Data Science for Public Policy Part II

Aaron R. Williams

2023-08-10

Table of contents

Preface

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

```
1 + 1
```

```
[1] 2
```

1 Advanced Quarto

This set of notes reviews Quarto basics and introduces advanced Quarto concepts including mathematical notation, cross references, and citations. The Quarto website includes more gems and is an example of high-quality technical documentation.

1.1 Review

1.1.1 Motivation

There are many problems worth avoiding in an analysis:

- Copying-and-pasting, transposing, and manual repetition
- Running code out-of-order
- Maintaining parallel documents like a script for analysis and a doc for narrative
- Code written for computers that is tough to parse by humans

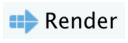
Not convinced? Maybe we just want to make cool stuff like websites, blogs, books, and slide decks.

Quarto, a literate statistical programming framework for R, Python, and Julia helps us solve many of these problems. Quarto uses

- plain text files ending in `.qmd` that are similar to `.R` and `.Rmd` files
- `library(knitr)`
- pandoc

Quarto converts these plain text documents into rich output documents like these class notes. The "Render" button appears in RStudio with a `.qmd` file is open in the editor window.

Clicking the "Render" button begins the process of rednering `.qmd` files.



When the button is clicked, Quarto calls `library(knitr)` and renders `.qmd` (Quarto files) into `.md` (Markdown files), which Pandoc then converts into any specified output type. Quarto and `library(knitr)` don't need to be explicitly loaded as the entire process is handled by clicking the "Render" button in RStudio.

Source: Quarto website

Quarto, `library(knitr)`, and Pandoc are all installed with RStudio. You will need to install a LaTeX distribution to render PDFs. We recommend `library(tinytex)` as a LaTeX distribution (installation instructions).

Exercise 1

1. Click the new script button in RStudio and add a "Quarto Document".
2. Give the document a name, an author, and ensure that HTML is selected.
3. Save the document as "hello-quarto.qmd".
4. Click "Render".

Quarto has three main ingredients:

1. YAML header
2. Markdown text
3. Code chunks

1.1.2 1. YAML Header

YAML stands for "yet another markup language". The YAML header contains meta information about the document including output type, document settings, and parameters that can be passed to the document. The YAML header starts with `---` and ends with `---`.

Here is the simplest YAML header for a PDF document:

```
---
format: pdf
---
```

YAML headers can contain many output specific settings. This YAML header creates an HTML document with code folding and a floating table of contents:

```
---
format:
  html:
    embed-resources: true
    code-fold: true
    toc: true
---
```

Parameters can be specified as follows

```
---
format: pdf
params:
  state: "Virginia"
---
```

Now state can be referred to anywhere in R code as `params$state`.

> **Warning**
>
> Unlike R Markdown, images and other content are not embedded in .html from Quarto by default. Be sure to include `embed-resources: true` in YAML headers to embed content and make documents easier to share.

1.1.3 2. Markdown text

Markdown is a shortcut for HyperText Markup Language (HTML). Essentially, simple meta characters corresponding to formatting are added to plain text.

```
Titles and subtitltes
------------------------------------------------------------

# Title 1

## Title 2

### Title 3


Text formatting
```

```
------------------------------------------------------------

*italic*

**bold**

`code`

Lists
------------------------------------------------------------

* Bulleted list item 1
* Item 2
  * Item 2a
  * Item 2b

1. Item 1
2. Item 2

Links and images
------------------------------------------------------------

[text](http://link.com)
```

1.1.4 3. Code chunks

```
Code is added to R Markdown documents inline with `r 2 + 2`.
```

More frequently, code is added in code chunks:

```
  ```{r}
  2 + 2
  ```
```

```
[1] 4
```

The first argument inline or in a code chunk is the language engine. Most commonly, this will just be a lower case `r`. `knitr` allows for many different language engines:

- R
- Julia
- Python

- SQL
- Bash
- Rcpp
- Stan
- Javascript
- CSS

Quarto has a rich set of options that go inside of the chunks and control the behavior of Quarto.

````
```{r}
#| label: important-calculation
#| eval: false

2 + 2
```
````

In this case, `eval` makes the code not run. Other chunk-specific settings can be added inside the brackets. Here[^1] are the most important options:

Option	Effect
echo: false	Hides code in output
eval: false	Turns off evaluation
output: false	Hides code output
warning: false	Turns off warnings
message: false	Turns off messages
fig-height: 8	Changes figure width
fig-width: 8	Changes figure height

Default settings for the entire document can be changed in the YAML header with the `execute` option:

```
execute:
  warning: false
```

[^1] This table was typed as Markdown code. But sometimes it is easier to use a code chunk to create and print a table. Pipe any data frame into `knitr::kable()` to create a table that will be formatted in the output of a rendered Quarto document.

> **Exercise 2**

1. Add `date: today` to your YAML header after `title`. This will update every time the document is rendered.
2. Copy the Markdown table from this table generator and add it to your `.qmd` document.
3. Create a scatter plot of the `cars` data with `library(ggplot2)`. Adjust the figure width and height using options within the chunk.
4. Click "Render".

1.2 Math Notation

This course uses probability and statistics. Occasionally, we want to easily communicate with mathematical notation. For example, it may be convenient to type that X is a random variable that follows a standard normal distribution (mean = 0 and standard deviation = 1).

$$X \sim N(\mu = 0, \sigma = 1)$$

1.2.1 Math Mode

Use `$` to start and stop in-line math mode and `$$` to start multi-line math mode.

Here's an example with in-line math:

```
Consider a binomially distributed random variable, $X \sim binom(n, p)$.
```

Consider a binomially distributed random variable, $X \sim binom(n, p)$.

Here's an example with a chunk of math:

```
$$
P(X = x) = {n \choose x} p ^ x (1 - p) ^ {n - x}
$${#eq-binomial}
```

$$P(X = x) = \binom{n}{x} p^x (1-p)^{n-x} \tag{1.1}$$

1.2.2 Important Syntax

Math mode recognizes basic math symbols available on your keyboard including `+`, `-`, `*`, `/`, `>`, `<`, `(`, and `)`.

Math mode contains all greek letters. For example, `\alpha` (α) and `\beta` (β).

Table 1.2: My Caption

LaTeX	Symbol
`\alpha`	α
`\beta`	β
`\gamma`	γ
`\Delta`	Δ
`\epsilon`	ϵ
`\theta`	θ
`\pi`	π
`\sigma`	σ
`\chi`	χ

Math mode also recognizes $\log(x)$ (`\log(x)`) and $\sqrt{x}$ (`\sqrt{x}`).

Superscripts (`^`) are important for exponentiation and subscripts (`_`) are important for adding indices. `y = x ^ 2` renders as $y = x^2$ and `x_1, x_2, x_3` renders as x_1, x_2, x_3. Brackets are useful for multi-character superscripts and subscripts like s_{11} (`s_{11}`).

It is useful to add symbols to letters. For example, `\bar{x}` is useful for sample means ($\bar{x}$), `\hat{y}` is useful for predicted values ($\hat{y}$), and $\vec{\beta}$ is useful for vectors of coefficients ($\vec{\beta}$).

Math mode supports fractions with `\frac{x}{y}` ($\frac{x}{y}$), big parentheses with `\left(\right)` (()), and brackets with `\left[\right]` ([]).

Math mode has a symbol for summation. Let's combine it with bars, fractions, subscripts, and superscipts to show sample mean `\bar{x} = \frac{1}{n}\sum_i^n x_i`, which looks like $\bar{x} = \frac{1}{n}\sum_i^n x_i$.

`\sim` is how to add the tilde for distributed as. For example, `X \sim N(\mu = 0, \sigma = 1)` shows the normal distribution $X \sim N(\mu = 0, \sigma = 1)$.

