

# R Markdown Tutorial

## 1. Introduction

R Markdown allows you to take your R code and create documents in Word, HTML, or PDF formats that fit your data analysis and reporting needs. R Markdown is a “text-markup” language, which allows you to create documents with headers, links, images etc. from a plain text file in .Rmd format, where ‘Rmd’ stands for ‘read meta data.’ It is fast becoming a standard for writers who want to focus on creating documents—not worrying about proprietary document formats.

This document is meant as a very brief introduction of R Markdown - you can find more detailed information about R Markdown at: [R Markdown from R Studio](#).

## 2. Installation

Like the rest of R, R Markdown is free and open source. You can install the R Markdown package from the Comprehensive R Archive Network ([CRAN](#)) with writing and executing the following code in an R script or the console pane:

```
install.packages("rmarkdown")
```

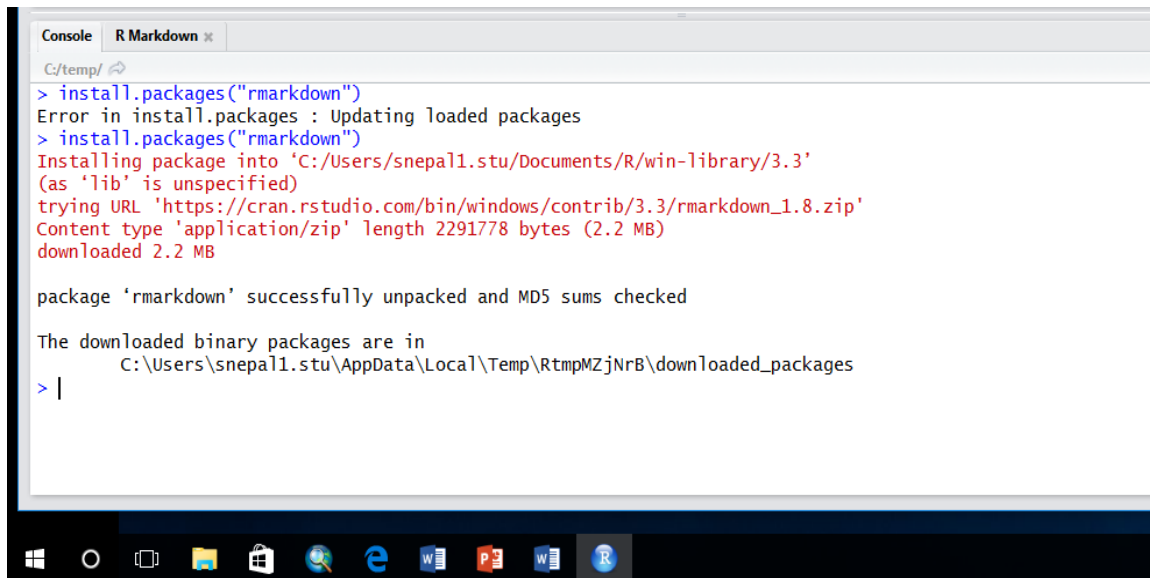
The image shows a screenshot of the R Studio console window. The window title is "Console | R Markdown x". The current directory is "C:/temp/". The user has entered the command `> install.packages("rmarkdown")`. The console output shows an error: "Error in install.packages : Updating loaded packages". The user then enters `> install.packages("rmarkdown")` again. The output shows the package is being installed into the user's library: "Installing package into 'C:/Users/snepa11.stu/Documents/R/win-library/3.3' (as 'lib' is unspecified)". It then shows the download process: "trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.3/rmarkdown\_1.8.zip'", "Content type 'application/zip' length 2291778 bytes (2.2 MB)", and "downloaded 2.2 MB". Finally, it shows "package 'rmarkdown' successfully unpacked and MD5 sums checked" and "The downloaded binary packages are in C:\Users\snepa11.stu\AppData\Local\Temp\RtmpMZjNrB\downloaded\_packages". The console ends with a prompt `> |`. The Windows taskbar is visible at the bottom of the screenshot.

Figure 1: Example code to install R Markdown in R Studio.

**Starting a new R Markdown project (see Figure 2):**

Go to the top left corner of the R window and go through the following steps.

File > New File > R Markdown.

