Documenting your data analysis using R Markdown

A Michelle Edwards, Ph.D., MLIS

May 5, 2020

Table of Contents

Workshop Overview

This is an introductory R workshop:

- That will focus on using RMarkdown to document your analysis
- R and SAS analyses

Learning environment:

- Let's learn and have fun!
- If you have questions, please ask them

Learning Outcomes

By the end of this workshop you will be able to:

- Be able to create your own RMarkdown and Word document

Setting up our R Studio

Let's first set up our working directory. You have a number of options here:

1. Go to Session -> Set Working Directory -> then navigate to the location on your laptop you would like to use as your working directory for this session
2. Go to Files on the right hand side of your screen -> navigate to the folder you would like to work from -> then click on the More icon and select Set As Working Directory
3. Type the setwd command as shown below - remember that your slashes are the opposite direction or you need to type 2: /Workshops/R/W20 is the same as \Workshops\R\W20

```
setwd("~/Workshops/R/S20/R_Markdown")
```

Let's also install - if required the following packages:

- **rmarkdown** - to create our finished documents

To install a new package:

1. Using code: install.package("lme4")
2. Using the menus - Tools -> Install Packages -> Enter the name of the package in the dialogue box -> ensure that the install dependencies is checked

Once you have the packages installed, we need to load them for our session. Two different ways to do this as well:

1. In R Studio - right hand bottom box - go to the Packages tab - scroll down to the package you want to load - put a check in the box next to it.
2. In the script window - using code - library()

Exercise 1

1. Install the **rmarkdown** package
2. Load the packages.

```
library(rmarkdown)
```

Getting started with RMarkdown

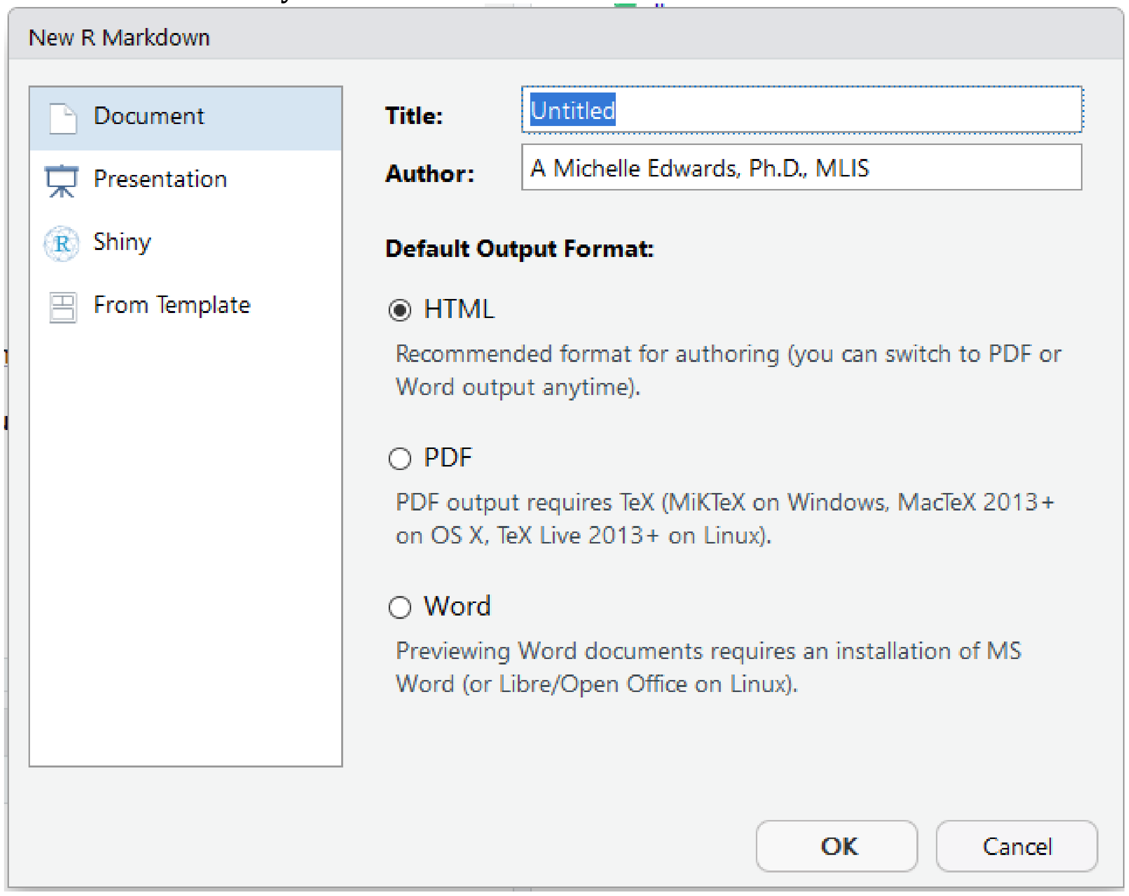
Available resources:

- Cheatsheets: (Help -> Cheatsheets)
  - R Markdown Cheatsheet
  - R Markdown Reference Guide
- Online documentation - see R Markdown Quick Tour from RStudio
- R Markdown: The Definitive Guide written by Yihui Xie, J.J. ALlaire, and Garrett Grolemund - the creators of R Markdown - I have a copy if you'd like to see it
- R community

Create First R Markdown Document

Open a new RMD file. File -> New File -> R Markdown…

This will open a new window in the Editor space. You will be asked a few bits of information to start your file.


New R Markdown

Title: Untitled

Author: A Michelle Edwards, Ph.D., MLIS

Default Output Format:

[✓ HTML] PDF Word

HTML: Recommended format for authoring (you can switch to PDF or Word output anytime).

PDF: PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

Word: Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

Let's start by giving our document a Title - your choice! Fill in the Author box. Let's work with a Word document to start. Take note of the choices you have at your diposal. I prefer to start with a Word document to take advantage of the Table of Contents option - then I will save my Word as a PDF. Please try the different options on your own.

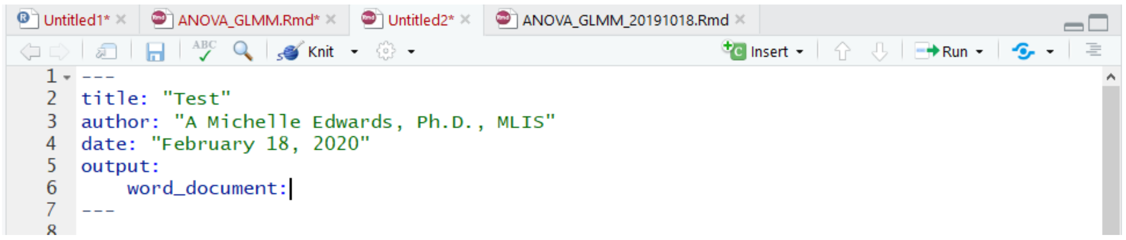
Once you click OK - you will be presented with a new window and the information you filled in will be visible. The only thing that is missing to start is saving your new RMD (R markdown) file. So let's do that first. Remember the file will be saved in your Working Directory that you set at the beginning of this session.

Format or Layout of the RMD file.

Notice that there are bits of code in the RMD file you just saved. These are examples to showcase what you can do with this type of file. You'll notice that there are greyed out areas that contain R syntax and everything else is where you will add your own content. The information at the top of the file is the same information you filled in the opening dialogue box. You can make changes to this at any time. Notice that the date is automatically filled in.

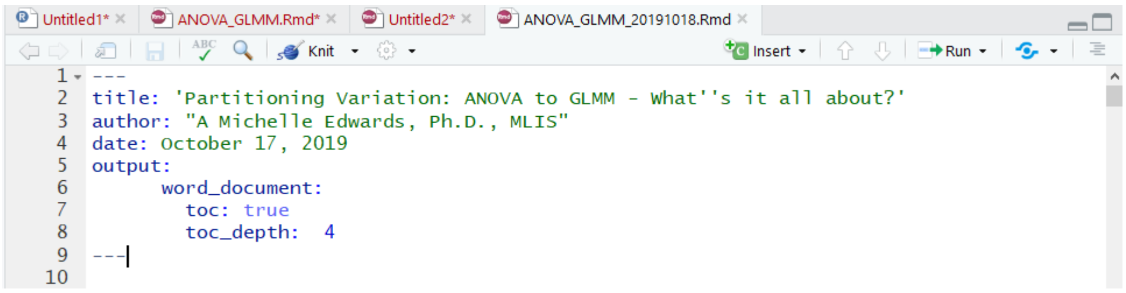
The output type that we selected is Word. Let's chat about some of the options you have here before we start writing code and creating our Word document. I mentioned earlier that I prefer the Word option because I can take advantage of the TOC option. If you look carefully at this document, you will see that I used R Markdown to create it :) I created headings as I was writing this document to provide sections to organize the file. These headings are what RMD uses to create the Table of Contents in the Word document.

