Reproducible reporting

An introduction to Quarto

Janick Weberpals, RPh, PhD
jweberpals@bwh.harvard.edu
Division of Pharmacoepidemiology and Pharmacoeconomics
Brigham and Women's Hospital
Harvard Medical School

August 25, 2024

Table of contents

Problem statement

> *Wait, but how was that done exactly?*

Problem statement (i)

> *Wait, but how was that done exactly?*

- More often than not, statistical and computational methods are reported and phrased ambiguously, e.g.,

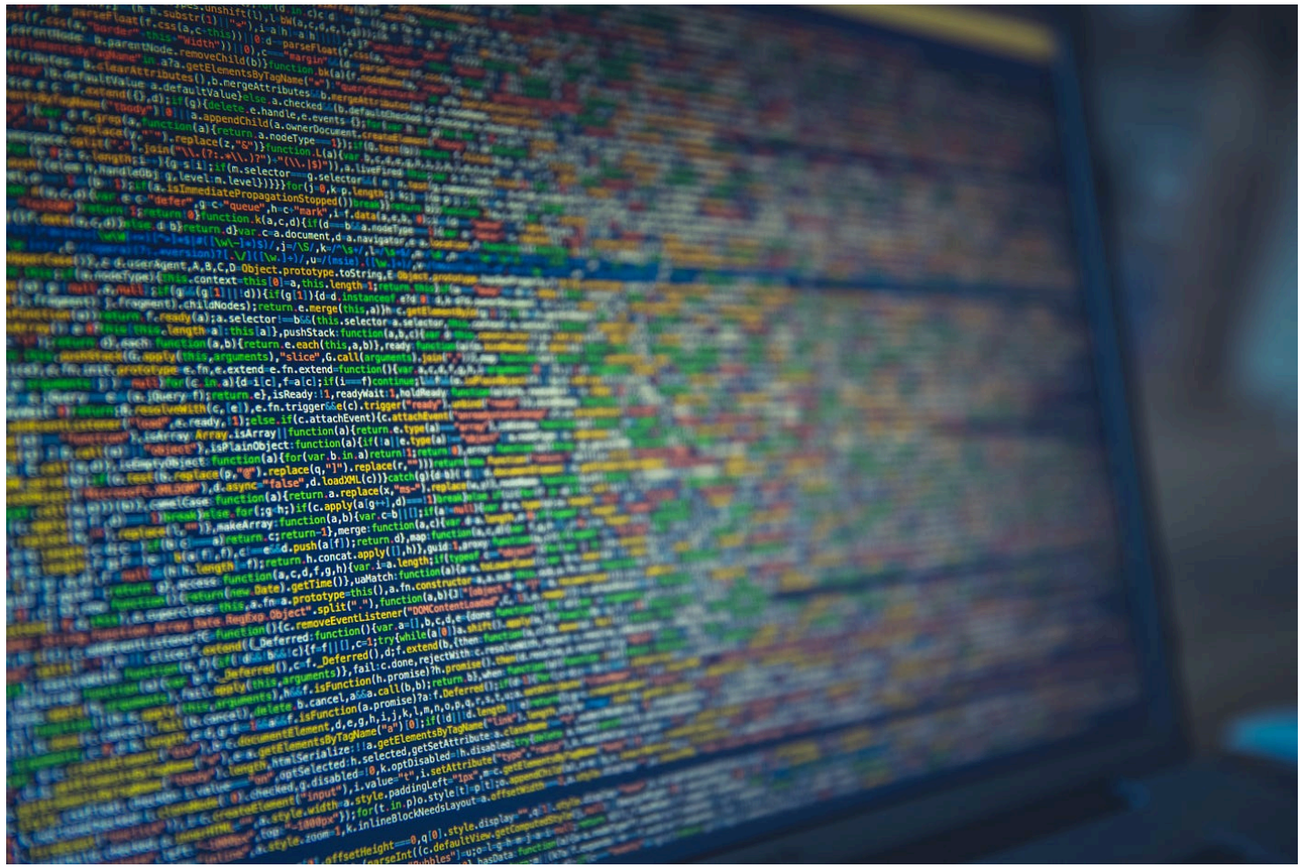
  > *"We measured the pre-exposure performance status within 90 days of the index date."*

- Does the 90-day window include or exclude the index date? What was done if there were multiple performance assessments per patient? …
- Take a moment and reflect if you would be able to exactly reproduce a study you published 10 years just based on the paper's methods section?

Problem statement (ii)

> *Wait, but how was that done exactly?*

One could find the details in the analytical programming code, BUT…



Is there a reproducibility crisis?

Nature survey 2016: More than 70% of researchers have tried and failed to reproduce another scientist's experiments, and more than half have failed to reproduce their own experiments[1]

