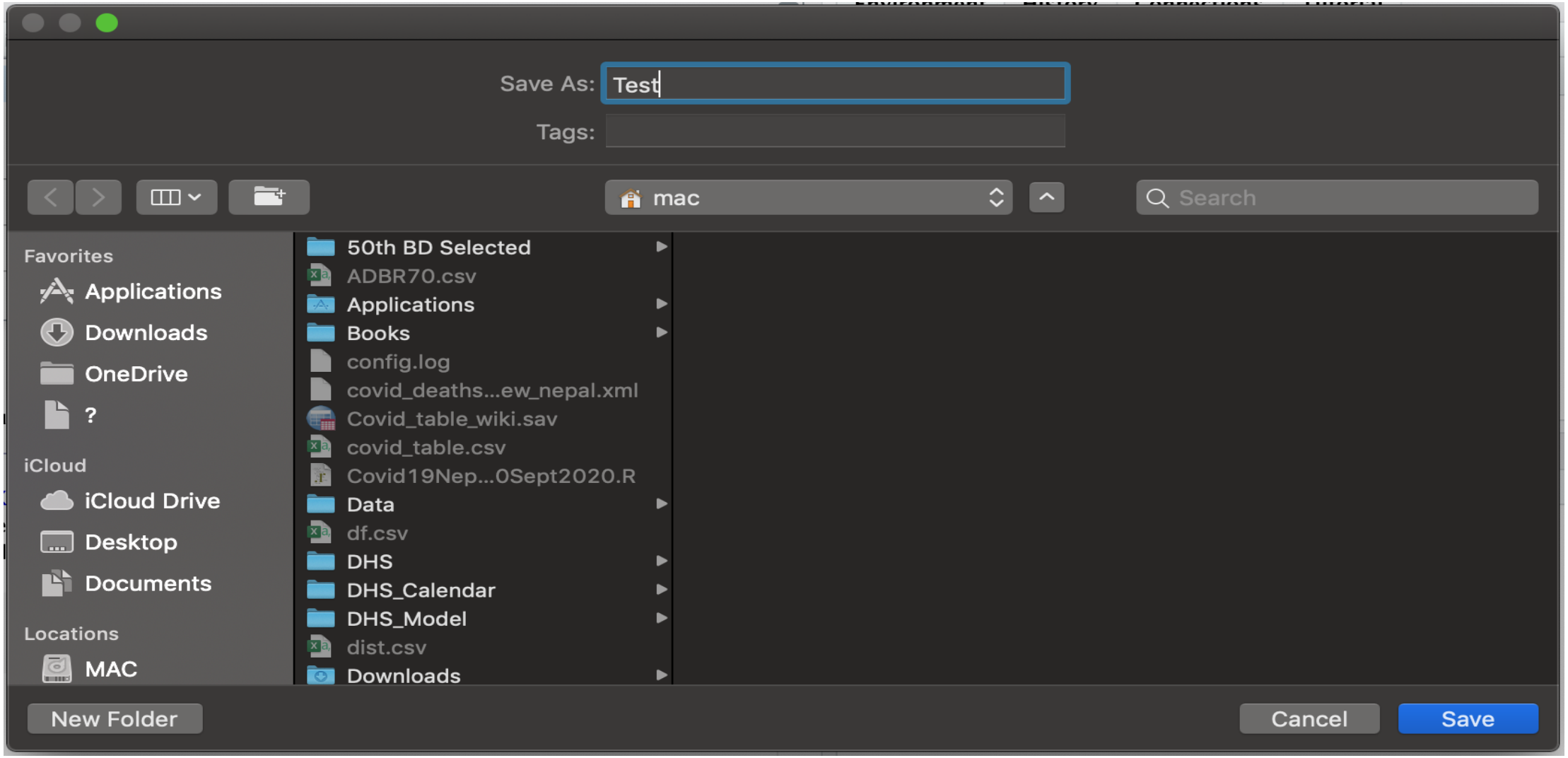
Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
```

2:1 | Test | R Markdown

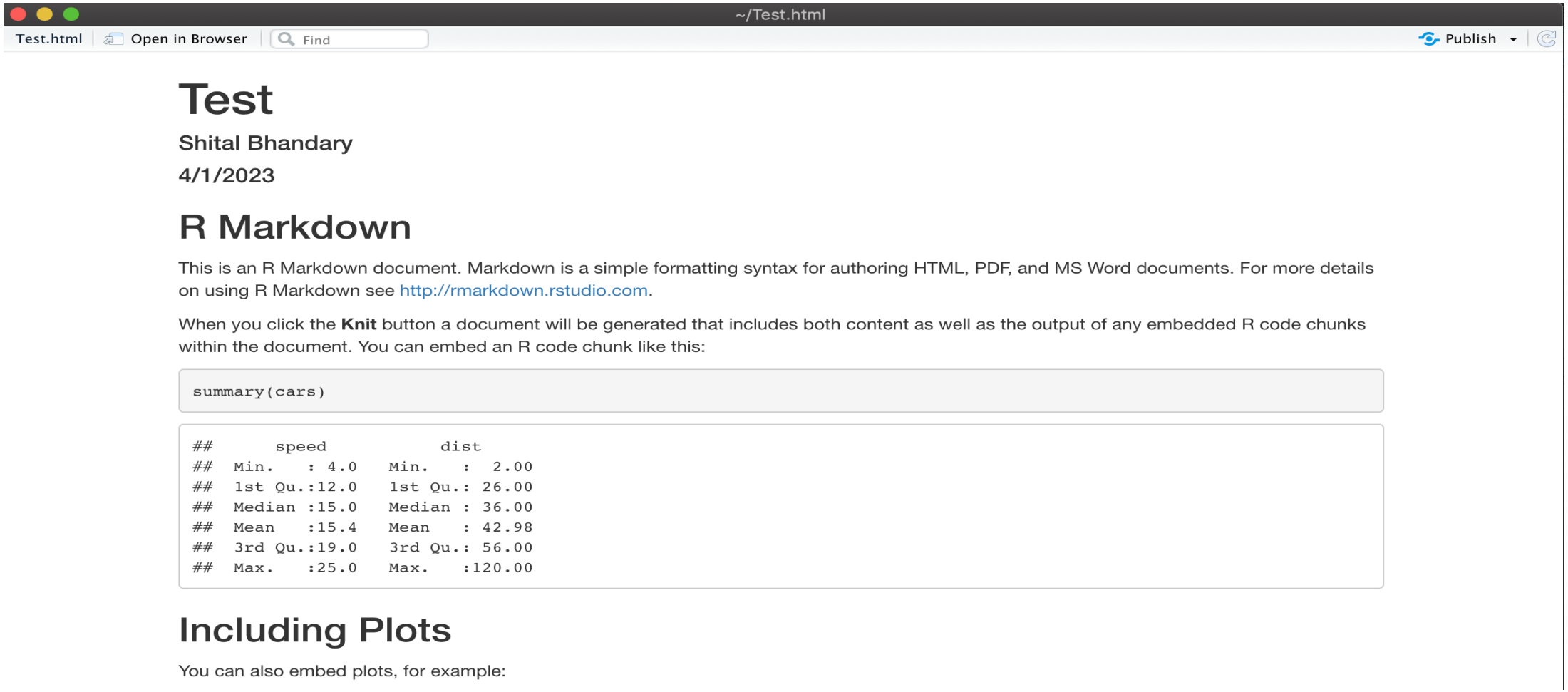
Then “knit” it to get ‘html’ or ‘pdf’ or ‘word’