Matrices are are a little bit more work in math mode. Consider the follow variance-covariance matrix:

```
\begin{bmatrix}
s_{11}^2 & s_{12}\\
s_{21} & s_{22}^2
```

```
\end{bmatrix}
```

$$\begin{bmatrix} s_{11}^2 & s_{12} \\ s_{21} & s_{22}^2 \end{bmatrix}$$

This guide provides and exhaustive look at math options in Quarto.

> **Warning**
>
> Math mode is finicky! Small errors like mismatched parentheses or superscript and subscript errors will cause Quarto documents to fail to render. Write math carefully and render early and often.

Exercise 3

1. Use math mode to type out the equation for root mean square error (RMSE).
2. Do you divide by `n` or `n - 1`?

1.3 Cross References

Cross references are useful for organizing documents that include sections, figures, tables, and equations. Cross references create hyperlinks within documents that jump to the locations of these elements.

Cross references also automatically number the referenced elements. This means that if there are two tables (ie. Table 1 and Table 2) and a table is added between the two tables, all of the table numbers and references to the tables will automatically update.

Cross references require two bits of code within a Quarto document:

1. A label associated with the section, figure, table, or equation.
2. A reference to the labelled section, figure, table, or equation.

1.3.1 Sections

Linking sections helps readers navigate between sections. Use brackets to label sections after headers and always begin labels with `sec-`. Then you can reference that section with `@sec-`.

```
## Review {sec-review}

See @sec-review if you are totally lost.
```

The cross references shows up like this: See Section 1.1 if you are totally lost.

It can be helpful to turn on section numbering with `number-sections: true` in the YAML header. Additionally, Markdown has a native method for linking between sections.

Exercise 4

1. Add a few section headers to your Quarto document.
2. Add a cross reference to one of the section headers.

1.3.2 Figures

Figure 1.1: Penguins

We can reference figures like Figure 1.1 with `@fig-penguins`.

1.3.3 Tables

We can link to tables in our documents. For example, we can link to the greek table with `@tbl-greek` Table 1.2.

1.3.4 Equations

We can link to equations in our documents. For example, we can link to the binomial distribution earlier with `@eq-binomial` Equation 1.1.

Exercise 5

1. Add a cross reference to your RMSE equation from earlier.

1.4 Citations

1.4.1 Zotero

Zotero is a free and open-source software for organizing research and managing citations.

Digital Object Identifier (DOI)

DOIs are persistent identifiers that uniquely identify objects including many academic papers. For example, `10.1198/jcgs.2009.07098` identifies "A Layered Grammar of Graphics" by Hadley Wickham.

Exercise 6

1. Install Zotero.
2. Find the DOI for "Tidy Data" by Hadley Wickham.
3. Click the magic wand in Zotero and paste the DOI.

4. Review the new entry in Zotero.

1.4.2 Zotero Integration

Zotero has a powerful integration with Quarto. In practice, it's one click to add a DOI to Zotero and then one click to add a citation to Quarto.

RStudio automatically adds **My Library** from Zotero. Simply switch to the Visual Editor (top left in RStudio), click "Insert", and click "Citation". This will open a prompt to insert a citation into the Quarto document.

The citation is automatically added with parentheses to go at the end of sentences. Delete the square brackets to convert the citation to an in-line citation.

Inserting the citation automatically adds the citation to the references section. Deleting the reference automatically deletes the citation from the references section.

Zotero Groups are useful for sharing citations and Zotero Group Libraries need to be added to RStudio. To set this up:

1. Select "Global Options"
2. Select "Citations" under "R Markdown"
3. Select "Selected Libraries" to the right of "Use libraries"
4. Select the Group libraries to add

Exercise 7

1. Cite "Tidy Data" by Hadley Wickham in your Quarto document.
2. Click "Render"

References