First step is to say we want to have a TOC (Table of Contents). To do this, scroll to the top section of your RMD file and hit enter after the output: This will cause the word_document to move to the next line. Add a ":" after word_document. Then hit your TAB key 2 times so your information looks like this



```
---
title: "Test"
author: "A Michelle Edwards, Ph.D., MLIS"
date: "February 18, 2020"
output:
    word_document:
---
```

On the next line we tell R that we want a toc - by saying TRUE and on the following line we tell it how many levels, by using the toc_depth: option.



```
---
title: 'Partitioning Variation: ANOVA to GLMM - What''s it all about?'
author: "A Michelle Edwards, Ph.D., MLIS"
date: October 17, 2019
output:
      word_document:
        toc: true
        toc_depth:  4
---
```

Now we've set up the environment for our Word document

Creating our first Word document

Before we talk about how we add information into our RMD file, let's run the example to see how we create the Word document and what it looks like. Now, I've been using R Markdown for a while - so I'm hoping that using only the **rmarkdown** package will work - Fingers crossed please!

To create the Word document, R uses a process called KNIT - for more information on this process, please review the video on R Studio - the link was provided in the section above. If you look just above your editor window you should see an icon that looks like a ball of wool with needles in it? Let's take a quick little tour into the option. If you click the little arrow beside it, you should see a few options - Knit to HTML, Knit to PDf, Knit to Word, and more. Because we filled in the dialogue box and the top of our file information is complete, R already knows we want to create a Word document. So by clicking on the icon, it will automatically create a Word document.

Exercise 2

1. Click on the Knit Icon to try it out!

If you are asked to save the file - please give it a name to save. You should only have to do this the first time you run the file.

Let's take a few minutes to debug if needed….

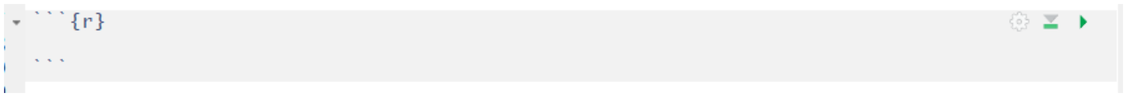
Did you notice the R Markdown below the editor window? It showed you what was happening as it created the document.

Editing the R Markdown Document

So now that you were able to run the example, take a closer look at the RMD file and the matching Word document. Notice that the RMD file only has the R script - but when you created the Word document - you now see the R script along with the output it creates. This is the beauty of using R Markdown.

R Chunks

The greyed out areas in the RMD file are referred to as R chunks. In order for these areas to work you need three quotes and {r} code to be at the top and the matching three quotes at the bottom. This lets RMD know that this is an R script that needs to be run. You can copy or type in any R script in these areas. If you are unsure whether it works or not, you can run the R chunk without running or knitting the RMD document together. To do this, you click on the green arrow at the corner of the R chunk



~~~
```{r}
```
~~~

Naming the R Chunks

It might be handy to name your R Chunks or pieces of R script. This takes time - but remember the goal of this exercise is to document your analysis. So, let's add a name to an upcoming R chunk, where we will set our Working Directory. To name the chunk you will type the name after the {r workd_dir} as an example

```
setwd("~/Workshops/R/S20")
```

Note that you will NOT see it in your Word document but it will be in your RMD and as it is running you would have seen in the R Markdown log.

Markdown options for Word, HTML, PDF

R Markdown uses different codes and combinations of characters to create options in your Word document. Best way to see some of these is to review the Cheatsheet. Trust me, I have it next to me every time I create a new RMD file.

Let's open the Cheatsheet and review some of the more common ones that may be used:

- Headings
- Ordered and unordered lists (bullets)
- Bold
- Italics
- Adding a link
- Adding an image

This document uses all of these features.

