

Data Wrangling and Visualization, Week 3

The Philosophy of Writing Text

Many people like writing in WYSIWYG writing environments (like Word), as they can tell immediately how a document looks. Others, for a variety of reasons, dislike that intensely. Writing in a *markup* language is a different approach to writing that *separates* writing the content, which happens in a plain text file, from viewing the formatted document, which happens in some separate output file.

While R is a computer language, written in plain text, for performing statistical analysis - kind of a supercharged version of Excel on steroids - a *markup* language is a type of computer language, written in plain text, for the purpose of authoring structured documents. Basically, it's kind of like the engineer or computer person's variant of Microsoft Word, except open source, based on plain text files, and where you declare, in plain text, what you want to do. Depending on the markup language you use, it might be more or less capable than Word. (Although it's probably better to think of it as "differently capable"). HTML, for instance, is a complicated markup language for creating webpages; LaTeX is a complicated and powerful markup language for creating structured documents, etc. We're going to use a much simpler markup language, but one that will still give us the desirable features of being open and portable while interfacing with R cleanly.

While we don't need to turn this into an extensive diatribe, there are at least three key reasons to prefer writing in a *markup* language, rather than Word:

- Many people find it freeing to separate the process of writing your content from worrying about how your document looks. Aficionados of this view (like yours truly) will argue that *writing* and *formatting* are two different tasks, and for the sake of mental clarity, it's best to attend to them separately, not together.
- Plain text documents - of which all markup languages are written in - are open and free. You'll never need a license to a program to use them, and if you decide to not purchase a subscription to Microsoft Word in the future, you won't lose any access to your text documents - although your .docx files are another story. You can also edit plain text documents in any of a million different editors, which honestly has both pluses and minuses.

- Plain text documents enable much greater *interoperability* with other programming languages. You can, for instance, call out to R, Python, or a variety of other software engines from directly within the plain text document, provided that the interpreter you’re using knows what to do with that callout. This enables excellent “notebook” type functionality.

With that said, writing in markup languages come with costs, too.

- You need to remember exactly how to mark things up, which can mean a lot of looking things up.
- A lot of the promise of markup languages is *if everything works correctly*, and the simple truth is that often, things don’t work correctly, and you have to get out of the car and look under the hood. In other words, you often have to know what you’re doing. Programmers love markup languages, because they don’t mind looking under the hood (honestly, that’s kind of a plus for them) and they like feeling like they have control.
- While some dislike writing things in Word, it is inarguable that a lot of very talented software engineers have spent a very long time designing the software to be (relatively) easy to use. If you just want to dash off a quick document, it’s hard for any markup language to beat that simplicity.
- In a related fashion, writing text in Rstudio (or other editors) can mean writing without niceties like spell checks and grammar checks. There are ways to fix that - and other editors to use - but if you notice more spelling errors in .qmd documents, it’s probably because this kind of stuff doesn’t come as easily.

Markdown

This is an Quarto document, which is based on a *markup* language known as *Markdown* (markup/Markdown - get it? programmers make jokes). Markdown itself is a simple and lightweight formatting syntax for authoring HTML documents. It was originally written by software developer/technology blogger John Gruber, for easier blog post writing, and you can see its original specification here: <https://daringfireball.net/projects/markdown/> and a more extensive general guide here <https://www.markdownguide.org>.

In a markup language, rather than entering a command (like control-B or command-B) or clicking a menu option to bold text, you explicitly declare in the text when you want formatting to change. A later step - rendering or typesetting - interprets these commands and applies them. For instance, if you want to **bold** text, you put two asterisks around what you want to bold. If you want to *italicize* text, you put one. You can also author lists, using hashes:

- Here is a first bullet point
- Here is a second bullet point

You can also create clickable web links with angle brackets, like this https://jackreilly.github.io/dwv_f25, or footnotes with carets and brackets, like this¹. Sections are created using the hashtag (look at the sections in the raw quarto document) and you can create hierarchy with few or more sections, like so:

This is a section

This is a subsection

This is a subsubsection

You can also create tables, using relatively simple characters:

Column A	Column B
Row 1	Value 1
Row 2	Value 2

And so on and so forth. The markdown guides will give you a feel for the other kinds of markup you can write in markdown.

Quarto

You can think of Quarto as extra scaffolding around Markdown that adds functionality. Most importantly, this is the ability to execute chunks of R code (or, actually, other code) in stream, but also includes other things, like the ability to turn your document into PDF and MS Word, or the ability to publish a webpage (relatively) easily.² For more details on using Quarto, see <https://quarto.org>.

There are a few initial warnings for writing in markdown and quarto:

- Space matters. Lists, headers, etc can be affected if you don't indent properly.
- Space especially matters in headers.
- Early on, it is good practice to semi-regularly *render* and preview your quarto documents, just to make sure everything looks like you expect.
 - Yes, this kind of contradicts the “worry about content and how it looks separately” philosophy of plain text writing, but early on, it's just practical.