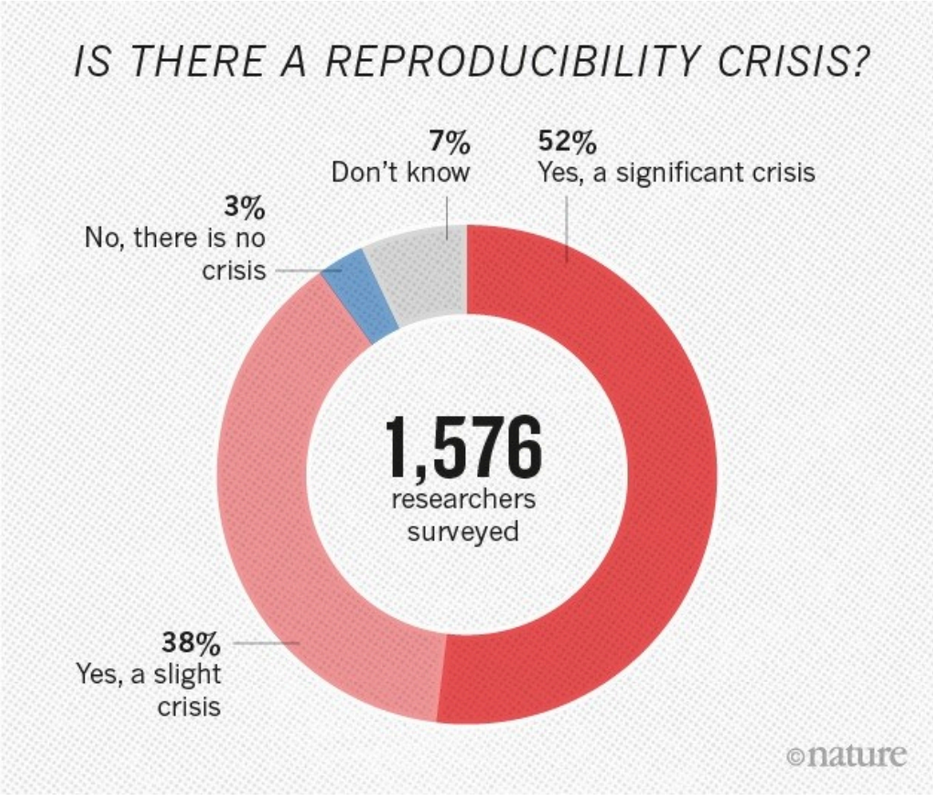


Figure 1: Nature survey on reproducibility issues in science.

What if…

What if…

If there was just a way to combine…

- the narrative prose that explains the methods used
- the analytic code we implemented to execute these methods
- the corresponding results

…all in one report?

Literate programming

> *Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do* (Donald Knuth, Turing Award recipient)

Definition

It is basically an annotated, executable manuscript!

Literate programming

Programming paradigm introduced in 1984 by Donald Knuth in which a computer program is given as an explanation of how it works in a natural language, such as English, interspersed (embedded) with snippets of macros and traditional source code, from which compilable source code can be generated.

[2]

In other words…

Example

Methods section text:

"A propensity score model for exposure initiation was fit using logistic regression with age, sex and smoking as covariates. Patients were matched using nearest neighbor matching on the propensity score in a 1:1 ratio without replacement targeting the average treatment effect among the treated (ATT)."

```r
MatchIt::matchit(
  formula = exposure ~ age_num + female_cat + smoking_cat,
  data = smdi::smdi_data,
  ratio = 1,
  method = "nearest",
  distance = "glm",
  link = "logit",
  estimand = "ATT",
  replace = F
  )
```

```
A matchit object
 - method: 1:1 nearest neighbor matching without replacement
 - distance: Propensity score
             - estimated with logistic regression
 - number of obs.: 2500 (original), 1996 (matched)
 - target estimand: ATT
 - covariates: age_num, female_cat, smoking_cat
```

History of literate programming

- Literate programming is a concept pioneered by Donald Knuth, a Turing Award recipient known for creating TeX.
- The main idea behind the early form of literate programming was to upend the traditional programming practices of the time by systematically including human readable text accompanying and explaining the logic and the purpose of a program.
- As he describes in "Literate Programming", Knuth considers the programmer as an "essayist" who should strive to communicate the purpose of a program in order to create better code.
- While initially centered in the domain of computer science, it more recently resurged in the interdisciplinary world of "data science".

https://bernhardbieri.ch/blog/2022-08-25-litteralprogramminginstata/

Popular legacy frameworks - RMarkdown

- RMarkdown is a powerful open-source tool for combining analysis and reporting into the same document
- Can be used with few programming languages, but mostly focused on R programming language

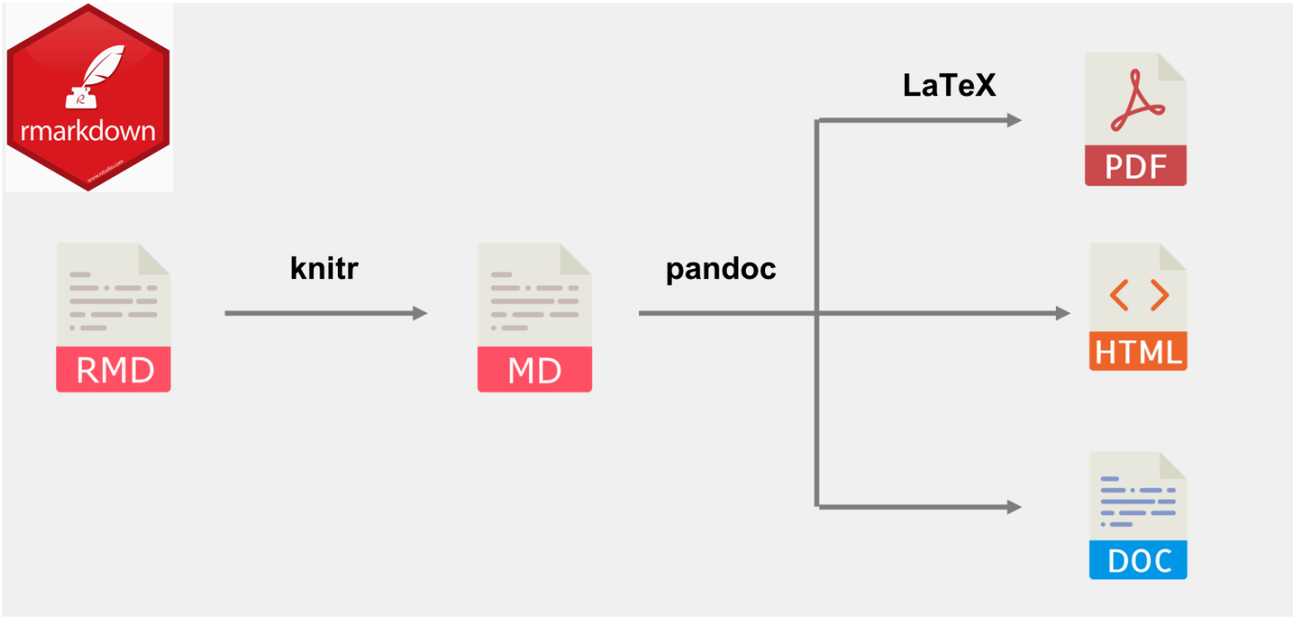


Figure 2: Diagram illustrating how an RMarkdown document is converted to the final output document.

