1. R MARKDOWN BASICS

Markdown languages.

Markdown is a lightweight computer language for creating formatted documents. Instead of clicking on icons and selecting features from pull down menus, the desired formatting is typed into the document and integrated with content. Formatting options like bold, italics and underline are specified with special characters that are typed before and after regular text. When the document is "knitted" (AKA processed, parsed, rendered), the formatting commands are executed and then hidden from view. The knitted document typically takes the form of .html, .pdf or .docx and is designed to be more polished and readable than the Markdown document itself.

R Markdown is one of many variations (more like a dialect) on the Markdown language. R Markdown files allow a user to write Markdown text (like what you are reading right now) and also use any of several programming languages (R, Python, SQL, etc) to analyze data all in the same document. The Markdown text is generally used to explain and interpret the analysis, while the programming languages are used to manipulate and analyze data. R Markdown files include a third language called YAML that determines some of the final knitted document characteristics, such as whether it will be a webpage (.html) a flat file (.pdf) or a MS Word document (.docx).

R Markdown files - 3 Parts

R Markdown files are pre-structured into three parts:

1) A YAML header Located at the top of the file and between two sets of three dashes. The YAML header has metadata that determines how the Markdown document will be rendered or "knitted" into a polished final document. If you are viewing the .Rmd file, scroll to the top to see the YAML header with it's characteristic "key: value" organization. Because RStudio structures the YAML header when you open a new .Rmd file, you will not need to learn YAML or make changes to the YAML header.

2) Markdown space (white background) New R Markdown files contain pre-formatted text in markdown space that helps to explain the features of the markdown file. The text you are reading now is written in markdown space. As you can see, the markdown language allows you to write in mostly plain English.

3) Code chunks (gray background) Code chunks interrupt the markdown space and are areas where you can write code using R, Python, SQL and other languages. This is where the business of statistics happens! New R Markdown files come pre-formatted with several code chunks that help demonstrate the R language and functionality of an R Markdown file.

Many R textbooks (and the guide you are reading now) are written entirely with R Markdown files, because of the ability to display code, tables, graphs and other types of code output, along with formatted explanations of the code. *R Markdown: The Definitive Guide* is one of many text books written entirely using R Markdown files. The source code for the book can be downloaded from this github link.

Code chunks

The setup code chunk

The first code chunk is the r setup code chunk shown in figure 1 below:

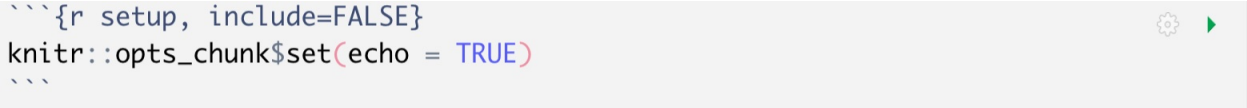

Figure 1: setup code chunk

The "setup" statement inside curly brackets relates to options that we will not utilize in this course and is only present in the first code chunk. Everything else about this code chunk influences the appearance of our document after it is knitted. Unfortunately, this first code chunk is somewhat tedious to explain, but here it goes. . .

All code chunks begin and end with three back tick marks. These back ticks tell RStudio to stop reading R Markdown text and start reading in the programming language that is specified inside the curly brackets, "r" in this case.

The "include=FALSE" statement means that the code inside this particular code chunk will not appear in the knitted document. In fact, if you are reading this material in the .Rmd file, now is a great time to click the "Knit" button above this panel. You may need to wait several seconds for the file to be rendered. Once the knit is completed, periodically compare the knitted document to the .Rmd file one as you continue working through the material.

The single command in the middle of the code chunk will establish a default behavior for all code chunks that follow. In this case that behavior, is to show R code in the knitted document. Sometimes we change this argument to "FALSE" when we want our audience to ignore the code and only see the output and our explanations written in markdown. The double colon operator, "::", is a special command for reaching into software packages and grabbing commands. In this case, the :: is used to reach into the software package called knitr and grab the command called "opts_chunk".

Adding new code chunks

Code chunks are used to organize code into sections analogous to paragraphs of an essay. When we wish to add one or more lines of code that work together, we set our cursor to the desired location in markdown space and then add a code chunk by one of several methods shown below:

1. Key shortcut (quickest): Simultaneously press "option + shift" + i".
2. Code chunk pull down menu: select "R" from the green and white +C icon.
3. Code menu, select "Insert Chunk" from the Code menu at the top left of your screen.

Practice Add a new code chunk below this line using the key short cut method. Type <print("Hello World")>, (but without < and >) on the blank line of code. With the cursor anywhere on this line of code press "command/control" + "Enter"

R Packages

Software packages like knitr provide commands that extend the functionality of R. These packages are created and donated by R users all over the world. R packages must be installed once on your personal computer, just like any other software, and then loaded within an R Markdown document before they can be used. Newer versions of R Studio are capable of detecting packages that are needed but not installed on your

computer. If you are reading this in R Studio, check above the code panel in R Studio for a message related to uninstalled packages and click "Install" if the option exists.