So what are you documenting?

The beauty of R Markdown is that everything is in one file. You have your R script and the matching output. All you need to do now is to add your thoughts, notes, description, interpretation to the document. You might write about what the upcoming R script is going to do - what the analysis is - what model you are using, etc.. Then after the R script you may discuss what you are seeing in the output. If you want to see what the output looks like BEFORE you create the Word document, remember that you can run the R chunk by clicking on the green arrow.

Let's use a test file in R and run a summary() function on it. Add a new R chunk by going to Insert and selecting R

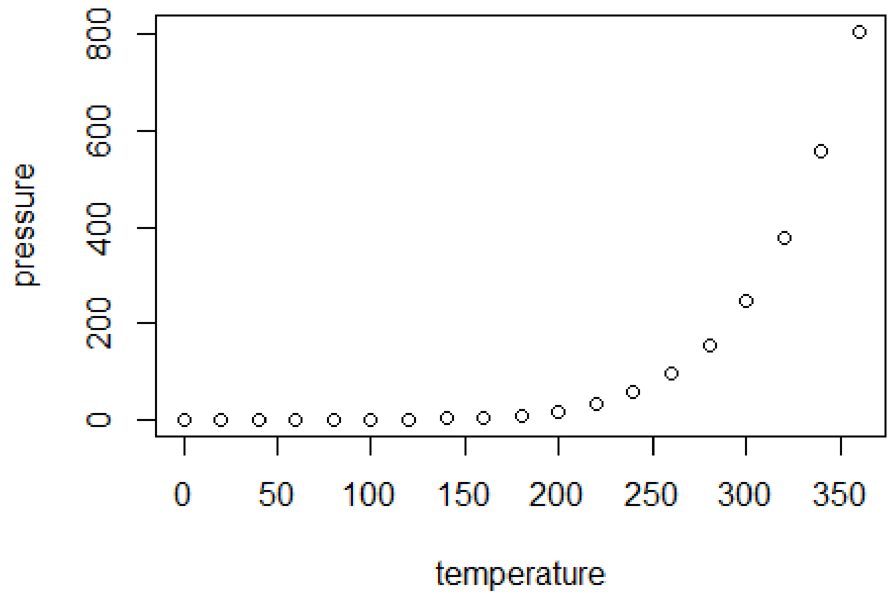
You can add a name to this R chunk by typing a name after r - type a space first and then your name. Let's call this one *cars*. On the next line - let's type in the R function summary(cars)

```r
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



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

You can see how you can write a blurb about the analysis you are conducting, then RUN the analysis, and write another blurb after the output as your interpretation. Pretty cool eh???

TO make your document pretty or ugly - use the Cheatsheet to guide you. Once you've created your document as a Word, you can then continue to make format changes in Word. Then save it all as a PDF.

This makes your research replicable - the only thin missing is the data. But you can point to the source of the data.

SAS???

Yes! YOu can even use this for SAS analysis - **HOWEVER!!!** YOu can only use it with the licensed version or SAS 9.4 I'm hoping and I'll keep an eye out for updates on how to use this with SAS Studio. The other "gotcha" with using it with SAS - you need to think about the code we will run here as if you were running the code on a central server. FOr those of you in the ABS department - you are familiar with this. This also means that the fancy plots are not available. But let's take a look.

First you will need to install and load the **SASmarkdown package**

```
library(SASmarkdown)

## SAS found at C:/Program Files/SASHome/SASFoundation/9.4/sas.exe

## sas, saslog, sashtml, sashtml5, and sashtmllog & sashtml5log engines

##    are now ready to use.
```

A small example

Just like we did with R - let’s use a dataset that is included with the SAS program. sashelp.class We are going to run a Proc Means on the dataset.