Popular legacy frameworks - RMarkdown

```
---
title: "RMarkdown example"
output: html_document
author: Janick Weberpals
date: "2024-08-01"
---

{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)

new_users <- smdi::smdi_data_complete |>
  dplyr::select(
    dplyr::ends_with("cat"),
    dplyr::ends_with("num"),
    exposure
    )

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

{r mean_age}
library(smdi)
library(gtsummary)

mean(new_users$age_num)

## Including Plots

You can also embed plots, for example:

{r pressure, echo=FALSE}
hist(new_users$age_num)

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

RMarkdown example

Janick Weberpals

2024-08-01

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

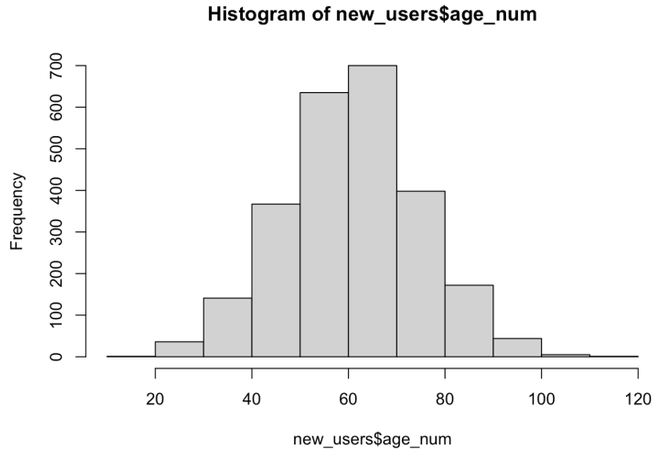
```
library(smdi)
library(gtsummary)