The knitr package ("knit r") provides commands that help customize the appearance of a knitted document. In the setup code chunk, we used the opts_chunk command from knitr to set the default echo parameter to TRUE. This means that the code we write inside of subsequent code chunks will also appear in our knitted document. However, the "include = FALSE" command inside the curly brackets overrides the default for any particular code chunk, meaning we will not see setup code chunk echoed in the knitted document and we can hide other code chunks if we wish.

The knitr package (like all packages) has many other commands as well. Sometimes the commands inside of packages have conflicts with base R commands or other packages. The setup chunk in new R Markdown file uses "::" to avoid these potential conflicts by pulling the opts_chunk command out of the knitr package, without actually loading the entire knitr package.

As it turns out, the knitr package has many useful commands and few (if any) conflicts for basic statistical analyses. As such, I recommend changing the setup code chunk to load the knitr package and then use the opts_chunk command on a separate line. Let's recreate this code chunk below and eliminate "include = FALSE" from the curly braces, so that we see this code chunk in the knitted document:

```
library(knitr) # The library command is used to load individual packages.
opts_chunk$set(echo = TRUE)
```

Note that this code chunk is missing "setup" in the curly braces. Only one code chunk can function as the setup code chunk, so we must omit this component.

Loading multiple packages (library commands)

Often we know in advance that you we will use certain packages in our analysis and choose to load those packages in the setup code chunk. All of the packages loaded below must have already been installed at some point before the library commands can run without error.

```
library(knitr)
opts_chunk$set(echo = TRUE)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(kableExtra)
```

```
## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 4 > 1' in coercion to 'logical(1)'
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

Notice from the output, that when a package is first loaded R will display messages indicating if the load was successful, if conflicts exist and if other commands are masked by commands in the new package.

Loading multiple packages (p_load)

Because writing multiple library commands is a bit tedious, I often use p_load from the pacman package to simultaneously load multiple packages. Also, to avoid showing messages and warnings that R produces when packages are loaded, use "message=FALSE" and "warnings = FALSE" in the set command as shown below.

```
pacman::p_load(knitr, tidyverse, kableExtra)
opts_chunk$set(echo = TRUE, message=FALSE, warnings = FALSE)
```

Markdown formatting

You can write *mostly* plan text in an R Markdown document. However, there are some basic ways to format text for a more polished appearance. The more polished appearance is created when the Markdown file is rendered or "knitted". For example, you can use asterisks to create bold or italic font in the knitted document as shown below:

italics
bold
bold and italics

Note that you will not see the formatting applied until the the document is knitted, or unless you are using the RStudio visual editor, which you should avoid for now. Markdown has a few quirky formatting features. Leaving a blank line on a Markdown document will also leave a blank line in a knitted document. However, without that blank line, text on sequential lines will run together, unless either two spaces or a backslash are left at the end of a line. The three lines of bold text below have neither two spaces nor a backslash at the end of the line and will run together when the document is knitted.

this text will run together with this text on one line in the knitted document

Hashtag Features in R Markdown space

Hashtags followed by a space have three important and related features in R Markdown:

Feature 1 - Collapsible document sections Hashtags followed by a space create collapsible sections of the R Markdown file. Notice the small gray arrow to the left of the hashtags above. Click on the arrow a few times to see what happens. You'll notice that large sections of the Markdown document collapse into a single line, leaving only the text next to the hashtag visible as a header. Sections with more hashtags become nested within sections with fewer hashtags. Essentially, the hashtags allow an outline of headers to be built into the R Markdown file.

Feature 2 - R Studio outline You can imagine that when an R Markdown file grows to hundreds of lines, getting lost in these documents happens easily. R Studio provides a handy outline feature based on hashtag headers in Markdown space. Click on the Outline button in the upper right of this panel and expand the window that opens. Every text line you see corresponds to a place in the Markdown document where a hashtag header has been created. Clicking within this outline takes you to that part of the document. Click around a bit, but then come back to this section and close the outline if you need monitor space.

Feature 3 - Knitted document headers Hashtag headers appear in larger, darker font in the knitted document. One hashtag creates the largest header (like for "1 MARKDOWN BASICS"). You'll have to knit the document to see these features. If you haven't done so already, knit this document by clicking on "Knit" above this panel.

Important Tip The space after the hashtag is critical! The text will not be formatted as a header in the knitted document without that space. Notice how the text below appears in the knitted document.

####Not a header because a space is missing after the hashtags.

Use headers to organize your document!!!

Think of a .Rmd file as an instruction manual or text book with big units indicated by a single hashtag, chapters by two hashtags, chapter sections by three hashtags, etc.

More Markdown Features

Download the [R Markdown Cheat Sheet] (https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf) for more markdown features.

Knitting

Before you can knit an R Markdown file, you will need to have installed all the packages used in library commands (knitr, tidyverse, pacman, magrittr, kableExtra). You will need to have an internet connection to reach a CRAN mirror where these packages are hosted and made available for installation. Use the "Packages" menu in the bottom right panel to install packages, or use install.packages("package_name") in the R console. Again, packages ONLY need to be installed once (like any other software) on a computer. Do NOT include install.package() commands inside an R Markdown chunk. This is pointless and time consuming.

If you have installed those packages AND any packages that they depend on, click on the Knit menu and choose the HTML or PDF options. View the document that is created. If the knit fails, read the error messages and try to fix the problem. Often this means installing additional packages called dependencies. Seek a TA for help if necessary.