Lab 4.1 - Introduction to RStudio and R Markdown. Presentations

Rakesh Kumar

September 27, 2022

In Lab 4, you will learn how to use, edit and create a R Markdown document (like this one) using RStudio. You should follow the instructions in this document to complete the assignment. Knit this document to view the nicely rendered HTML, which can make it easier to read the questions.

If you need help as you use R Markdown in this lab and others in the future, consult the following resources:

- Cheat sheet
- Home page with guides
- Reference book

The below is a code chunk, but instead of using the `r` engine we're creating and alert block that will make the question show up with a blue background in the HTML output. Unfortunately, this creates and error when exporting to PDF, so it can only be used for HTML.

Submission Instructions

rubric={mechanics:2}

You receive mark for submitting your lab correctly, please follow these instructions:

- Follow the general lab instructions.
- Click here to view a description of the rubrics used to grade the questions.
- Push your `.Rmd` AND **all the files** you will create as part of the lab to your GitHub repository.
  - The reason for pushing all the files is that `.Rmd` does not contain the rendered output from running the cells. If someone is checking out your work there needs to be an HTML file to view the output, so it is good to get into this habit. - `.ipynb` renders nicely on GitHub, which is why we did not include the HTML file for previous labs.
- Upload a `.Rmd` version of your assignment to Gradescope.
- Include a clickable link to your GitHub repo for the lab just below this cell (it should look something like this https://github.ubc.ca/MDS-2022-23/DSCI_521_lab1_yourcwl.

https://github.com/kambojrakesh/rakesh_07_lab04

Editing R Markdown documents

This document is called an R Markdown document. It is a literate code document, similar to Jupyter notebooks where you can write code and view its outputs. To start, let's set our working directory by creating a new R Project for lab 4.

Text and rendering R Markdown documents

In a R Markdown document any line of text not in a code chunk (like this line of text) will be formatted using Markdown. Similar to JupyterLab, you can also use HTML and LaTeX here to do more advanced formatting. To run a code chunk, you can press the green play button in the top right corner of the chunk.

Question 1

rubric={correctness:1}

To render the HTML files we are going to create, the first step is to activate GitHub pages.

https://kambojrakesh.github.io/rakesh_07_lab04/

Question 2

rubric={mechanics:1}

Create a new code chunk below using the r language engine that runs some R code (it does not need to be complicated, but it should have an output). Ensure that you can render/knit the document after you add that chunk.

```
temp <- c(2,50,3,9,8,11,6)
count <- 0
for (val in temp) {
  if(val %% 2 == 0)  {
      count = count+1
  }
}
print(count)
```

```
## [1] 4
```

Question 3

rubric={mechanics:1}

Create a new code chunk, and add a meaningful name to the code chunk. Try using the pop-up-like menu to navigate between the named code chunks Don't forget to knit/render the document after you make this change to ensure everything is still working.

```
temp <- c(2,50,3,9,8,11,6)
count <- 0
for (val in temp) {
  if(val %% 2 == 0)  {
      count = count+1
  }
}
print(count)
```

```
## [1] 4
```

Question 4

rubric={mechanics:1,reasoning:1}

Create a new code chunk that uses a code chunk option. Write out in your own words what the code chunk option is doing.

Code chunk options helps to customize chunk behavior, like a chunk is evaluated, whether we want to include rendered document.

Multiple code chunk options

To have multiple code chunk options you separate them by a comma. For example, if in addition to suppressing warnings, we want to run the code but not output the results, then we can add the `include = FALSE` argument to the code chunk after the `warning = FALSE` option.

include = FALSE prohibits code and results from being included in the final file.

include = False we can temporarily disable the warnings to prevent package loading alerts.

Question 5

rubric={mechanics:1,reasoning:1}

Create a new code chunk that uses at least two code chunk options. At least one must be different to the ones mentioned above. Write in your own words what each code chunk option is doing.

```
num = 10
if((num %% 2) == 0) {
  print(paste(num,"is Even number"))
} else {
  print(paste(num,"is Odd number"))
}
```

```
## [1] "10 is Even number"
```

```
print(paste("Hi","Hello ","World"))
```

```
## [1] "Hi Hello  World"
```

echo : display the code along with the result in knit report.

cache: helps to save the results for the future.

error: knit will not display error generated in final report.

warning: knit not display any warning the knit report.

1.5. YAML Header and document output options

R Markdown files contains three types of content:

1. Plain text mixed with simple Markdown formatting.
2. Code chunks surrounded by ```.
3. An (optional) YAML header surrounded by `---`.

You have been introduced the first two types of content, but not the third (although you probably saw it at the top of this document). The (optional) YAML header, which is located at the very top of R Markdown files sets some general global parameters, including:

- title
- author
- output
- etc

Example YAML Header

```
---
title: "Reproducible Data Science Report"
author: "Florencia D'Andrea"
date: "September 4, 2022"
output: html_document
---
```

Most important from a workflow perspective is **output**. Possible output options include:

- `output: html_document`
- `output: md_document`
- `output: pdf_document`
- `output: word_document`
- `output: xaringan::moon_reader` (xaringan presentation - html)

Question 6

rubric={mechanics:1}

Navigate to the YAML header at the very top of this document and edit it so that you include an `author` (yourself) and a `date` (lab due date). Include what you added below here as well as a fenced Markdown code block.

```
author: Rakesh Kumar and date : September 27, 2022
```

Creating R Markdown documents

You can use the "File" menu inside RStudio to create new R Markdown documents by selecting: **File > New File > R Markdown** This will bring you to another menu where you can choose the type of output (don't be afraid to pick something, you can always change the `output` type once you have the `.Rmd` file).