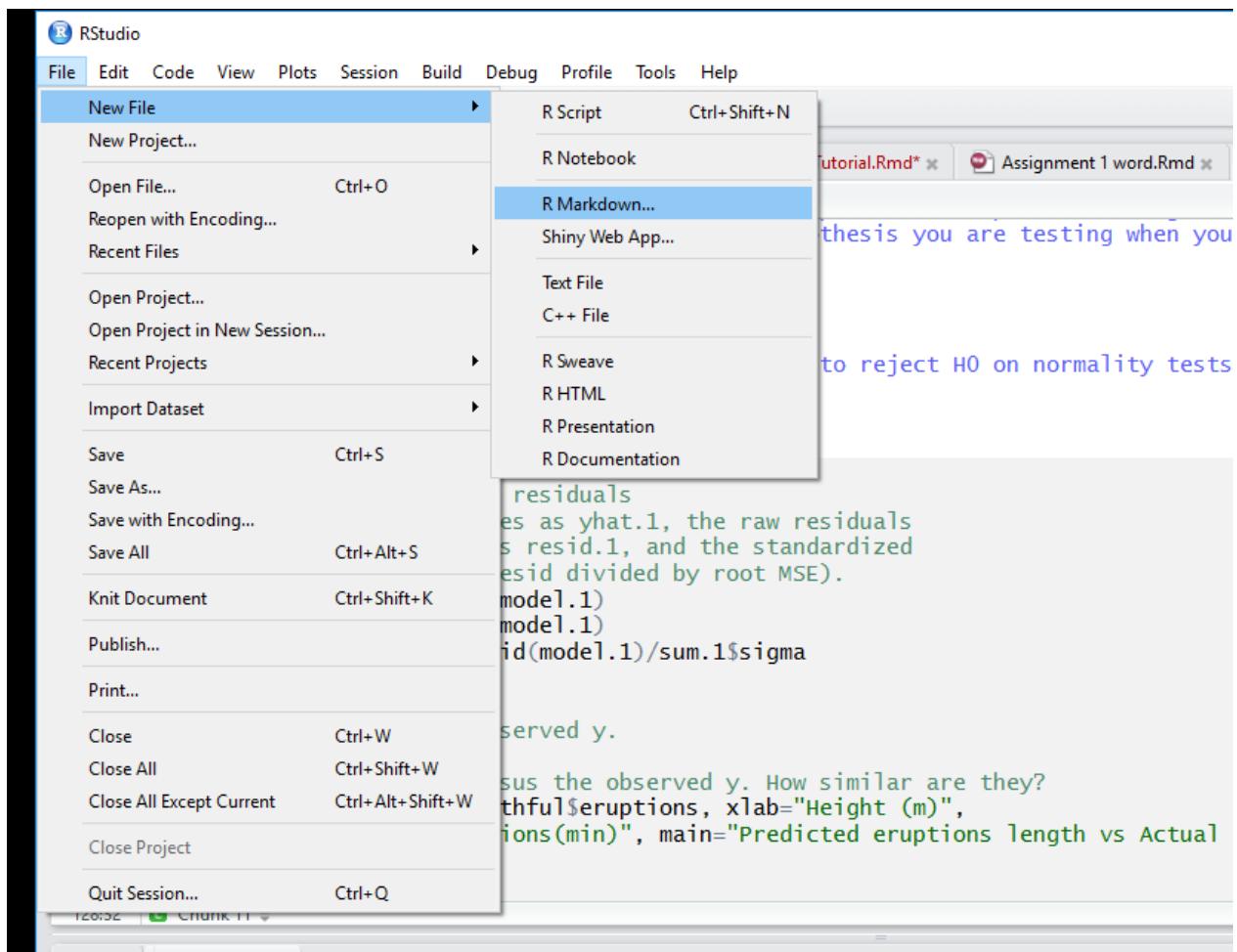


Figure 2: Starting up a new R Markdown project.

A new dialogue will appear that will allow you to fill out the header of the document you are about to create. In this window, you can specify the following (Figure 3):

- Your name (Author)
- Title of the document
- Format of output file (Word, PDF or HTML) – NOTE: For all FRST 232 assignments, you will choose the option “Word.”