You will be asked to save it:



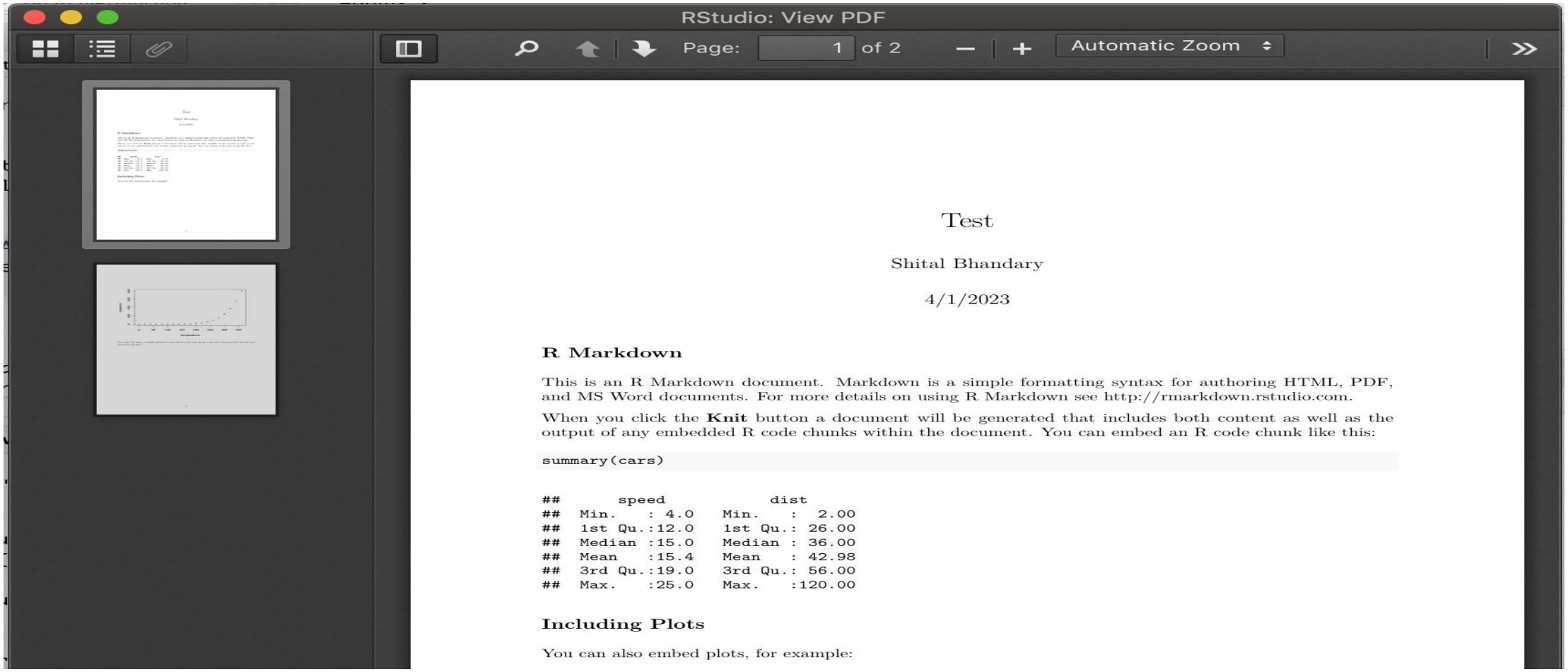
To get the HTML file with R Markdown:



R Studio: File → New File → R Markdown

- New R Markdown → Document → Title → Test → OK
- What do you get?
- Click the “knit” button → ”Knit to PDF” → “Test” → Save
- It will save “Test.pdf” in your working directory if you have the required LaTex to PDF package like TinyTex (you can install it with this command in R: tinytex::install_tinytex() if required!)