To create a written report, we generally recommend using the default `output: html_document` as it is easier to read than PDF (note - LaTeX does not render nicely in such documents sadly, so if you are using a lot of LaTeX then you may want to choose `output: pdf_document`). If you want to create an `.md` file to publish on GitHub, it is recommend to instead use `output: github_document`. To get this from the menu above you need to navigate to the "From Template" option on the left panel and then select "GitHub Document (Markdown)".

Question 7

rubric={mechanics:2}

1 - Create a new RMarkdown report (a different file than this one) in the same directory as this RMarkdown file. Use `html_document` as the `output`. After you have rendered it, paste the link to the HTML output as a link to your GitHub repository (remember to push all your files!)

2 - Then, navigate to the YAML header at the very top of that `.Rmd` document and edit it so that the `output` is `pdf_document`. Then knit/render the document. Note the different output. Add and commit that rendered both the `.html` and `.pdf` files to the GitHub repository for this lab and paste the two links below this question.

https://github.com/kambojrakesh/rakesh_07_lab04/blob/main/First_Markdown_document_Q7.1.html

https://github.com/kambojrakesh/rakesh_07_lab04/blob/main/First_Markdown_document_Q7.2.html

https://github.com/kambojrakesh/rakesh_07_lab04/blob/main/First_Markdown_document_Q7.2.pdf

Question 8

rubric={mechanics:6}

1. Go back to the `.Rmd` file you created in question 7, and include at least two Markdown text sections (each should have a header) and at least two separate code chunks in it (these can be really simple). Save the new R Markdown document and give it a new meaningful name.
2. Render/knit the new R Markdown document to get an `.html` file. Put the `.Rmd` document and the rendered `.html` file under version control using Git, and push/upload the file to your GitHub repository for this homework. Paste a link to these files as your answer below.

https://github.com/kambojrakesh/rakesh_07_lab04/blob/main/Code_Check_Markdown_Q8.1.pdf

https://github.com/kambojrakesh/rakesh_07_lab04/blob/main/Code_Check_Markdown_Q8.1.html

https://github.com/kambojrakesh/rakesh_07_lab04/blob/main/Code_Check_Markdown_Q8.1.Rmd

Question 9 (Optional)

rubric={mechanics:1,reasoning:1}

1. Take the R Markdown report created in Question 8 and change the output to `github_document` and render it. Put the rendered `.md` file under version control using Git, and push/upload the file to your GitHub repository for this homework. Try to look at the file on GitHub.ubc.ca in your homework repo? What do you see? How is it rendered?

YOUR ANSWER GOES HERE

Question 10

rubric={mechanics:6}

1. Create a presentation using RStudio. Do this in a different file than this one but in the same directory as this RMarkdown file. You can use xaringan or Quarto to create the slides. On the book you will find links that will guide you on how to create each type of slide. If you are not sure which one to use, it is safer to use Xaringan as Quarto is quite new and you will have to learn how to use `.qmd` files. But if you want to explore Quarto, we will accept both options (Xaringan or Quarto slides) as correct. Give this file a meaningful name.
2. Create at least 4 slides. At least two slides must include a code chunk or cell (these can be really simple). Save the new document.
3. Render/knit the new document to get a `html` presentation file.
4. Put the new document and the rendered `.html` file under version control using Git, and push/upload the file to your GitHub repository for this lab
5. Activate GitHub pages and paste the link below.

https://kambojrakesh.github.io/rakesh_07_lab04/Markdown_Presentation_Q10.html#1

(Challenging) Question 11

rubric={reasoning}

In a paragraph or two, compare and contrast the use of reproducible tools (e.g., R Markdown and Jupyter) and non-reproducible tools (Word, Powerpoint, Keynote, etc) for presentations and reports. Include advantages and disadvantages for each.

Reports and presentations play a very vital role in the Data Science field. Different tools like R Markdown, Jupyter, Word, and PowerPoint can be used for the purpose. Both of these types have there on pros and cons.

The report or presentation must have the following points what we want to analyze, what happened, and what we should do about it.

Reproducible tools exist to develop open-source software, open standards, and services for interactive computing across dozens of programming languages. These tools are mainly ready to get started. We can try some of this software via browser as well.

Non-reproducible tools enable us to produce elegant slideshow presentations, sophisticated pitch decks, and a robust presentation creator to communicate your message.

Advantages and Disadvantages of reproducible tools

Reproducible tools are convenient when we do not have a straightforward end procedure and are still in the prototype phase of our automated workflow. The disadvantages of these tools are that they take long asynchronous jobs and are challenging to test. We have to perform operations carefully, and proficiency in software is required.

Advantages and Disadvantages of non-reproducible tools

Non-reproducible tools make it simple to communicate to a big audience while maintaining eye contact by simply forwarding the slides with a keystroke and doing away with the requirement for handouts to help the audience understand the information. The main disadvantage like too many features can get overwhelming and most parts usually remain unknown.