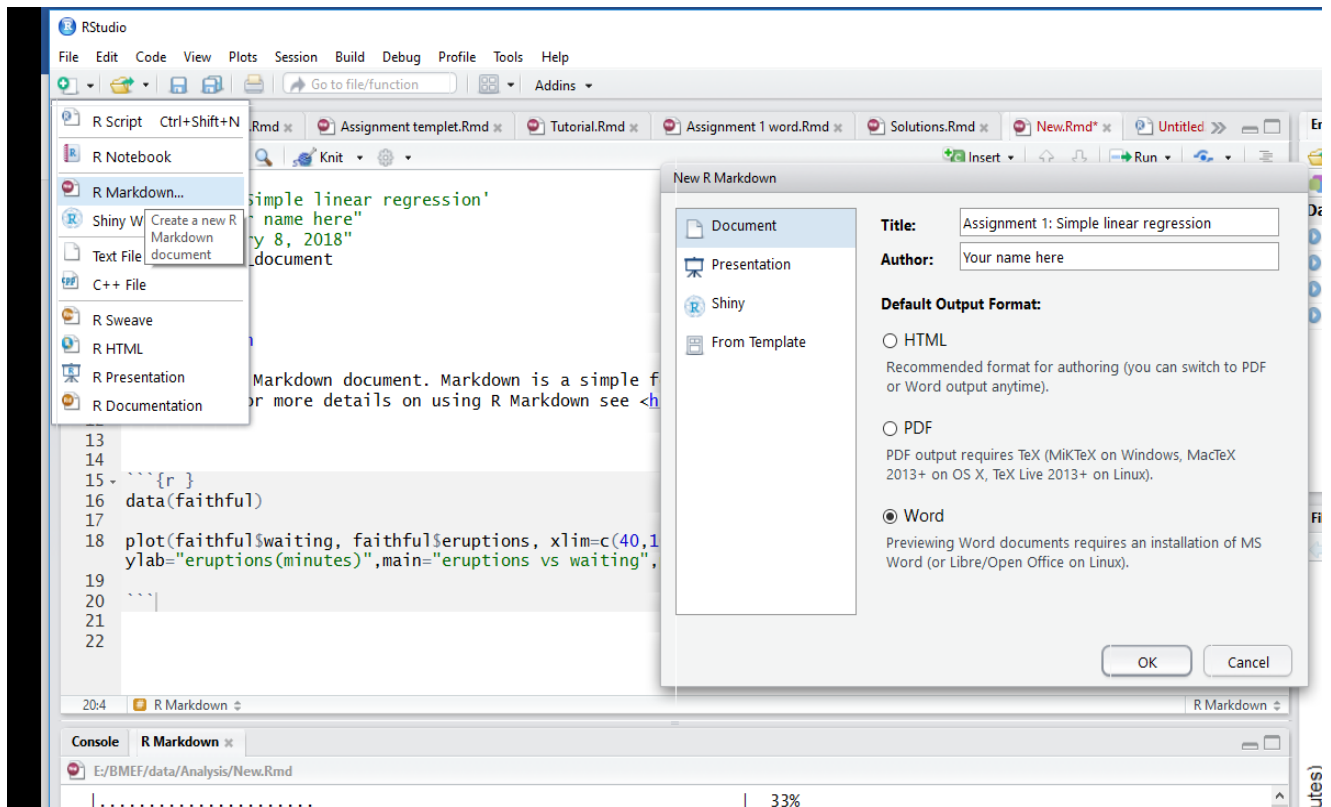


Figure 3: Creating an R Markdown Word document.

### Structure of an R Markdown file

The R Markdown file can be divided into the following parts (Figure 4):

- Front matter: Titles, author name, date for the documents created and the information on the type of document that has been created comes under this section.
- R Markdown: Any comments you want to make or any write-up about the documents can be done in this section.
- R code: Any code you want to execute and process will be included in this section.

