Research Toolkit

Nikita Tkachenko

3/14/23

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
```

Chapter 1

Introduction

This is a book created from markdown and executable code.

See Knuth (1984) for additional discussion of literate programming.

Knuth, Donald E. 1984. "Literate Programming." *Comput. J.* 27 (2): 97–111. https://doi.org/10.1093/comjnl/27.2.97.

```
1 + 1
```

```
[1] 2
```

Chapter 2

Summary

In summary, this book has no content whatsoever.

```
1 + 1
```

```
[1] 2
```

Chapter 3

Set up

We will be using R and RStudio IDE to download, clean, explore, and model our data. R language was developed by statisticians for statisticians and had intuitive syntax and workflow for creating readable and **reproducible** code.

3.1 Download R

R is maintained through The Comprehensive R Archive Network.

3.1.1 For macOS

1. Go to https://cran.r-project.org/
2. Select "Download R for macOS"
3. If you have Apple Silicon mac (M1,M2…) download the latest version that has -arm64 in its name (R-4.2.2-arm64.pkg)
4. If you have Intel Max Download one without -arm64 (R-4.2.2.pkg)
5. Follow the steps from the installation wizzard
6. The installer lets you customize your installation, but the defaults will be suitable for most users
7. Your computer might ask for your password before installing new programs

3.1.2 For Windows

1. Go to https://cran.r-project.org/
2. Select "Download R for Windows"
3. Click on "base"

4. Click the first link at the top of the new page (Download R-4.2.2 for Windows)
5. ink downloads an installer program, which installs the most up-to-date version of R for Windows.
6. Follow the steps from the installation wizzard
7. The installer lets you customize your installation, but the defaults will be suitable for most users
8. You might need administration privileges to install new software on your machine

3.2 Download RStudio

Rstudio is like a Microsoft Word for writing text, but instead of text RStudio helps you write in R. RStudio is also free and easy to install! Go to RStudio Website, click on DOWNLOAD RSTUDIO or select your operation system below, and follow the isntallation instructions.

3.3 Configure RStudio

First watch an introduction to RStudio: https://www.youtube.com/watch?v=FIrsOBy5k58 and introduction to projects: https://www.youtube.com/watch?v=MdTtTN8PUqU

Now the most crucial part!!! Change the default theme of RStudio from light to dark! To do this

1. Go to the top panel
2. Select the Tools tab
3. Navigate to Global Options
4. Select Appearance
5. In Editor Theme, select “Dracula”
6. Click Apply

Now let’s install a cooler font with ligatures! Install fira-code: https://github.com/tonsky/FiraCode/wiki/Installing. **Restart** RStudio for the font to load. Then go back to Appearance, choose FiraCode, and hit apply.

3.4 Install Packages

An R package is a collection of useful functions, documentation, and data sets that can be used in your own R code after it is loaded. These packages typically

focus on a specific task and make use of pre-written routines for various data science tasks.

You can install packages with a single line of code:

```
install.packages("tidyverse")
```

You can also install multiple packages at the same time using c():

```
install.packages(c("tidyverse","gapminder"))
```

You can load packages using the library() function:

```
library("tidyverse")
```

Run the following command to install packages we will use in class