mean(new_users$age_num)
```

```
## [1] 60.90256
```

Including Plots

You can also embed plots, for example:



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

Figure 3: Example of an RMarkdown report and the corresponding output

Popular legacy frameworks - Jupyter

- Project Jupyter is a non-profit, open-source project, born out of the IPython Project in 2014 as it evolved to support interactive data science and scientific computing across all programming languages
- Most often seen to be used by Python programmers

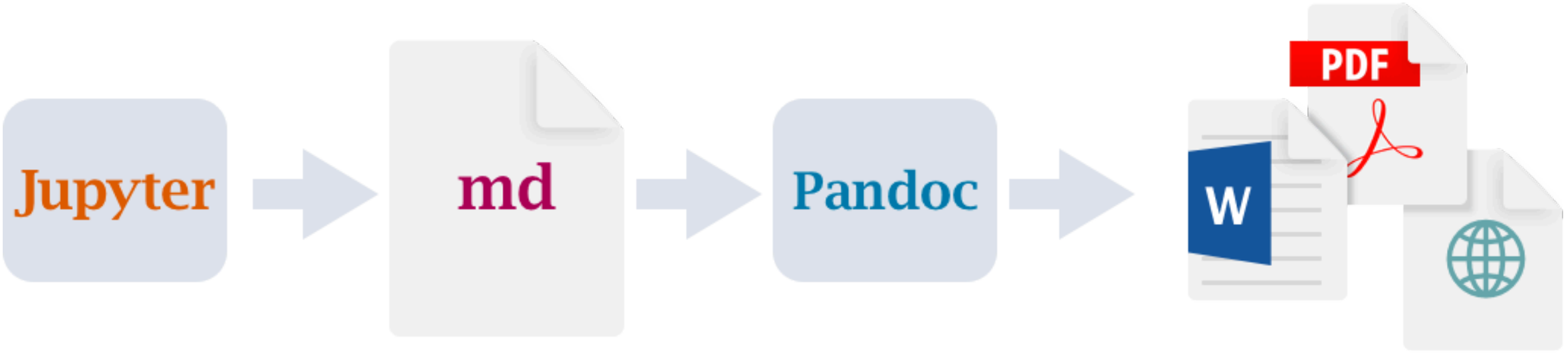


Figure 4: Illustration of Jupyter notebook workflow

Popular legacy frameworks - Jupyter

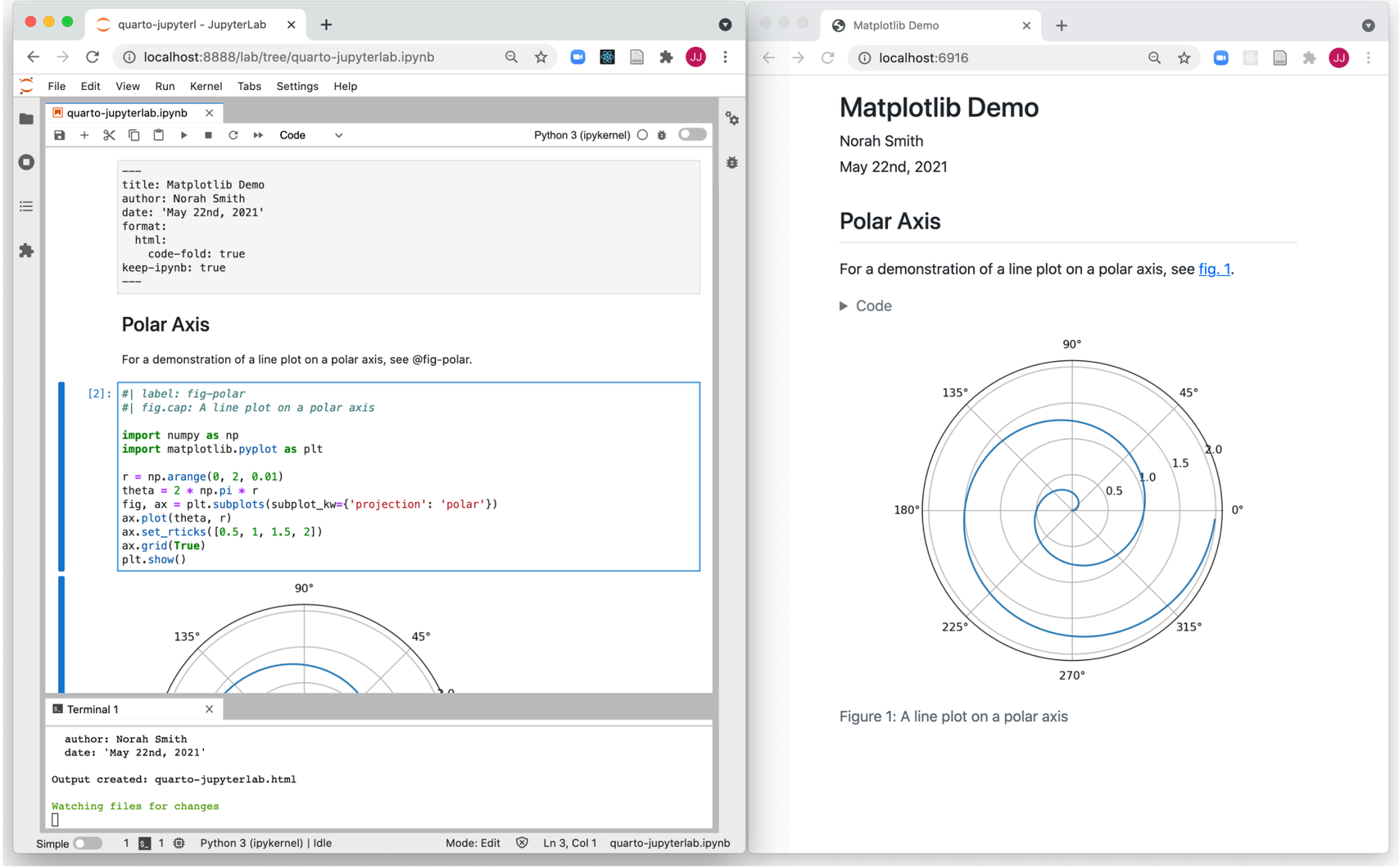
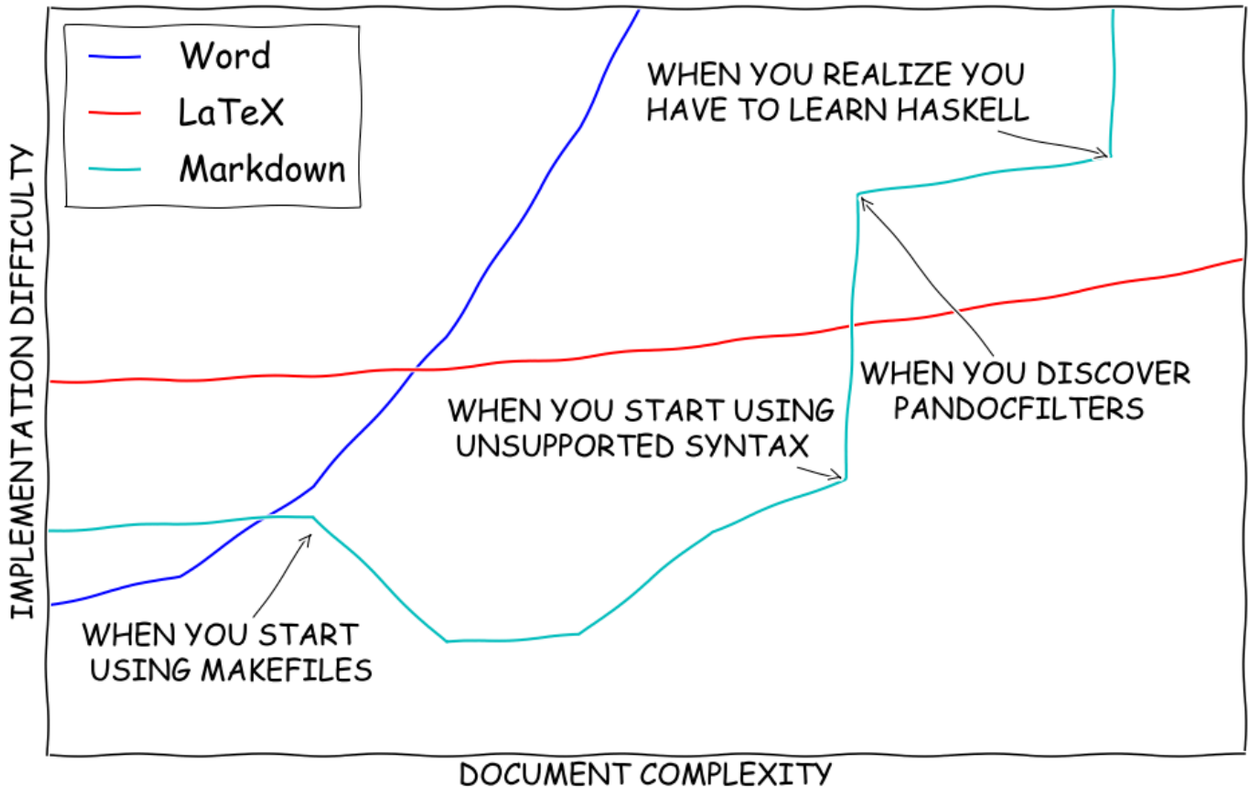


Figure 5: Example of a Jupyter Notebook report and the corresponding output

Document complexity of technical report

Strengths and weaknesses of technical reporting systems



Quarto

Dynamic study reporting

Introduction to Quarto

- An open-source scientific and technical publishing system

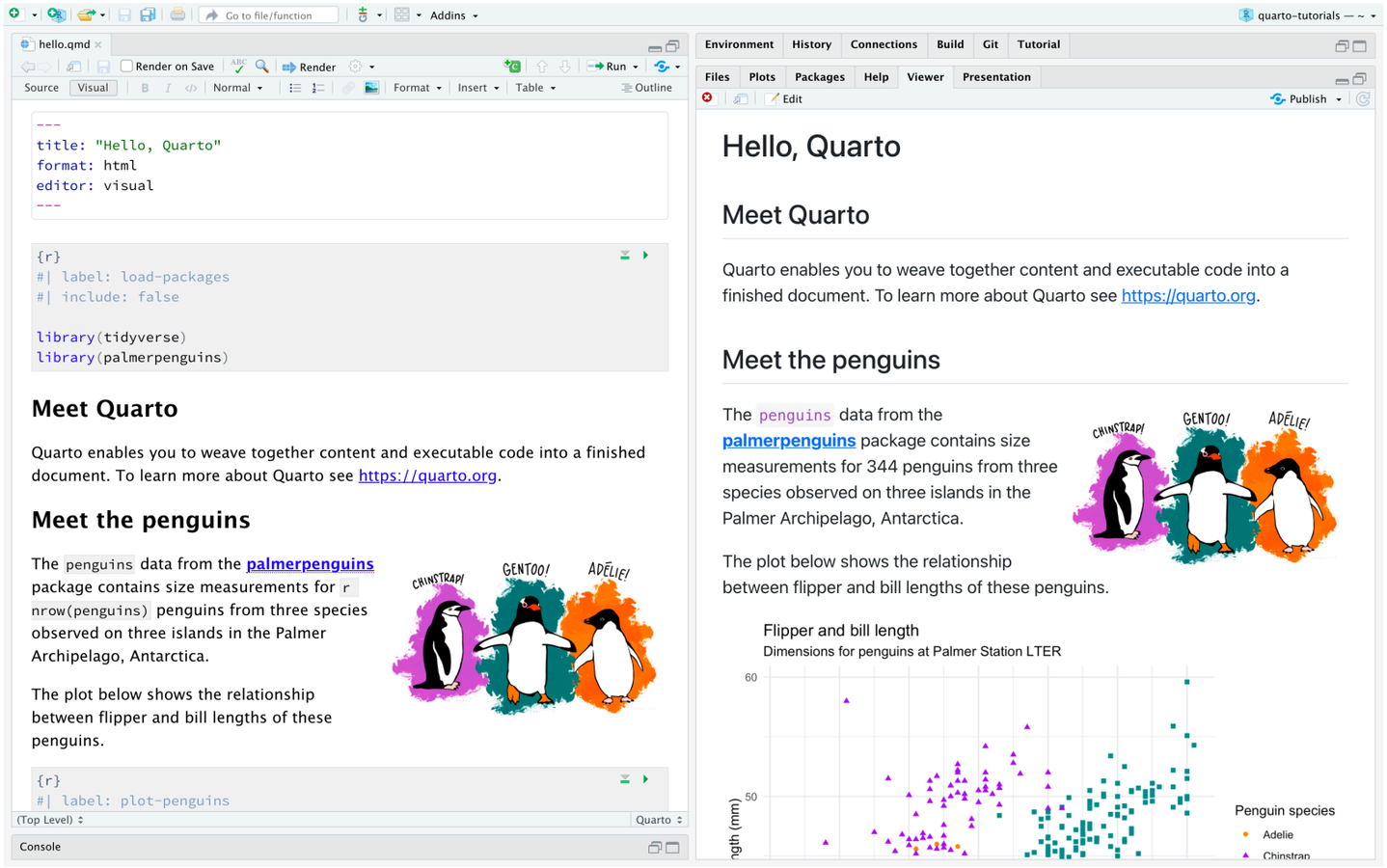


Figure 6: Side-by-side example of a Quarto document (left) and a rendered .html output

Introduction to Quarto

- Unifies the functionality of many tools, packages and open source platforms into a single consistent system
- Extends it with native support for a large number of open-source programming languages (R, Python, Julia, Stan, C++, etc.)
- Can be used with most common code editors (RStudio, Jupyter, VSCode, etc.)
- Proprietary programming languages (SAS, STATA) can also be integrated but require some additional setup
  - Additional resources for use of Quarto with SAS (setup, demo) and STATA can be found on the course website

Goal: single source publishing

- Since Quarto is a single source reporting system, we are not constraint to only output one document type but multiple given the same source document
- Example: Manuscript written for a journal, we can also render it into a website

Figure 7: From raw code and text to an elegant research report using Quarto

Quarto - ingredients for a research report (i)

- First we provide **metadata** about the project in a so called `YAML header` (Yet Another Markdown Language)
- Documentation on all YAML options can be found on https://quarto.org/docs/reference/