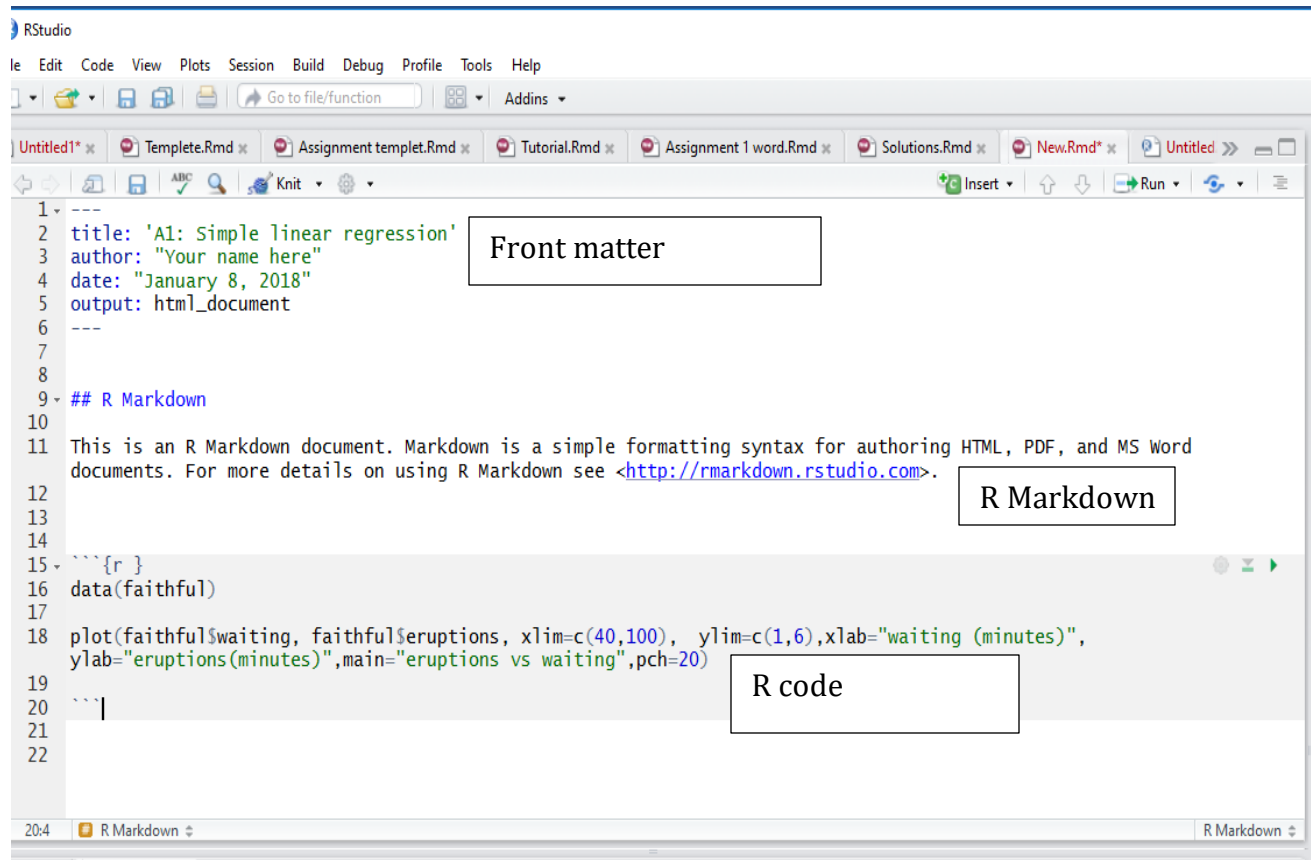


Figure 4: Division of R Markdown file.

### 3. Inserting code

Code in your .Rmd document begins with three backwards apostrophes ````` followed by `r` embedded within curly brackets and is closed by three back apostrophes at the end of the code.

For example:

```

```{r}
data(faithful)

plot(faithful$waiting, faithful$eruptions,
xlim=c(40,100), ylim=c(1,6),xlab="waiting (minutes)",
ylab="eruptions (minutes)",main="eruptions vs waiting",
pch=20)
```

```

## 4. Generating documents

When you click the Knit button (Figure 5), a document is generated that includes both content and output of any embedded R code within the created document (Word, PDF or HTML).

