Tools of the Trade

HES 505 Fall 2025: Session 2

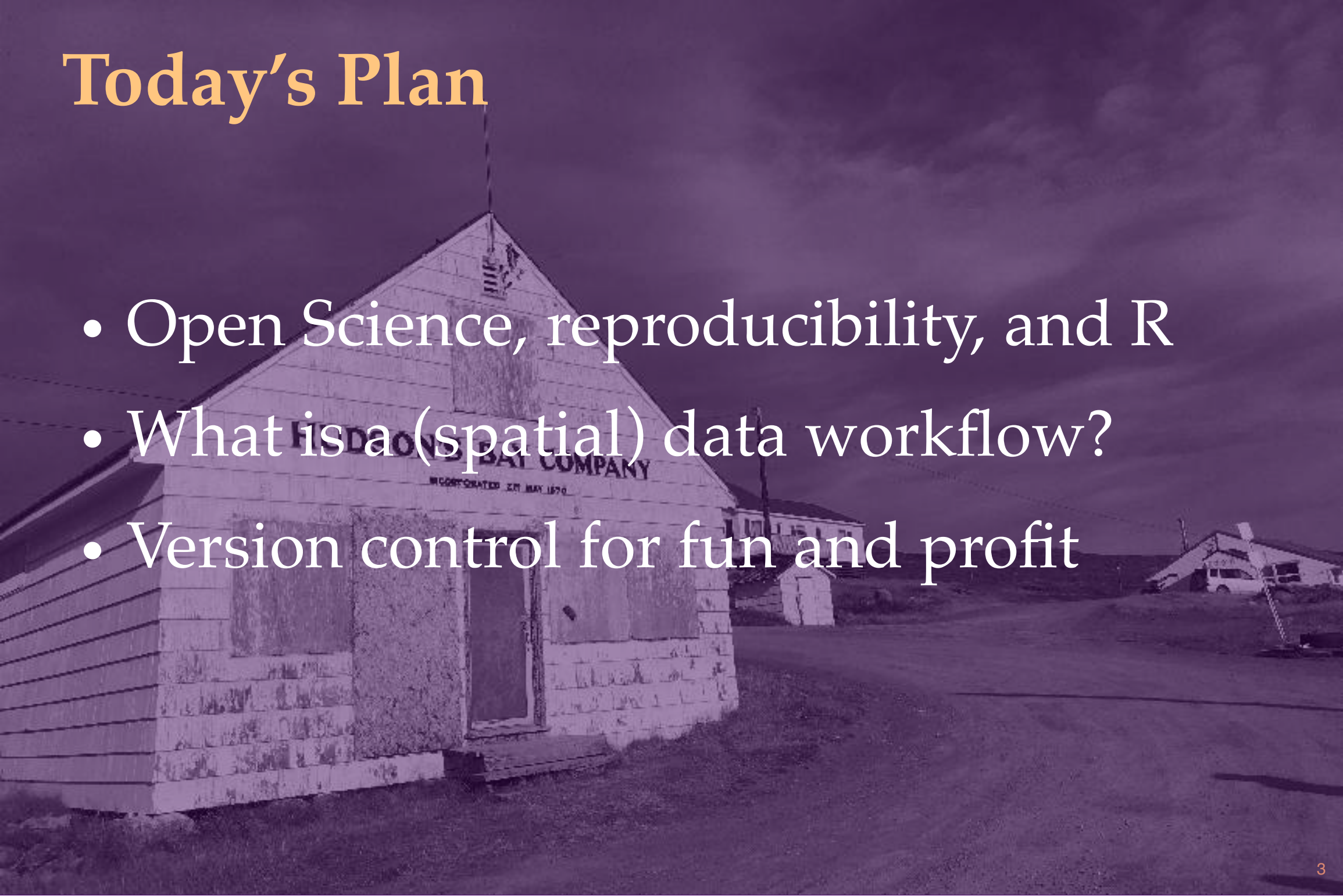
Matt Williamson

Checking in

1. What can I clarify about the course?
2. Are there any challenges you can already see?

Today's Plan

- Open Science, reproducibility, and R
- What is a (spatial) data workflow?
- Version control for fun and profit