```yaml
---
title: "My RWE study report"
author: "Janick Weberpals"
date: last-modified
toc: true
code-fold: true
number-sections: true
bibliography: references.bib
csl: pharmacoepidemiology-and-drug-safety.csl
format:
  html: default
  docx: default
  pdf: default
---
```

Quarto - ingredients for a research report (ii)

Plain text

- To describe objectives, methods (just like in a .docx document)
- Achieved via Markdown syntax
- ✅ Universal and reproducible formatting across output document types
- ❌ Syntax needs to be known (although many modern editors come with a GUI)

syntax

```
Plain text
End a line with two spaces to start a new paragraph.
*italics* and _italics_
**bold** and __bold__
superscript^2^
~~strikethrough~~
[link](www.rstudio.com)

# Header 1

## Header 2

### Header 3

#### Header 4

##### Header 5

###### Header 6

endash: --
emdash: ---
ellipsis: ...
inline equation: $A = \pi*r^{2}$
image: ![](path/to/smallorb.png)

horizontal rule (or slide break):

***

> block quote

* unordered list
* item 2
    + sub-item 1
    + sub-item 2

1. ordered list
2. item 2
    + sub-item 1
    + sub-item 2

Table Header  | Second Header
------------- | -------------
Table Cell    | Cell 2
Cell 3        | Cell 4
```

becomes

Plain text
End a line with two spaces to start a new paragraph.
italics and *italics*
bold and **bold**
superscript^2
~~strikethrough~~
link

Header 1
Header 2
Header 3
Header 4
Header 5
Header 6

endash: –
emdash: —
ellipsis: …
inline equation: $A = \pi * r^2$
image: 
horizontal rule (or slide break):

> block quote

- unordered list
- item 2
  - sub-item 1
  - sub-item 2

1. ordered list
2. item 2
   - sub-item 1
   - sub-item 2

Table Header	Second Header
Table Cell	Cell 2
Cell 3	Cell 4

Quarto - ingredients for a research report (iii)

- `Code chunks` make it possible to blend plain text, programming code and the corresponding output, e.g.,

> **Methods section text:**
>
> *"The propensity score was defined as the probability of each patient to initiate the exposure based on observed baseline covariates including age, sex and smoking."*

Code chunk following description of propensity score model:

```
1 # define the model fit
2 ps_fit <- as.formula(exposure ~ age_num + female_cat + smoking_cat)
3 ps_fit
```