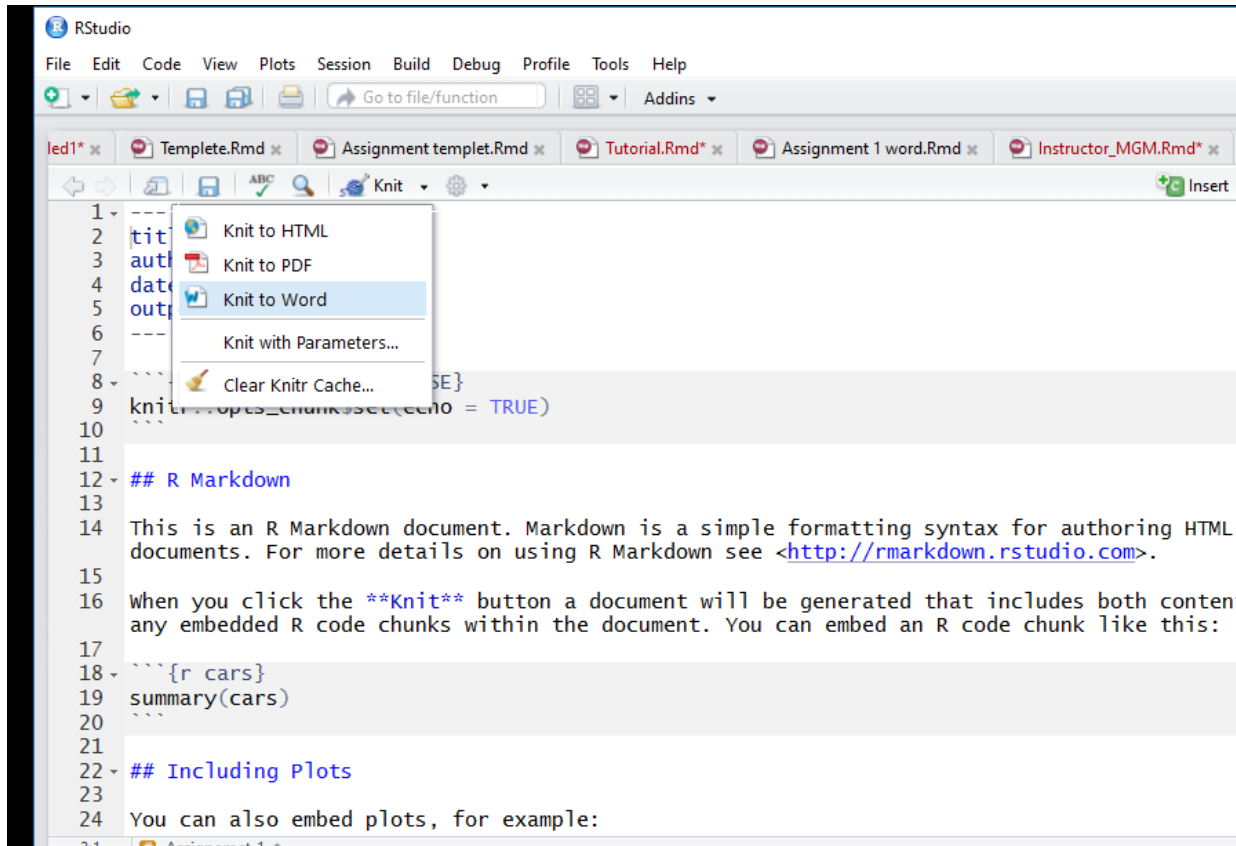


Figure 5: Generating HTML, PDF or Word documents.

You can generate documents in either MS Word, PDF or HTML format. An example for HTML format is given in Figure 6. You can change the format to Word or PDF depending on your requirements through the dropdown menu shown in Figure 5.

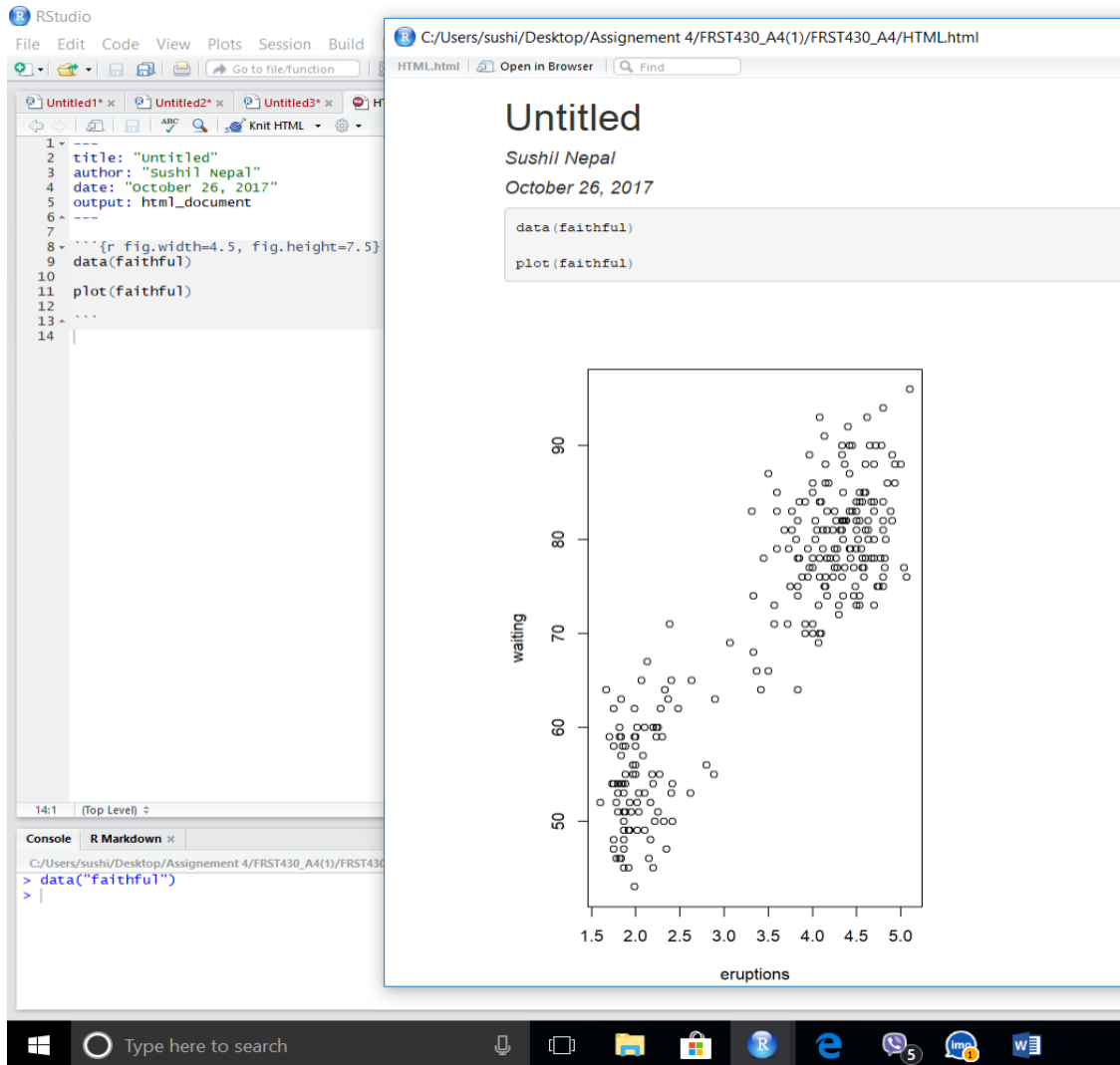


Figure 6: Document generated for HTML format using the code give in section 3.