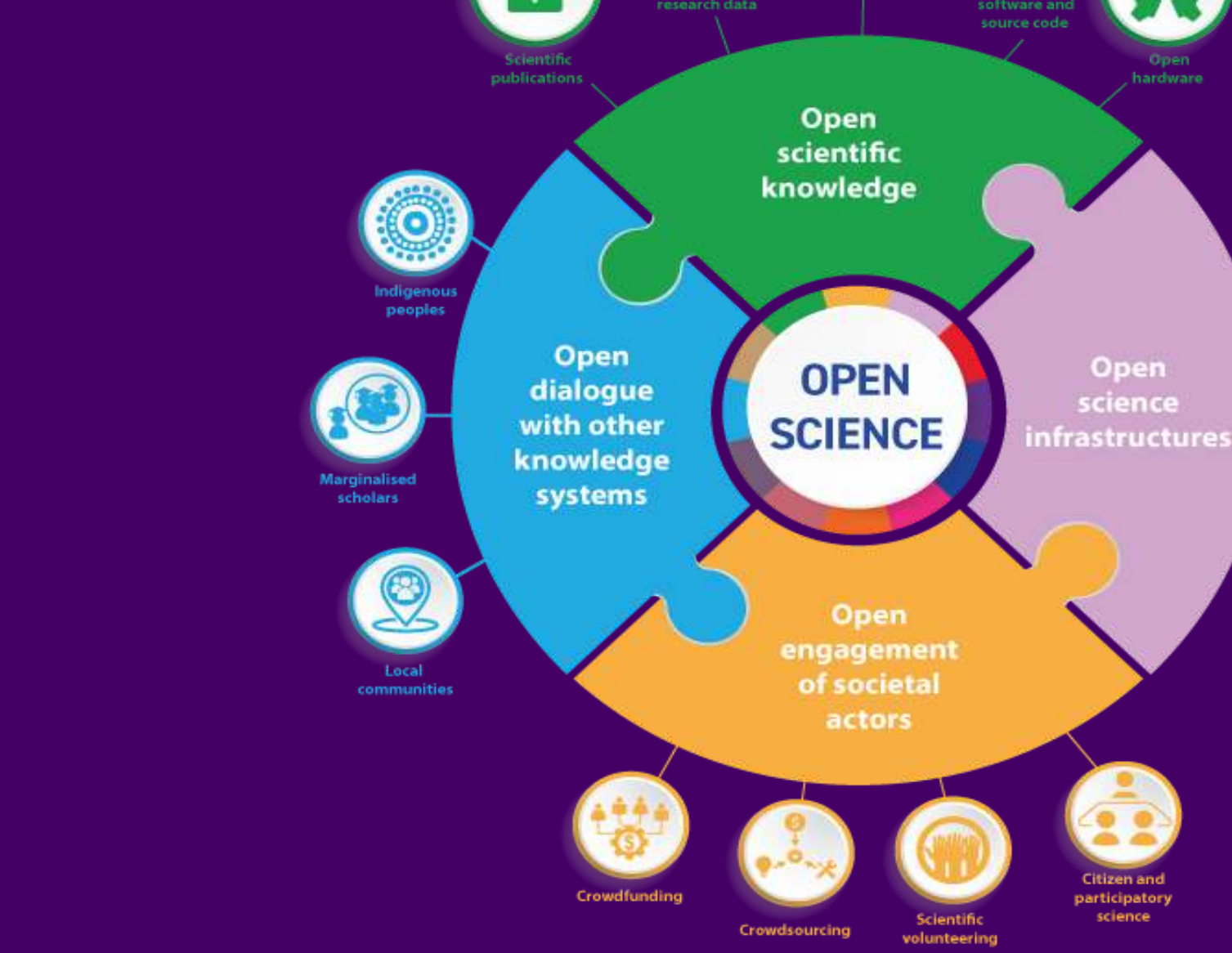
A More Democratic Science?

What is open science?

cientific research
nd its outcomes
eely accessible to
l

ccelerate research
ND improve
ublic trust

ur focus: Open
ource software and
ode



UNESCO.org, CC BY-SA 4.0, via Wikimedia Commons

Why open source software and code?

- Future-proof: OSS development is fast and ongoing
- Interoperability: Work across hardware types, integrate new software easily
- Free!! (To use and maintain)
- Sharing code and data enables innovation and *reproducibility*

Why (not) R?