```
exposure ~ age_num + female_cat + smoking_cat
```

Quarto - ingredients for a research report (iii)

- `Code chunks` are not limited to one programming language, but can flexibly accommodate multiple in the same document
- Language is chosen by the `{}` parameter at the beginning of each code chunk, e.g., `R` (left) and `Python` (right)

````
```{r}
# Load necessary libraries
library(dplyr)

# Set a random seed for reproducibility
set.seed(42)

# Simulate a dataset
n_patients <- 1000
data <- data.frame(
  patient_id = 1:n_patients,
  medication = sample(c('DrugX', 'DrugY'), n_patients, replace = T, prob
  adr = sample(c(0, 1), n_patients, replace = TRUE, prob = c(0.95, 0.05)
  )

# View the first few rows of the simulated data
head(data)
```
````

````
```{python}
# Load necessary libraries
import pandas as pd
import numpy as np

# Set a random seed for reproducibility
np.random.seed(42)

# Simulate a dataset
n_patients = 1000
data = pd.DataFrame({
    'patient_id': np.arange(1, n_patients + 1),
    'medication': np.random.choice(['DrugX', 'DrugY'], size=n_patients,
    'adr': np.random.choice([0, 1], size=n_patients, p=[0.95, 0.05])
    })

# View the first few rows of the simulated data
data.head()
```
````

Reporting elements

Figures

Quarto enables the integration, labeling and cross-referencing of `figures` using `@fig-missingness` which becomes Figure 8.

```
```{r}
#| label: fig-missingness
#| fig-cap: "Proportion of missingness among covariates with at least one unobserved value"

library(smdi)

smdi_vis(smdi_data)
```
```

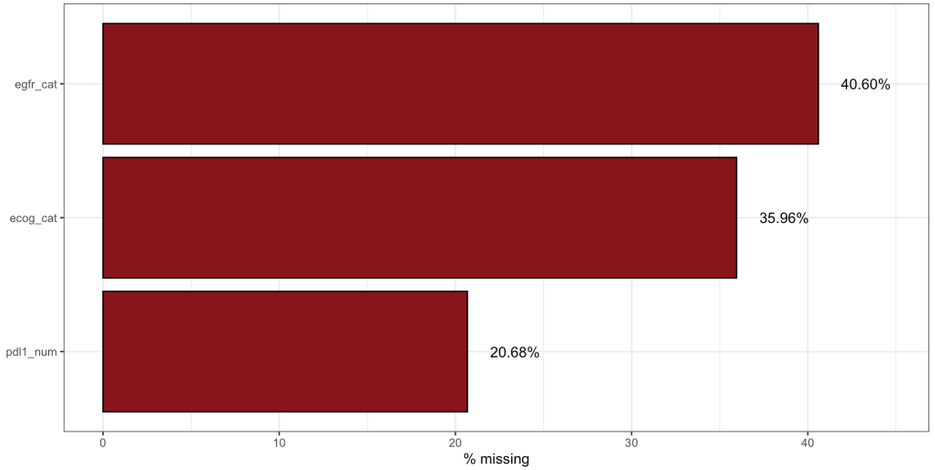
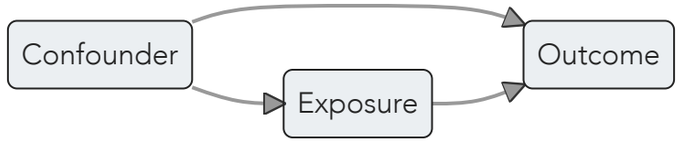


Figure 8: Proportion of missingness among covariates with at least one unobserved value

Flowcharts and diagrams

- Quarto has native support for embedding Mermaid and Graphviz diagrams
- This enables the depiction of flowcharts, sequence diagrams, state diagrams, gantt charts, and more using a plain text syntax inspired by markdown

````
```{mermaid}
flowchart LR
  C(Confounder) --> E(Exposure)
  C(Confounder) --> O(Outcome)
  E(Exposure) --> O(Outcome)
```
````



Tables

Similarly, we can also create, label and cross-reference `tables` using `@tbl-table1` which becomes Table 1.

```
```{r}
#| label: tbl-table1
#| tbl-cap: "Baseline patient characteristics."

library(gtsummary)

trial |>
  tbl_summary(by = trt, include = c(age, grade))
```
```

Table 1:
Baseline patient characteristics.

Characteristic	Drug A N = 98	Drug B N = 102
Age, Median (IQR)	46 (37 – 60)	48 (39 – 56)
Unknown	7	4
Grade, n (%)		
I	35 (36)	33 (32)
II	32 (33)	36 (35)
III	31 (32)	33 (32)

Equations

- Quarto also integrates LaTeX and styles to write equations, for example …

```
$$
\lambda(t | X) = \lambda_0(t) \exp(\beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p)
$$
```

… becomes:

$$\lambda(t|X) = \lambda_0(t) \exp(\beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p)$$

Referencing

Quarto has an in-built referencing system for which the referencing style can be chosen in the `YAML header` (.csl reference-style files see zotero.org/styles)

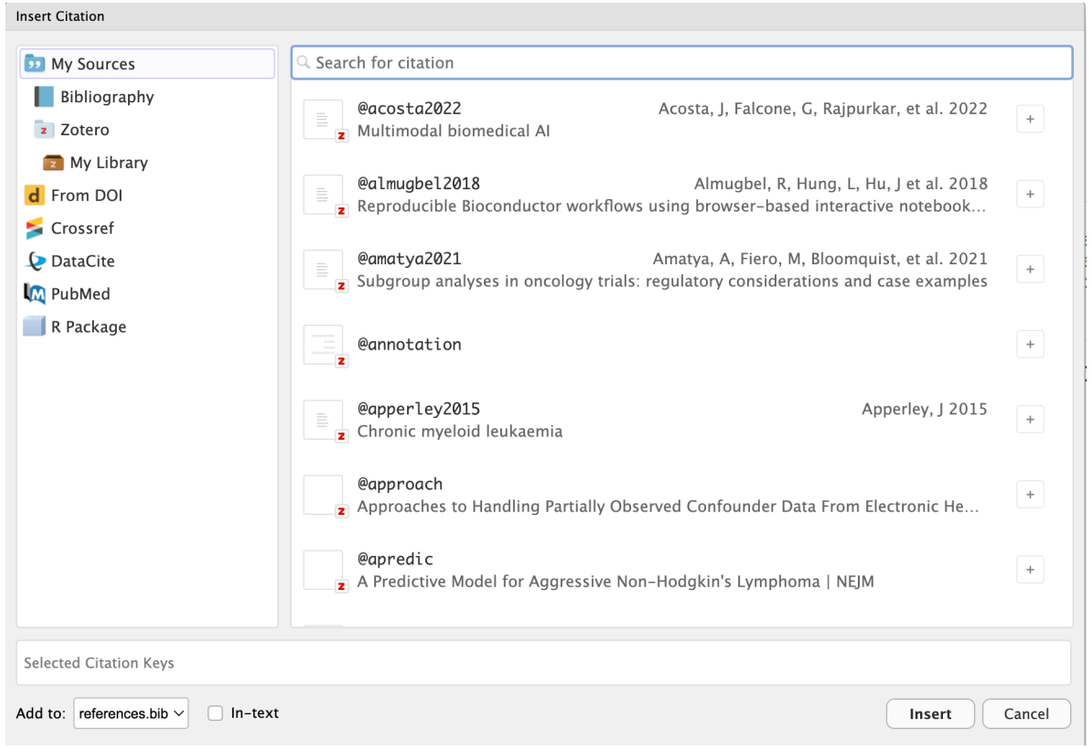


Figure 9: RStudio built-in reference manager