## 5. Including Graphs

You can also embed graphs into your R Markdown file using simple graphing code inside the R code sections. You can adjust the height and the width of your graph by using the options (`fig.width` and `fig.height`) as given in the example code below.

Example code here (manipulate the code using different height and width):

```

```{r fig.width=4.5, fig.height=7.5, echo=FALSE}
data(faithful)
plot(faithful)
```

```

October 26, 2017

```
data(faithful)
```

```
plot(faithful)
```

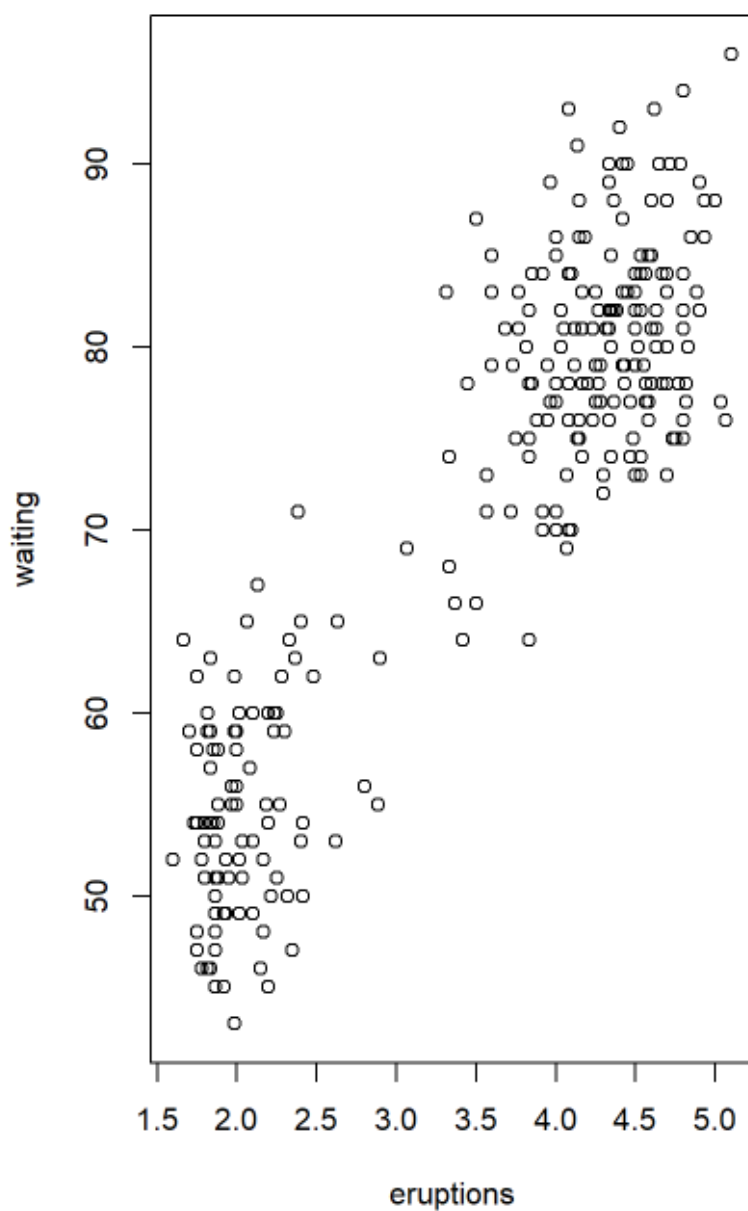


Figure 7: Output generated from R Markdown for graphs.

Note: Adding the code `echo = FALSE` will prevent the R code from being printed in the output document. In other words, the knitted files will only include the output of your codes!  
Source: <https://shiny.rstudio.com/articles/rmarkdown.html>

## 6. Formatting texts in your R Markdown chunks

### 6.1 Emphasis.

R Markdown syntax can be used to change how text appears in your output file. Here are a few common formatting commands using single `*` or double `**` asterisks and single or double underscores `'_'`:

*\*Italic\** -- single asterisk to indicate emphasis.

**\*\*Bold\*\*** -- Double asterisk to indicate bold.

*\_Italic\_* - Single underscores to indicate emphasis.