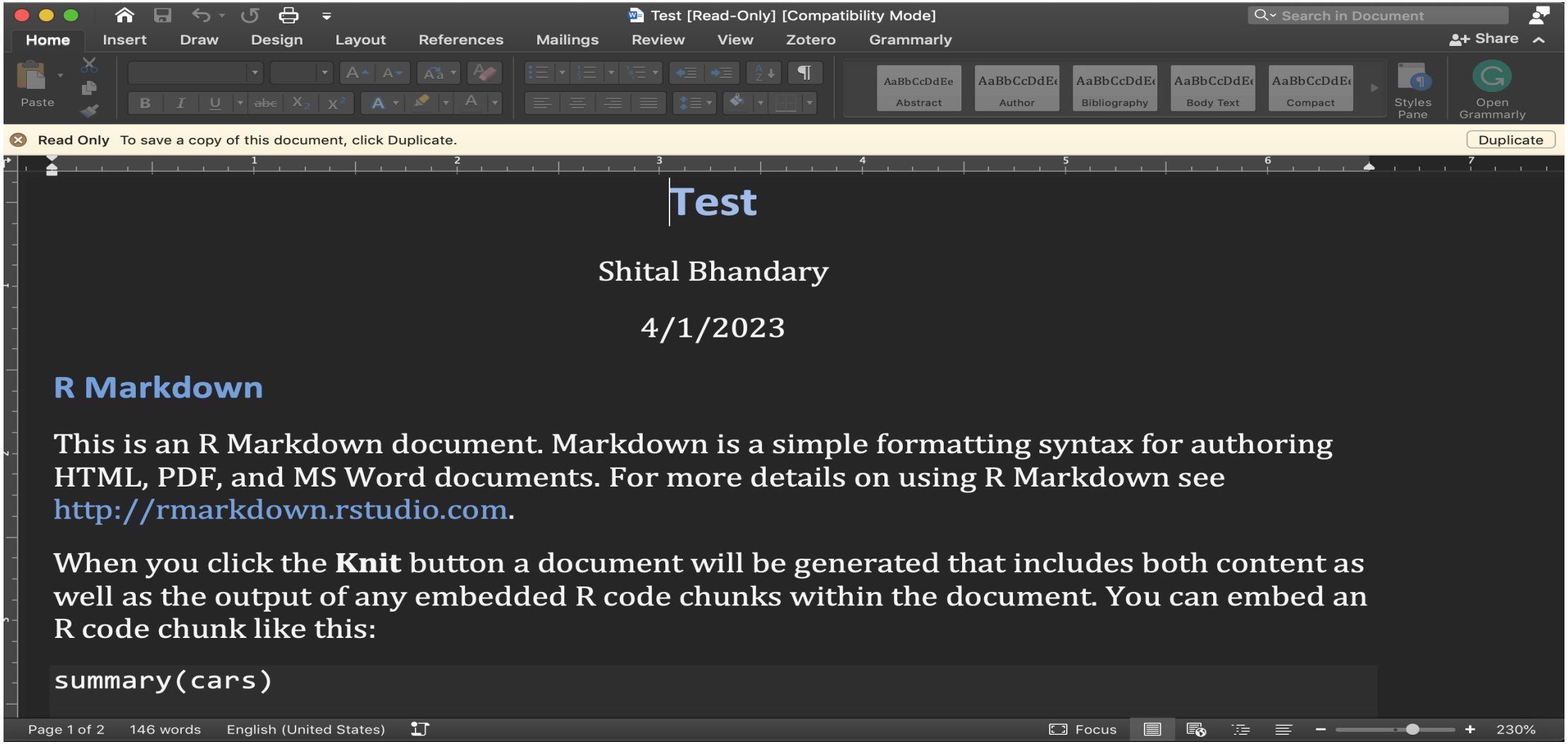
You will get this then:



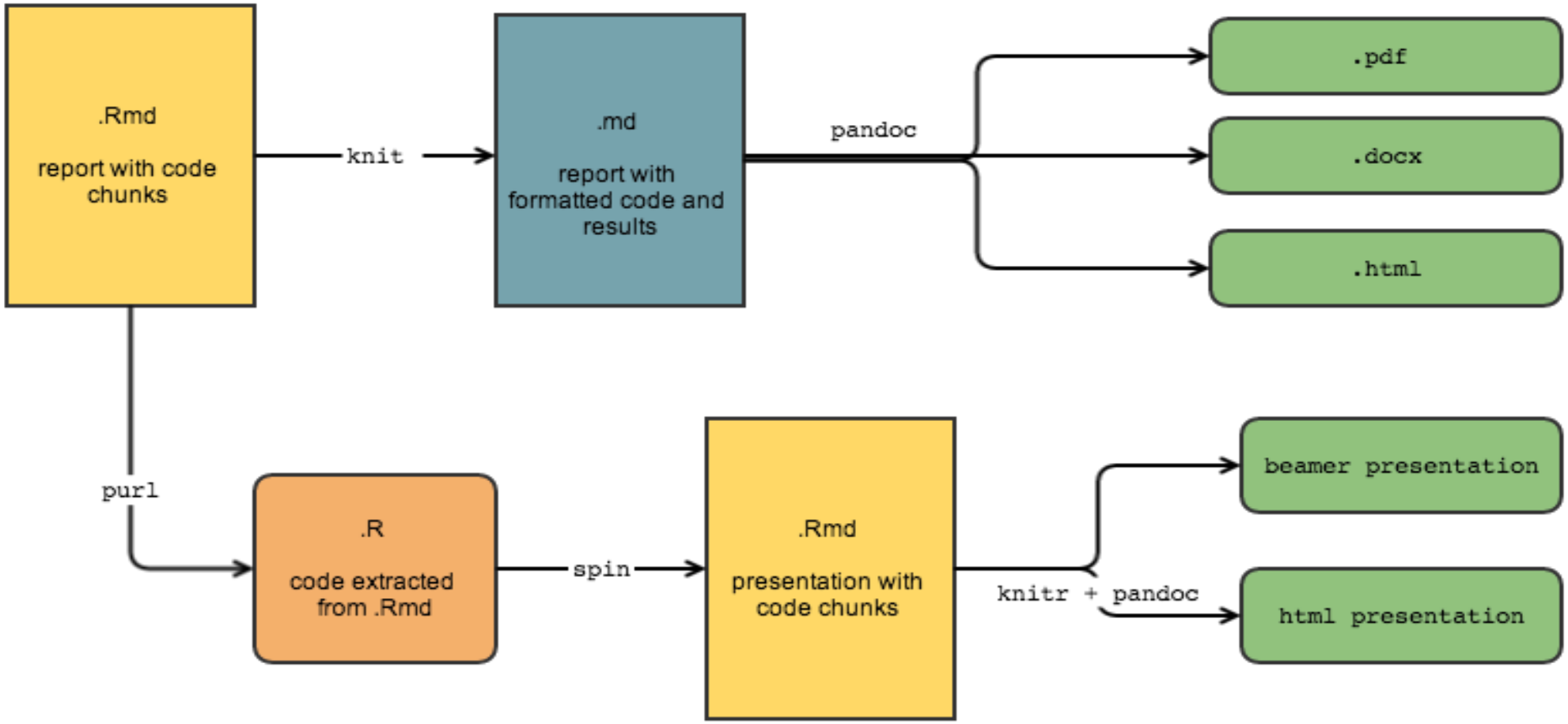
R Studio: File → New File → R Markdown

- New R Markdown → Document → Title → Test → OK
- What do you get?
- Click the “knit” button → ”Knit to Word” → “Test” → Save
- It will save “Test.docx” in your working directory if you have the MS Word software in your computer (you need to provide access to write using MS Word when asked!)

You will get this if all goes well:



R Markdown Workflow in R Studio:



https://sachsmc.github.io/knit-git-markr-guide/knitr/knit.html

So what?

- You must prepare all the reports using R markdown in this course
- You must be able to knit your file in HTML and PDF format
- You might need to install a separate package to knit the PDF files
- Make sure that you/your machine can do it!

Questions/queries so far?