```yaml
---
title: "My RWE study report"
bibliography: references.bib
csl: pharmacoepidemiology-and-drug-safety.csl
format: docx
---

In the 2009 publication, Schneeweiss et al.
[@schneeweiss2009high] introduced the
concept of high-dimensional propensity scores.
```

becomes ⬇️

"In his 2009 publication, Schneeweiss et al.[3] introduced the concept of high-dimensional propensity scores."

Inline code

- Inline code allows to execute code within markdown text, e.g. to automatically use the most up-to-date computations embedded in narrative language.

````
```{r}
# we determine the sample size in a code chunk to integrate into our narrative report
sample_size <- nrow(smdi_data)
```

The sample size comprised `{r} sample_size` patients.
````

- In a dynamic study report, we now don't need to manually copy-paste numbers, but only reference the resulting object of the computations performed in the code chunk, which then automatically shows the most-up-to date number:

> The sample size comprised 2500 patients.

Interactive reports

If we render study reports to output formats that support interactive elements (e.g., .html), Quarto provides even more tools to make study reporting interactive

Tabsets

Tabset | Age distribution | Biomarker distribution

`Tabset` allows us to divide content into multiple tabs for interactive exploration

```
1
2  ::: panel-tabset
3
4  ## Age distribution
5
6  {code}
7
8  ## Biomarker distribution
9
10 {code}
11
12 :::
```

Code folding

Sometimes it may not be desired to see the entire code to enable readers to focus on the text and results but still have the ability to see the code

`code-fold` parameter | Output

```
#| code-fold: true

library(ggplot2)

dataset |>
  ggplot(aes(x = exposure, y = age_num, fill = exposure)) +
  geom_boxplot(alpha = 0.5) +
  theme_minimal() +
  labs(
    x = "Exposure status",
    y = "Years of age",
    fill = "Exposure"
  )
```

Parameterized reporting (i)

- RWE studies usually include many sensitivity analyses to check the robustness of certain assumptions and models in the main analysis
- Often it is just one or few parameters that have to bee changed
- Copy-pasting code back and forth is very error-prone and should be avoided

Case study

Let's say we do propensity score matching with a certain caliper and want to run a sensitivity analysis with a different caliper but we don't want to copy-paste any of the code

Parameterized reporting (ii)

- The `YAML header` that contains the metadata about the study report also has an option to define study parameters that can be flexibly changed
- Let's say we do propensity score matching with a certain caliper and want to run a sensitivity analysis with a different caliper

```yaml
---
title: "My RWE study report"
params:
  ps_caliper: 0.05
---
```

- The actual caliper is now replaced with `params$ps_caliper`

```
MatchIt::matchit(
  formula = exposure ~ age_num + female_cat + smoking_cat,
  data = smdi::smdi_data,
  caliper = params$ps_caliper
  )
```

Parameterized reporting (iii)

Now, the exact same script can be rendered using different parameters.

```
# load libraries
library(quarto)
library(here)

# render report with parameters

# main analysis with 5% caliper of propensity score standard deviation
quarto_render(
  input =  "analysis.qmd",
  metadata = list(title = "Main analysis"),
  execute_params = list(ps_caliper = 0.05)
  )

# sensitivity analysis with 1% caliper of propensity score standard deviation
quarto_render(
  input =  "analysis.qmd",
  metadata = list(title = "Sensitivity analysis I"),
  execute_params = list(ps_caliper = 0.01)
  )
```

Parameterized reporting (iv)

To render reports with parameters, you can also pass them on the command line/Terminal using the `-P` flag (this works for both `.ipynb` or `.qmd` files):

Terminal

```
quarto render analysis.qmd -P ps_caliper:0.05

quarto render analysis.qmd -P ps_caliper:0.01
```

Quarto-based research reports (i)

The ultimate goal…

- reproduce your entire study, from deriving analytic cohorts to the very last sensitivity analysis, upon rendering your quarto project
- have access to all computations embedded in a manuscript text or linked research report

Terminal