```
install.packages(c("tidyverse", "janitor", "esquisse", "modelsummary", "styler"))
```

Part I

Writing and Lit Review

Chapter 4

Write

WYSIWYG (What You See Is What You Get) editors are programs that allow you to make and edit content visually without having to know how to code. This means that you can see how your content will look as you're creating it. They're very helpful for people who don't have experience with coding or markup languages. Examples of WYSIWYG editors that you may already be familiar with include Google Docs and Microsoft Word. Learning how to use these tools is important because they are some of the most widely used ones, and they teach you valuable skills like formatting and writing. Plus, after using them for a while, you'll start to appreciate the simplicity and efficiency of markup languages.

I strongly recommend that you develop expertise in these tools, as the skills you acquire are highly transferable to other similar editors. Additionally, it's a valuable investment as your colleagues are likely to use them as well. To get started, I suggest checking out the MOS Certification Series on LinkedIn Learning. These tutorials are designed for those preparing to take MOS exams and cover a broad range of functionality in Word, Excel, and PowerPoint. Even if you don't plan on taking the exams, I highly encourage you to watch the tutorials. They're a great resource for building your skills and improving your proficiency in these essential tools!

here is the link

4.1 Markup Languages

Markup languages are sets of codes that provide structure and formatting to documents, such as web pages, eBooks, and scientific papers. Unlike WYSIWYG editors, which allow users to create content visually, markup languages require the use of specific tags or codes to indicate how the content should be

displayed. These tags define the document structure, formatting, and other attributes. Some of the most commonly used markup languages include HTML, LaTeX, and Markdown. By learning a markup language, you can have greater control over the appearance and functionality of your digital documents. Additionally, markup languages can help you focus more on writing letting templates handle all the formatting. I promise after you switch there is no coming back! Your assignments and web pages will look beautiful every time.

4.1.1 HTML

When you hear markup language you might think of Hypertext Markup Language (HTML), which is used everywhere on the web to build up structure and content. HTML uses tags to describe how to display content in a browser. For instance, <br> will create a line break, <img> will add an image and so on. HTML is an essential component of web development, and learning how to use HTML is the first step towards customizing your own web pages and web applications. It is also often used in manipulating text in your graphs.

4.1.2 LaTeX

Have you ever wondered why academic papers look so beautiful or why every textbook looks the same? They are all written in LaTeX, a typesetting system used for academic and scientific publishing. It is a markup language that enables precise formatting of complex technical documents. It is also famous for beautiful mathematical equations. With LaTeX, users can create professional-looking documents with a high degree of customization and flexibility. It is easy to pick up, but hard to master. You can find many introductory tutorials online. Also, ChatGPT does a great job at helping with LaTeX questions. If you have an equation you want to transfer into LaTeX go to MathPix. It lets you convert picture or screenshot into code.

Now you need to find an editor! The default is to use the online editor Over Leaf. You can start writing without much headache, perfect! And if you have any question refer to their extensive collection of guides.

4.1.3 Markdown

Markdown is perfect down the last-minute details. It was created as a lightweight markup language for creating formatted text using a plain-text editor. It appeals to human readers in its source code form as a response to the complexity and inefficiency of existing markup languages like HTML.

Markdown is popular among developers, writers, and bloggers due to its simplicity and flexibility. It can also include links, images, and other multimedia

elements. Markdown is widely supported and can be used with a variety of tools and platforms, including text editors, note-taking apps, and content management systems. By learning Markdown, you can create well-formatted and easy-to-read content quickly and efficiently without the need for complex formatting tools or specialized software. This book is written entirely in Markdown!

You can write markdown in almost any text editor. Nonetheless, I recommend MacDown for Mac and Markdown Pad for Windows, or if you are willing to pay, try Typora.

4.1.4 Yet Another Markup Language (YAML)

YAML (Yet Another Markup Language) is a human-readable data serialization language often used for configuration files and data exchange. It uses indentation, key-value pairs, and directories and supports different data types. Directories are used to organize data as key-value pairs with indentation to indicate hierarchy. The key represents a unique identifier or name, and the value represents associated data or content. YAML is easy to read and write and is popular in web development and software configuration. In YAML,

```yaml
book:
  title: "Dead Souls"
  author: "Nikolai Gogol"
  date: today
  chapters:
    - index.qmd
    - chapter1.qmd
  appendix:
    - appendix1.qmd
```

```yaml
% Key value pairs
title: "Dead Souls"
author: "Nikolai Gogol"
date: today