```
Proc means data=sashelp.class;
Run;

                            The MEANS Procedure

 Variable    N            Mean         Std Dev         Minimum         Maximum
 -----------------------------------------------------------------------------
 Age        19      13.3157895       1.4926722      11.0000000      16.0000000
 Height     19      62.3368421       5.1270752      51.3000000      72.0000000
 Weight     19     100.0263158      22.7739335      50.5000000     150.0000000
 -----------------------------------------------------------------------------
```

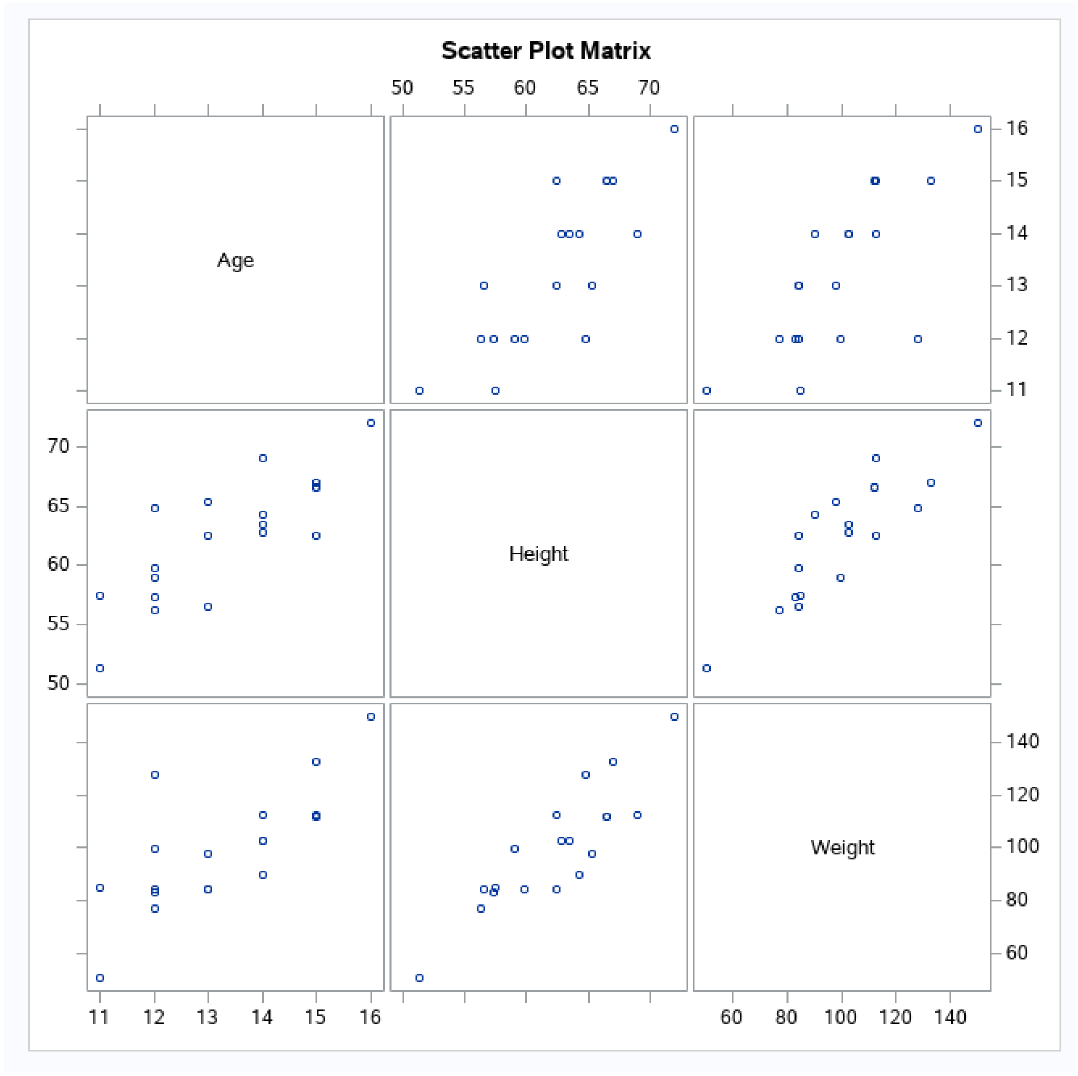
Output is very similar to what we see when we run it on our systems. However, some folks might notice the old courier type font? Take you back? This is happening because we are using the server-side of our SAS, no ODS and no fancy formatting.

Let’s try another example of code. A Correlation with some plots.