```
quarto render
```

Quarto-based research reports (ii)

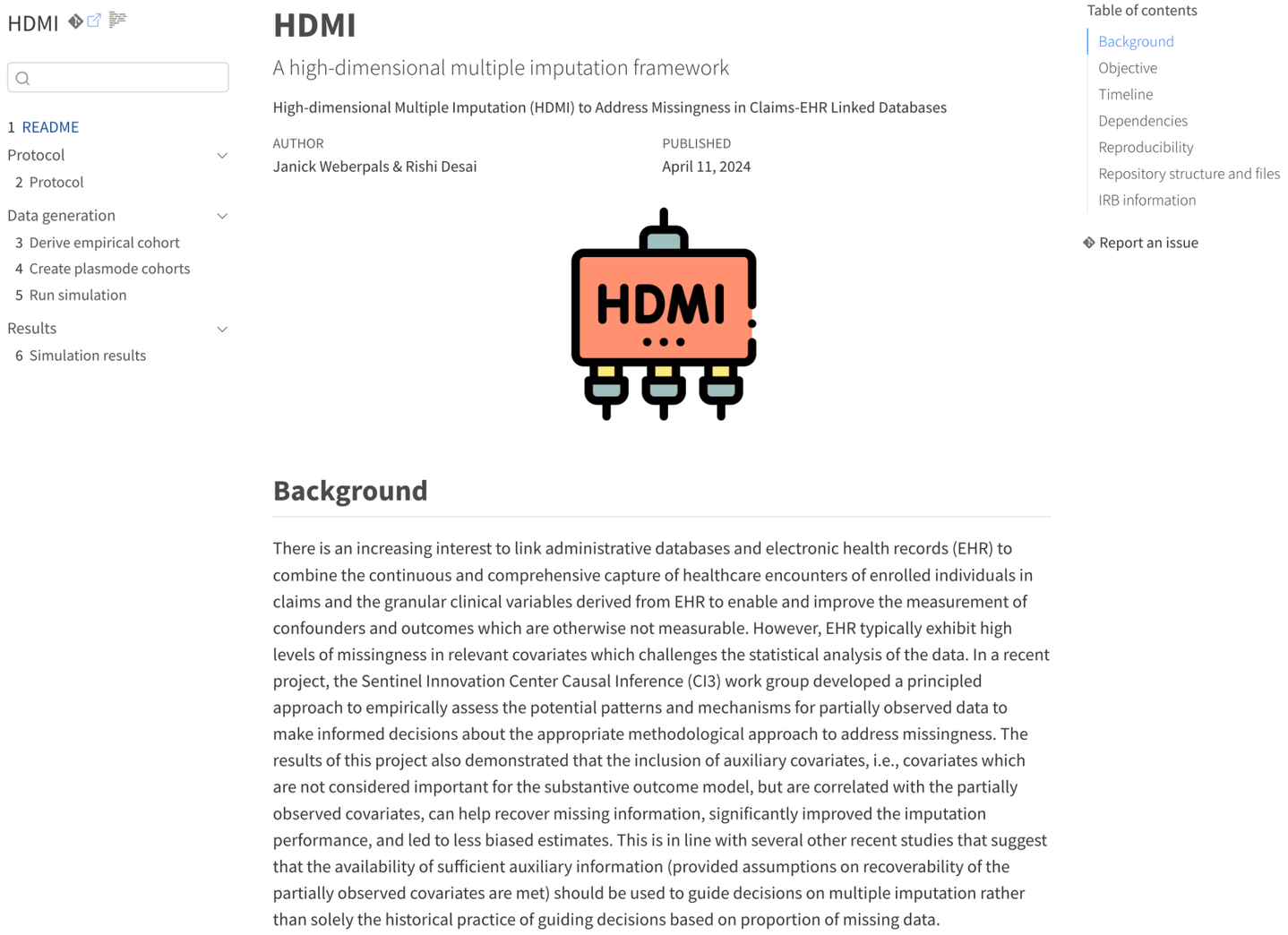


Figure 13: An example of a Quarto-based research report can be found in the course materials website

Quarto manuscripts

Quarto **manuscripts** (Quarto 1.4+), in addition to doing everything you can do with journal articles, can

- produce manuscripts in **multiple formats** (including LaTeX or MS Word formats required by journals), and give readers easy access to all of the formats through a website
- **publish computations** from one or more notebooks alongside the manuscript, allowing readers to dive into your code and view it or interact with it in a virtual environment
- Example of a reproduced NEJM trial[4]: https://mine-cetinkaya-rundel.github.io/indo-rct/

Quarto manuscript templates

Quarto Journal Templates

The `quarto-journals` organization collects a curated set of journal templates for Quarto.

Use a journal template with the command:

```
quarto use template quarto-journals/<template-name>
```

Journal / Publisher	Name	Install
Association of Computing Machinery	acm	`quarto use template quarto-journals/acm`
American Chemical Society	acs	`quarto use template quarto-journals/acs`
American Geophysical Union	agu	`quarto use template quarto-journals/agu`
Biophysical journal	biophysical-journal	`quarto use template quarto-journals/biophysical-journal`
Elsevier Journals	elsevier	`quarto use template quarto-journals/elsevier`
American Statistical Association Journals	jasa	`quarto use template quarto-journals/jasa`
Journal of Statistical Software	jss	`quarto use template quarto-journals/jss`
Public Library of Science	plos	`quarto use template quarto-journals/plos`

Figure 14: The quarto-journals organization collects a curated set of journal templates for Quarto.

Quarto summary

Quarto …

- Is a technical publishing system compatible with most programming languages and editors
- Is a single source reporting system that can produce many different types of outputs (.docx, .pdf, .html, websites, presentations, etc.)
- Main ingredients: YAML header (metadata), text, code chunks
- Can be used as to blend narrative text, programming code and output in one document
- Has all capabilities of common reporting systems (e.g., MS Word) and many that go beyond (inline coding, dynamic and interactive elements, …)
- Is a useful tool to parameterize and streamline analysis pipelines avoiding error-prone copy-pasting

Curious?

Test it out yourself with the `Hello, Quarto` tutorial:

https://quarto.org/docs/get-started/hello/rstudio.html

Further resources

- Getting started with Quarto (download and installation)
- Introduction to Quarto (R for Data Science)
- Basic Quarto workflows
- Reproducible Manuscripts with Quarto

References

1. Baker M. 1,500 scientists lift the lid on reproducibility. *Nature* 2016; **533**: 452–454. doi:10.1038/533452a.
2. Knuth DE. Literate programming. *The computer journal* 1984; **27**: 97–111.
3. Schneeweiss S, Rassen JA, Glynn RJ, Avorn J, Mogun H, Brookhart MA. High-dimensional propensity score adjustment in studies of treatment effects using health care claims data. *Epidemiology* 2009; **20**: 512–522.
4. Elmunzer BJ, Scheiman JM, Lehman GA, *et al.* A randomized trial of rectal indomethacin to prevent post-ERCP pancreatitis. *New England Journal of Medicine* 2012; **366**: 1414–1422.