% Directory/Map
chapters:
  - index.qmd
  - chapter1.qmd
```

4.1.5 Pandoc

Pandoc (all documents) converts files from one markup format into another. It supports a wide range of formats, including HTML, Markdown, LaTeX, PDF, and Microsoft Word. With Pandoc, users can convert documents from one markup language to another quickly and easily, without having to edit or reformat the content manually.

Pandoc can be used for a variety of purposes, such as converting a Markdown file to HTML for a web page, or converting a LaTeX document to PDF for printing. It can also be used to merge multiple documents into a single file, or to extract content from a document in a specific format.

Pandoc is highly customizable, with many options and settings available for controlling the output format and appearance of the converted document. It is widely used in academic and scientific publishing, as well as in web development and documentation. By using Pandoc, users can save time and effort by automating the conversion process between different markup formats.

Pandoc is a command-line tool, but it can be integrated into some markdown editors. It extends the capabilities of markdown by adding support for tables, footnotes, citations, and more. For example, this book was written in markdown and converted into HTML and PDF using Pandoc.

4.2 Quarto

At this point, you might be wondering, "why did I have to read about these technologies?" Because the combination of LaTeX, Markdown, Pandoc, and YAML provides a powerful and flexible set for the creation of documents. Those familiar with Sweave, Rmarkdown, and Jupyter Notebooks will find Quarto to be a similar tool. Quarto builds upon the success of Rmarkdown and Jupyter Notebooks, combining the best features of Markdown with new functionality and eliminating the need for additional packages. It provides attractive default formatting options and easy customization. If you have experience writing in Markdown, Quarto will be a breeze to use.

What can it do?

- Create Reproducible Documents and Reports.
- Create Dynamic Content with Python, R, Julia, and Observable.
- Create professional-grade content, including articles, reports, presentations, websites, blogs, and books, in a variety of formats such as HTML, PDF, MS Word, ePub, and more.
- Author with scientific markdown, including equations, citations, crossrefs, figure panels, callouts, advanced layout, and more.
- Create interactive tutorials and notebooks.

4.2.1 Installing

To use quarto you don't need any special software, if you would like you can use even use a text editor to create your .qmd files and command line to render the document. However, working in IDE will make your life much easier. I prefer to use RStudio, after all, Quarto is a product of Posit (former RStudio), but it Visual Studio Code also works great. If you have not installed RStudio nor Visual Studio Code yet follow INSERT CHAPTER LINK.

4.2.1.1 RStudio

From within Rstudio you can install quarto as you install any other package. Additionally, you should install TinyTeX, a minimal set of packages required to compile LaTeX documents. You can both by running the following command in R.

```
install.packages(c("quarto", "tinytex"))
```

4.2.1.2 Visual Studio Code

Install the Quarto extension by going to Extensions -> search for "Quarto" -> install. Then run the following command in the terminal if you don't have TinyTeX installed:

```
quarto install tinytex
```

4.2.2 Your First Document

Write Create a quarto project. Go to top panel -> File -> New Project -> Directory -> Quarto Project -> Create

File -> Quarto Document -> Write the Title of your document -> Select the format you want (HTML is the default) -> Create

An (optional) YAML header demarcated by three dashes (—) on either end. you can learn more about YAML on Quarto's website. There a lot useful option for displaying.

```
---
title: "Title"
editor: visual
format: pdf
---
```

If you prefer WYSIWYG style you can switch to visual editor. in the top left corner. Additionally, if you rick "Render to Save" new document preview will be updated after each save.

Now it is time to get writing! To start, H1 heading can be defined in yaml `title: "Title"` and as `# Title`.

Markdown Syntax	Output
# Header 1 ## Header 2	# Chapter {.heading-output} ## 4.3 Section
### Header 3	### 4.3.1 Subsection
#### Header 4	#### 4.3.1.1 Subsubsection
##### Header 5	##### 4.3.1.1.1 Paragraph
###### Header 6	###### 4.3.1.1.1.1 Subparagraph
italics **bold** superscript^2^ <https://nber.org> [NBER](https://nber.org)	*italics* **bold** superscript^2 https://nber.org NBER
![caption](monalisa.jpeg)	

Markdown Syntax	Output

* unordered list
* Item 2
 	
  • sub-item 1
 	
  • sub-sub item
	• unordered list
	• item 1
	
	  – sub-item 1
	
	  ∗ sub-sub-item 1

1. ordered list
2. item 2
 	
  i) sub-item 1 A. sub-sub-item 1
	1. ordered list
	2. item 2
	
	  i) sub-item 1 A. sub-sub-item 1

inline math: $E = mc^{2}$ display math: $$E = mc^{2}$$	inline math: $E = mc^2$ display math: $$E = mc^2$$

If you want to add a line break add a line break add an empty line between your paragraphs, otherwise it will continue as the same text.

Use '...' to add inline code: `print(hi, friend)`

Use "' to delimit blocks of source code:

```
```
print(hi, friend)
```
```

Making tables in Markdown is not complicated. The most frequently used table is pipe table it lets you se alignment and captions with `:`. Tables can get complicated pretty quickly, if you ever get stuck refer to quarto's table documentation.

```
| Default    | Left    | Middle | Right   |
|------------|:--------|:------:|--------:|
| Hola       | Gable   | 3.141  | Nile    |
| Bonjour    | Temple  | 2.718  | Amazon  |
| Konnichiwa | Chaplin | 4.123  | Yangtze |

: Table Demonstration
```

Table 4.2: Table Demonstration

	Default	Left	Middle	Right
Hola		Gable	3.141	Nile
Bonjour		Temple	2.718	Amazon
Konnichiwa		Chaplin	4.123	Yangtze

If you ever feel lost or struggle with formatting, consider using the visual editor. It provides a familiar interface and is particularly useful for creating and previewing tables. To adjust options, for example the number of list options, simply click on the circle icon with an ellipsis next to it, and a selection menu will appear. In addition to this, the visual editor offers extensive customization options for other elements, such as images and tables.

```
::: {.callout-note}
You can copy paste (Ctrl + C; Ctrl + V) a picture in Visual Mode!
:::
```

You can learn more at Quarto's website.

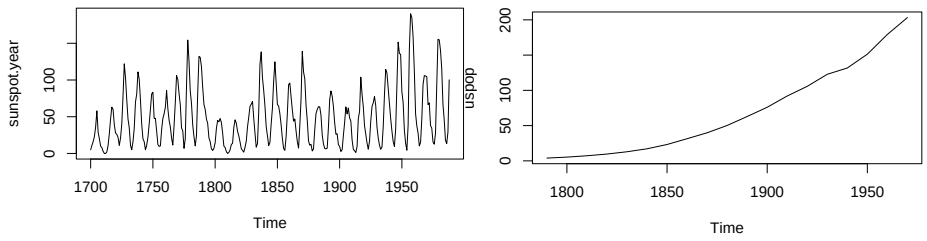
Chapter 5

Layout and References

Knitr is a package that takes care of the middle step between evaluating code and producing pdf/html. The package run your code and places its output into a final markdown file, which is later converted by pandoc.

Knitr lets you sen cell options that influence code blocks' execution and output. They are put at the top of a block within comments. For example:

```
plot(sunspot.year)
plot(uspop)
```



(a) Sunspot

(b) US Population

Figure 5.1: Plots

There is large number of options, but I will show the most commonly used ones. To begin in Figure **??**, `label` is a unique id for a code cell, which can be referred to in text with `@fig-plots`. Similarly, you can refer to tables, chapter, and files. `fig-cap` defines a caption for the entire plot. `fig-subcap` gives the two plots their individual sub-captions. `layout-ncol` let's us display our plots, pictures, etc. in separate columns. And `plot()` makes the plots. If you would like your code to fold use `code-fold = true,` above option `show` was used to have it opened by default. `code-summary` defines text for collapsed code blocks.

Another common options to use within code blocks are:

Table 5.1: from R Markdown Cheat Sheet

Option	Value	Explanation
eval	true	Whether to evaluate the code and include its results
echo	true	Whether to display code along with its results
warning	true	Whether to display warnings
error	false	Whether to display errors
message	true	Whether to display messages
include	true	Prevents any output (code, results) from being included
tidy	false	Whether to reformat code in a tidy way when displaying it
results	"markup"	Type of output format: "markup", "asis", "hold", or "hide"
cache	false	Whether to cache results for future renders

`tidy: true` is super useful once you want to include code inside your document as it will format it nicely.

5.1 Div Blocks

If you are familiar with HTML you will recognize blocks. You can add div blocks with wrapping text in `:::` or more semicolons. It is useful when you want put pictures in a grid. Here is a simple example:

```
::: {layout-ncol="2"}
![Surus](surus.png)

![Hanno](hanno.png)
:::
```

It must be separated from the preceding and following blocks by blank lines. Divs can be nested inside other Divs. For example, here we put a note and some text onto the margin.

```
::::: {.column-margin}

::: {.callout-note}
```

```
Note!
:::

More text
:::::
```

The short code enables you to insert a native page break into a document that will be compatible with all the other formats:

Because R, YAML, HTML, LaTeX have different notations for commenting. So, the one that will work universally within quarto is HTML's `<!-- comment here -->`.

Note

Here is a

More content

5.2 Diagrams

You can also create beautiful UML (Unified Modeling Language) diagrams within quarto with Mermaid and Graphviz. The flow chart below was maid with Mermaid!

```
flowchart LR
  A[Hard edge] --> B(Round edge)
  B --> C{Decision}
  C --> D[Result one]
  C --> E[Result two]
```

This might be your first time hearing that there is a language behind diagrams. UML is a standard graphical notation to describe software designs. It is a powerful tools for planning, visualizing and documents your projects. There are different types of diagrams to depict structures, behaviors and interactions with the standard set of symbols and notation. We will meet some of them in the chapter on Relational Databases.

5.3 Citations

> "Proper citation adds credibility to your work and acknowledges the work of others." - *Chat GPT*

Adding citations to your work shouldn't be stressful or confusing. With Quarto's seamless integration with Zotero, you can easily add citations in your preferred style and create a reference list, all without hassle. How cool is that? I think pretty cool.

Quarto utilizes Pandoc to generate citations and bibliographies in your preferred style. To source your citations, you'll need a .bib or .bibtex file, and optionally a .csl file for the citation style. Simply, add `bibliography: references.bib` to you YAML header in `_quarto.yml`.

```
bibliography: references.bib
```

You can easily cite your article using `@yourcitation9999`. Visual mode also provides suggestions, and entering the article's DOI will help locate and insert it even if it is not in your bibliography. For more information on citation methods, see Quarto Citation and Pandoc Citations.

Markdown Format	Output (author-date format)
@abadie2017 says cluster you SE.	Abadie et al. (2017) says cluster you SE.
Some thing smart [@abadie2017; @bai2009].	Some thing smart (Abadie et al. 2017; Bai 2009).
Abadie says cluster [-@abadie2017].	Abadie says cluster (2017).

If you've successfully created your bibliography in Zotero, adding citations to your document will be a breeze. Simply start typing and Zotero will suggest citations to add to your bibliography file. For a paper with more than 10 citations, I recommend using Better Bibtex, which allows you to connect citation keys to the paper as you write, just make sure Zotero is open.

To generate your citations from a document (say cited in Obsidian) without having to re-cite everything, you can use the `bbt_update_bib()` function from the rbbt package. Ensure that Zotero is running and that you're in the markdown document where you want to update citations. Run the `bbt_update_bib()` function to create a bibliography, and specify any additional arguments as needed.

```
bbt_update_bib(
  path_rmd, # Path to your Markdown document.
  path_bib = bbt_guess_bib_file(path_rmd), # Path to the references.bib file
  translator = bbt_guess_translator(path_bib), # type of bibliography file to genera
)
```