**\_\_Bold\_\_** - double underscores to indicate bold.

### 6.2. Creating an unordered list.

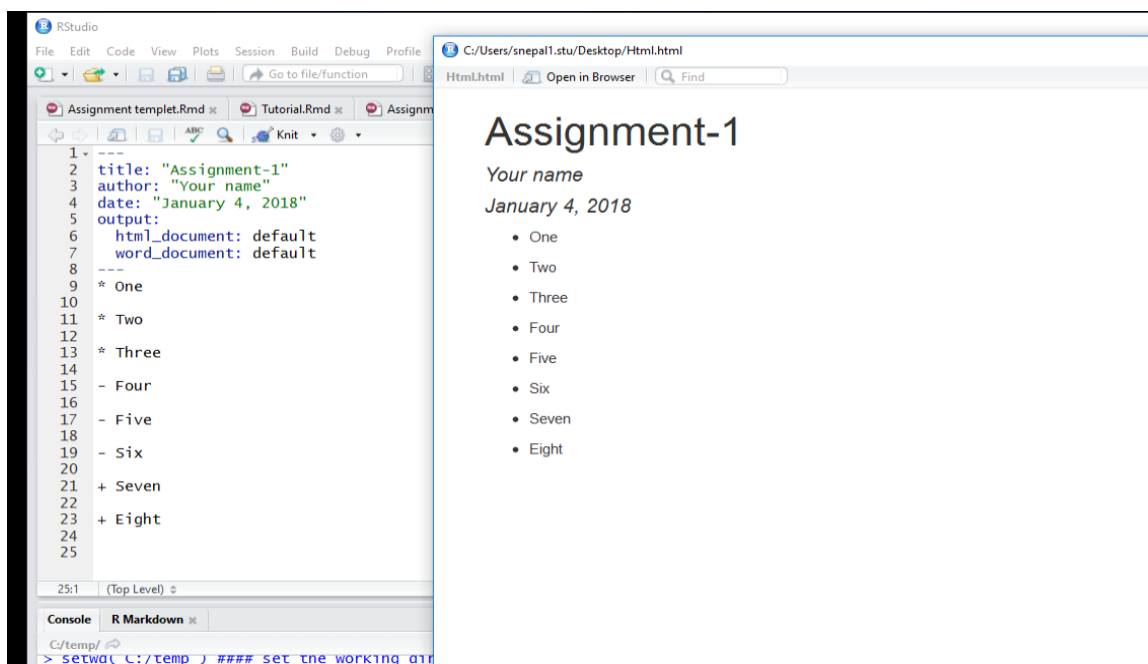


Figure 8: Creating an unordered list.

An unordered list can be created using stars (`*`), and/or hyphens (`-`) and/or pluses (`+`). The choice is yours. Whichever of the three you decide to use will give you the same result. The text with a single space follows these signs.



### 6.3 Creating an ordered list.

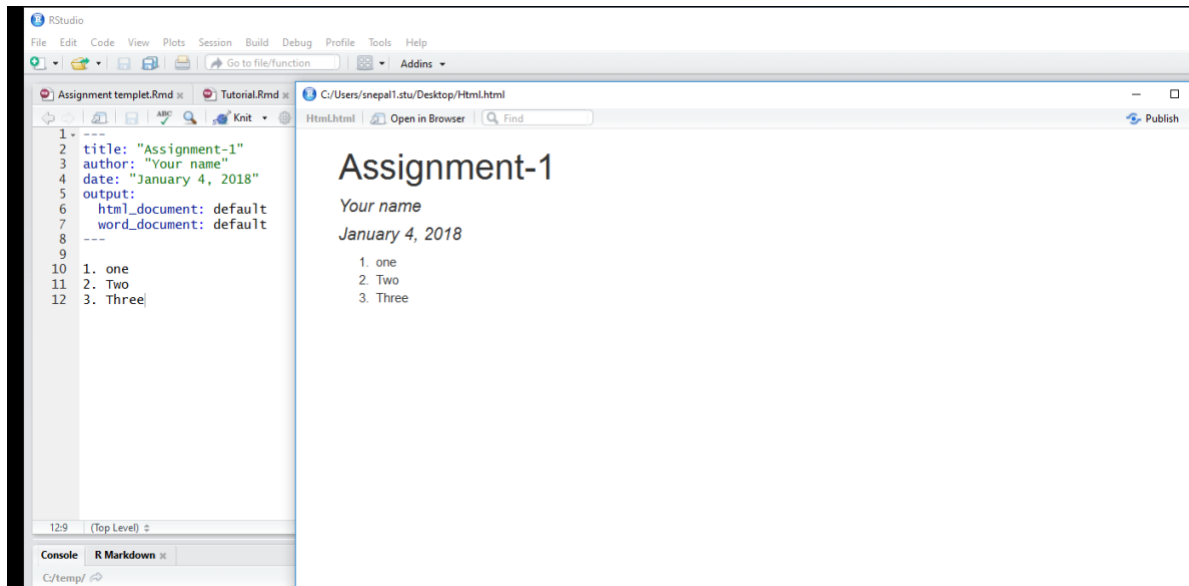


Figure 9: Creating an ordered list.