¹This is the text of my footnote. I can do everything here I can do elsewhere, like **bold** and *italicize*.

²Our course webpage, for instance, is written in Quarto and hosted on GitHub.

- Whenever I’m working in a markup language, I tend to render (or typeset) the document when I come to a pause, or when I’m not quite sure what to do next. But I don’t let it interrupt the flow of my writing or code. The “render on save” option, at the head of your document, is useful for this.

How To Start

You can start a new Quarto/RMarkdown document, complete with the template, by selecting File»New File»Quarto (or RMarkdown).

When you click the **Render** button (or **Knit**, if you’re using RMarkdown) a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. (HTML is a good default to use, but PDF and Word is obviously useful as well. Different output formats may require you to install additional software, like LaTeX.) Upon rendering, RStudio will open a web browser with your html document in it, and the HTML document, by default, will appear in the same directory on your computer as the Quarto (.qmd) file. You can select different output formats by declaring them in the beginning of your document and then selecting that output format through the dropdown menu next to “Render” in Rstudio.

As an aside, you should go look at the source for your rendered HTML file (just open it in a text editor). That’s what a more complex markup language looks like! Seeing that always makes me appreciate Quarto more.

One of the biggest reasons to use Quarto is because you can embed R code directly in it, using tick marks. You can embed an R code chunk like this:

```
a <- 1 + 1
a*3
```

```
[1] 6
```

Note that the output document shows the code (because we asked it to) but also prints out any results that we generate.

Let’s use one of the built-in datasets in R, the “cars” dataset. The cars dataset is a toy dataset, with 50 observations and 2 numeric variables:

- speed → Speed of cars (in miles per hour)
- dist → Stopping distance (in feet)

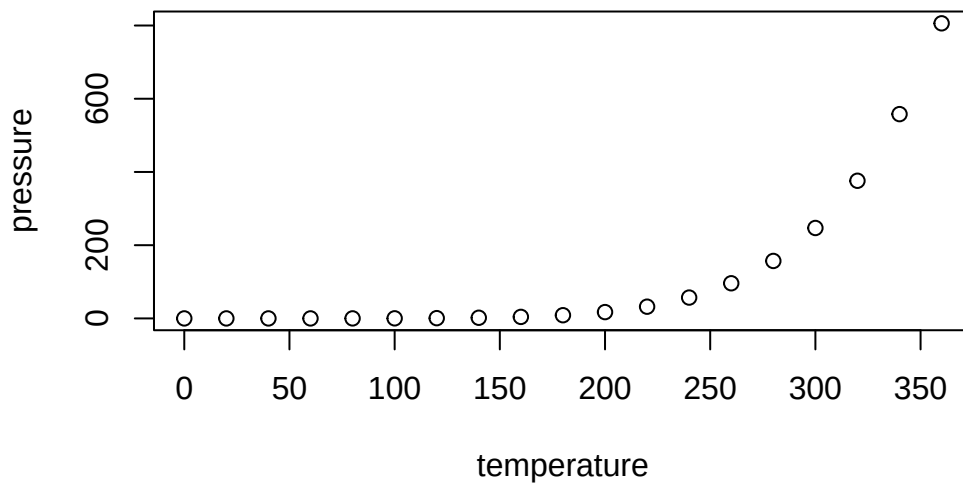
We can tell R to use the cars dataset, and then use the “summary” command to summarize the dataset.

```
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, let's use the pressure dataset, which reports temperature and pressure of mercury at that temperature.



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

Loading external data

By default, the working directory for R code chunks is the directory that contains the Rmd document. So make sure the dataset you want to use is in the same folder (directory) as your qmd file.

Here, we'll load a subset of the 2005 British Election Study, which saw Tony Blair re-elected, and look at a few correlates surrounding his popularity.

```
bes<-read.csv("bes2005subset.csv")
summary(bes)
```

LabourVote	IraqWarApprovalDich	IraqWarSuccessScale	BlairFeelings
Min. :0.0000	Min. :0.0000	Min. : 0.000	Min. : 0.000
1st Qu.:0.0000	1st Qu.:0.0000	1st Qu.: 1.000	1st Qu.: 4.000
Median :0.0000	Median :0.0000	Median : 3.000	Median : 6.000
Mean :0.3963	Mean :0.3261	Mean : 3.378	Mean : 5.852
3rd Qu.:1.0000	3rd Qu.:1.0000	3rd Qu.: 5.000	3rd Qu.: 8.000
Max. :1.0000	Max. :1.0000	Max. :10.000	Max. :10.000
NA's :1768	NA's :1286	NA's :1258	NA's :655

EconRet	SelfLR	SocialSpending
Min. :1.000	Min. : 0.000	Min. : 0.000
1st Qu.:2.000	1st Qu.: 5.000	1st Qu.: 5.000
Median :3.000	Median : 5.000	Median : 6.000
Mean :2.721	Mean : 5.418	Mean : 6.154
3rd Qu.:3.000	3rd Qu.: 7.000	3rd Qu.: 7.000
Max. :5.000	Max. :10.000	Max. :10.000
NA's :1320	NA's :1247	NA's :732

Once you do something in one code chunk, note that access to those objects persist to the next code chunk, but *also* note that they don't show up in your global environment.

Any regular R commands can be used in the R snippets. For instance, the `cor()` command calculates a correlation coefficient, and takes two or more arguments: the two variables (Don't forget, you can use the `help()` command to show how to use a function).

Let's say we want to see if there is a relationship between feelings toward Tony Blair and Evaluation of the Economy, and then the relationship between Feelings towards Blair and Social Spending:

```
cor(bes$BlairFeelings, bes$EconRet, use="pairwise.complete.obs")
```

```
[1] 0.3212059
```

```
cor(bes$BlairFeelings, bes$SocialSpending, use="pairwise.complete.obs")
```

```
[1] 0.1626111
```

We can also draw figures, the result of which will be included in the RMarkdown file (at least, those that are not commented out will be! Comments (the “#” sign) perform the same function as in regular R.)

```
table(bes$SelfLR)
```

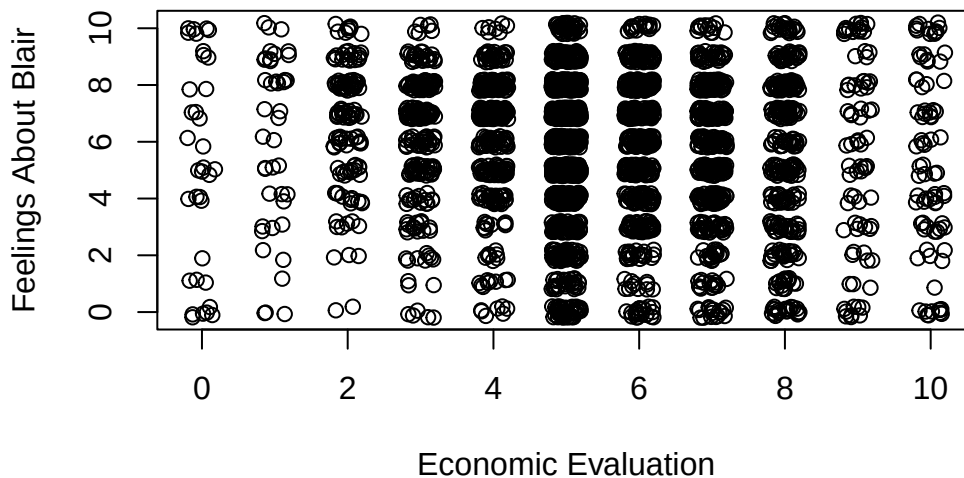
0	1	2	3	4	5	6	7	8	9	10
36	41	124	241	366	1295	532	454	295	78	82

```
#plot(bes$BlairFeelings~bes$EconRet)
```

```
#plot(jitter(bes$BlairFeelings)~jitter(bes$EconRet))
```

```
plot(jitter(bes$BlairFeelings)~jitter(bes$SelfLR), main="Opinion of Tony Blair and Economic L
```

Opinion of Tony Blair and Economic Evaluation



Images

Some people prefer to separate their code from their writing. Qmd files are great for just writing without code, too! And you can do all of your analysis in an ordinary .R file. But if you want to include graphics in a .qmd without code, you must point to where the image can be found. Qmd files, by default, look in the same location in your filesystem that they are in for secondary files. It's usually good practice to put images in their own separate subfolder inside that directory. A good workflow, actually, would be to use an R file to programmatically put all of your statistical graphics in a directory that Quarto then reads in automatically.

Once you put your graphics in the proper directory, you can easily include an image in a Quarto file like this:



Figure 1: Picture of Mt. Marcy

Mt. Marcy is the highest point in New York State. Some may feel that summitting Mt. Marcy and learning open-source scientific tools are equally challenging accomplishments.

Next week's assignment

1. Create an Quarto file; title it "Week 3 Homework.qmd"
2. Use the 2005 British Election study file (`bes2005subset.csv`) to create a jittered scatterplot of Feelings toward Blair and Feelings on Social Spending.
3. In a brief text segment, interpret what the figure shows you.
4. Load the 2000 National Election study file (`nes2000subset.csv`).

5. Find the correlation between income and partisanship, and draw a jittered scatter plot of partisanship (Y axis) and income (X-axis)
6. In a brief text segment, interpret what the correlation and figure tells you.
7. Render the document to HTML and docx and upload all three (.qmd, the .html, and .docx files) to Blackboard.