```

Proc corr data=sashelp.class plots=matrix;
run;
                            The CORR Procedure

                 3  Variables:    Age      Height   Weight

                            Simple Statistics

      Variable           N          Mean       Std Dev           Sum

      Age               19      13.31579       1.49267     253.00000
      Height            19      62.33684       5.12708          1184
      Weight            19     100.02632      22.77393          1901

                            Simple Statistics
```

```
               Variable       Minimum       Maximum

               Age           11.00000      16.00000
               Height        51.30000      72.00000
               Weight        50.50000     150.00000

            Pearson Correlation Coefficients, N = 19
                    Prob > |r| under H0: Rho=0

                          Age        Height        Weight

         Age          1.00000       0.81143       0.74089
                                     <.0001        0.0003

         Height       0.81143       1.00000       0.87779
                       <.0001                      <.0001

         Weight       0.74089       0.87779       1.00000
                       0.0003        <.0001
```

So - what happened??? If I run this in SAS Studio or on my SAS 9.4 - this is what I would get



Remember because we are using the SAS server - we do not have access to the ODS outputs and many of the newer plotting features.

Trying to link SAS code chunks

The 2 examples we just saw are calling the data with the analysis AND they are internal SAS datasets. What if we want to use our own data? Or make changes to the help dataset and save it locally - aha! Yes! There's a catch! Let's look at this DATA Step to start. We are creating a new dataset called class which will be a temporary dateset and should be saved locally - but there is no local since we are running the SAS code "on the fly". So in SASRMarkdown we are telling it to save this code and the results - or collectcode.

```
Data class;
  set sashelp.class;
  bmi = 703*weight/height**2;
Run;
```

To use this new dataset, in our R chunk we use the PROCstep function, which tells R to remember the last set of SAS commands and use it to run the current code.

```
Proc means data=class;
  var bmi;
Run;
                            The MEANS Procedure

                         Analysis Variable : bmi

     N             Mean          Std Dev          Minimum          Maximum
    ------------------------------------------------------------------
    19       17.8632519        2.0926193       13.4900007       21.4296601
    ------------------------------------------------------------------
```

Other ways to document your analysis????

Adding comments in your SAS syntax or your R script. This again can be challenging, take a lot of time - but it is worth the time! Adding a comment to your code to explain why you are running the analysis - what your outcome measure is, where it comes from, any manipulations you did, is very helpful. Remember we want to make our studies reproducible - we want to ensure the next individual that uses your data, can reproduce your results. Without information about your data and your analysis this will be next to impossible.

Comments in R

To add a comment in an R script we use the # sign. Let's repeat some code from before, but this time we'll add a comment to it.

```
# I'm using the S20 directory in my R Workshops as my working directory
setwd("~/Workshops/R/S20")
```

Notice in the R script area it is green type and in the Word document it is bronw and italicized. There is no limit to how many comments you can add in your script file. So feel free to document the story of your data analysis in your script file.

Comments in SAS

Very much like R, we can also add comments to our SAS code. In this case though to add a comment we have 2 options. To write 1 line of comment we can simply start with an * and end the line with a ;

Let's try it out:

```
* Running some descriptive statistics;
Proc means data=sashelp.class;
Run;
```

```
                            The MEANS Procedure

 Variable    N           Mean        Std Dev        Minimum        Maximum
 ------------------------------------------------------------------------
 Age        19     13.3157895      1.4926722     11.0000000     16.0000000
 Height     19     62.3368421      5.1270752     51.3000000     72.0000000
 Weight     19    100.0263158     22.7739335     50.5000000    150.0000000
 ------------------------------------------------------------------------
```

If we want to add a bit more, we can add a paragraph of information. We can do the same as above for each line OR we can start our paragraph with a /* and end the paragraph with a */

Let's see:

```
/* Running some descriptive statistics - going to write a paragraph to show h
ow to add the comment
  in SAS  */
Proc means data=sashelp.class;
Run;

                            The MEANS Procedure

 Variable    N           Mean        Std Dev        Minimum        Maximum
 ------------------------------------------------------------------------
 Age        19     13.3157895      1.4926722     11.0000000     16.0000000
 Height     19     62.3368421      5.1270752     51.3000000     72.0000000
 Weight     19    100.0263158     22.7739335     50.5000000    150.0000000
 ------------------------------------------------------------------------
```

Conclusion

Documenting your data analysis can be done using something like RMarkdown , which allows you to build the story before, during, and after the analysis- all in one document.

You can also begin to document your analysis by adding comments to your code.