We can create an ordered list by simply using the number followed by the text as given in Figure 9.

### 6.4. Controlling the size of the header

The size of the header is controlled by the number of hashtags (#) preceding the header. The more hashtags the smaller the font size. An illustration is given in Figure 10.

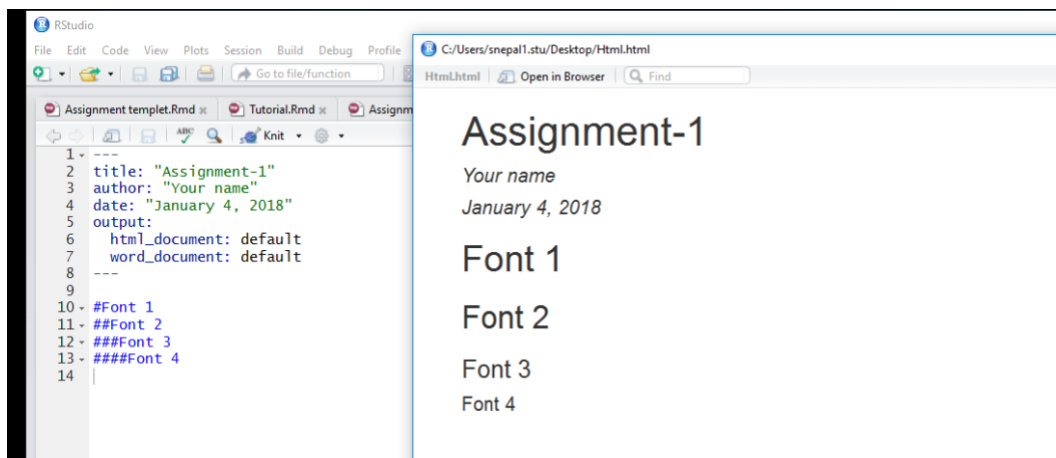


Figure 10: Controlling the size of the header.

## 6.5 External links

External links can be embedded in R Markdown documents using the square brackets `[]` inside which you type the link phrase (phrase that will appear on your Word output, which can be clicked to go to the link directly). The link to the page is then included inside parentheses `()`.

For example:

```
[stackoverflow] (https://stackoverflow.com/questions/29787850/how-do-i-add-a-url-to-r-markdown)
```

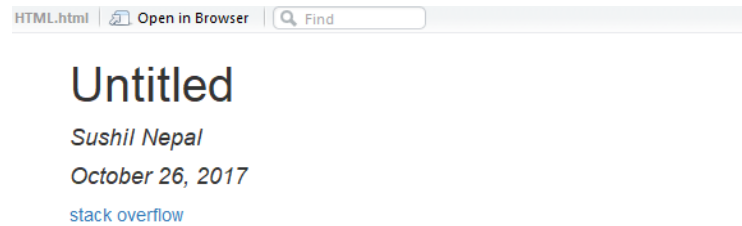


Figure 11: Output from above code for external links.

## 7. Using R Markdown to include Greek letters and symbols:

R Markdown allows you to include Greek letters and symbols using the LaTeX syntax (mathematics coding program). Commands are preceded by a dollar `$` sign followed by `{\backslash}`. Thus, very simple commands produce a Greek letter or symbol.

### *For example: Generating the Greek symbols*

Code:  `${\alpha} = 4$`

Output:  $\alpha = 4$

Code:  `${\alpha} \neq 4$`

Output:  $\alpha \neq 4$

Code:  `${\beta} = 4$`

Output:  $\beta = 4$

Code:  `${\beta} \neq 4$`

Output:  $\beta \neq 4$

***For example: Listing equations.***

Code: `\$A = \pi \times r^{2}\$`

Output:  $A = \pi \times r^2$

Code: `\$\hat{y}=4y+6\$`

Output:  $\hat{y} = 4y + 6$

***Superscript:***

In order to write a superscript in your text use circumflex (‘^’) sign. Start the text followed by the circumflex sign and put the text or number you want to add on the superscript and close it with circumflex sign.

For example:

Code: `R^{2}`

Output:  $R^2$

***Subscript:***

In order to write a subscript in your text use underscore (‘\_’) sign. Start the \$ sign followed with text which is then followed by the underscore sign and put the subscript you want in curly brackets and close it with dollar (‘\$’) sign.

For example:

Code: `\$X_{i}\$`

Output:  $X_i$