Profiling and Optimizing Codes in R

https://bookdown.org/rdpeng/rprogdatascience/profiling-r-code.html

- R comes with a profiler to help you optimize your code and improve its performance.
- In general, it's usually a bad idea to focus on optimizing your code at the very beginning of development.
- Rather, in the beginning it's better to focus on translating your ideas into code and writing code that's coherent and readable.

Profiling and Optimizing Codes in R

https://bookdown.org/rdpeng/rprogdatascience/profiling-r-code.html

- The problem is that heavily optimized code tends to be obscure and difficult to read, making it harder to debug and revise.

- Better to get all the bugs out first, then focus on optimizing.

Profiling

- Profiling is a systematic way to examine how much time is spent in different parts of a program.

- The reality is that *profiling is better than guessing*.

- The system.time() function computes the time (in seconds) needed to execute an expression and if there's an error, gives the time until the error occurred.

R profiler

- Rprof() #Turn on the R profiler
  - In conjunction with Rprof(), we will use the summaryRprof() function which summarizes the output from Rprof() (otherwise it's not really readable)
  - **You should NOT use system.time() and Rprof() together!**
  - Once you call the Rprof() function, everything that you do from then on will be measured by the profiler.

- Rprof(NULL) #Turn off the profiler

- **Read: Chapter 19- Profiling R code (R Programming for Data Science)**

Profiling R code with R Studio IDE

https://support.posit.co/hc/en-us/articles/218221837-Profiling-R-code-with-the-RStudio-IDE

- As R users, many, perhaps most, of us have had times where we've wanted our code to run faster.
- However, it's not always clear how to accomplish this.
- A common approach is to rely on our intuitions, and on wisdom from the broader R community about speeding up R code.
- **e.g., that apply functions are inherently faster than for loops**

- One drawback to this is it can lead to a focus on optimizing things that actually take a small proportion of the overall running time.

Example: With “loop” in R for row mean

- N <- 10000
- x1 <- runif(N)
- x2 <- runif(N)
- d <- as.data.frame(cbind(x1, x2))

- system.time(for (loop in c(1:length(d[, 1]))) { d$mean2[loop] <- mean(c(d[loop, 1], d[loop, 2])) })

- # user system elapsed
- # 13.912 0.204 14.150

Example: With built-in “apply” function

- N <- 10000
- x1 <- runif(N)
- x2 <- runif(N)
- d <- as.data.frame(cbind(x1, x2))

- system.time(d$mean1 <- apply(d, 1, mean))

```
# user   system   elapsed
# 0.180   0.000     0.179
```

```
#apply (x, 1 or 2, function)
# 1=Row; 2=Column
```

E.G.: With vectorized ‘rowMeans’ function

- N <- 10000
- x1 <- runif(N)
- x2 <- runif(N)
- d <- as.data.frame(cbind(x1, x2))

- system.time(d$mean3 <- rowMeans(d[, c(1, 2)]))

```
# user    system   elapsed
# 0.004     0.000     0.002
```

Comparison:

- Bad way, 15 seconds
- `x <- c() for (i in 1:1e+05) { x <- c(x, i) }`
- Good way (0.001 seconds)
- `y <- seq(1, 1e+05)`
- Better way (<0 seconds)
- `z <- 1:1e+05`

Questions/queries?

Unit 1: Project work

- All the examples and exercises provided in Unit 1 (Session 1-6) must be done in R Studio and saved it as “Rollnumber_Unit1.R”
- Write interpretation of the outcomes of those codes
- Compile/knit it as HTML file in R Studio
- Submit the .R and HTML files in Google Classroom for review and grading

Final note on Unit 1 of the syllabus:

- This is the end of “Unit 1” of the syllabus
- You need to do a project and submit it the Google classroom
- I will create the project based on the learning of this unit soon so that you can complete it and submit it there
- Happy learning!

Thank you!

@shitalbhandary