-)pen Source
- Huge useR community
- ntegrated analysis pipelines
- eproducible workflows

- Coding can be hard…
- Memory challenges
- Speed
- Decision fatigue

Anatomy of an R session

Moving beyond **R**ead-**E**val-**P**rint **L**oops

`cripts`: contain a ecord of the code in our analysis and the *bjects* you created

`unctions`: perform perations on objects

`ackages`: collections of elated functions

Code | Plot

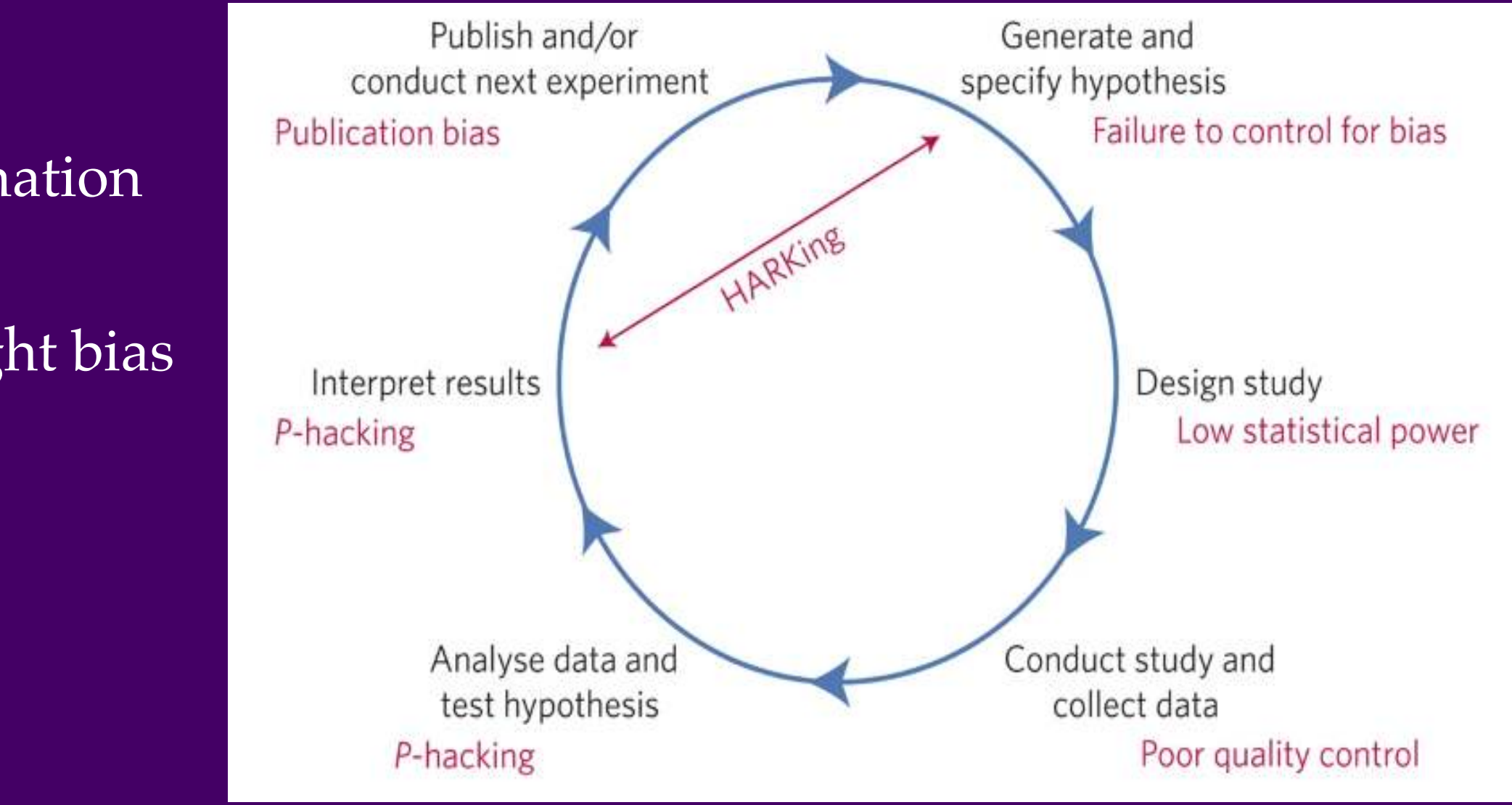
```
library
library
library
             <- c "#2E74C0"  "#CB454A"
          <- map_data "state"
        $       <- tolower         $
              <- left_join
   <- ggplot data =
             mapping = aes x =       y =
                           group =
                           fill =
   <-    + geom_polygon color = "gray90"
                        size = 0.1  +
    coord_map projection = "albers"
              lat0 = 39  lat1 = 45
   <-    + scale_fill_manual values =                    +
    labs title = "Election Results 2016"
         fill = NULL
```

Reproducible workflows

Science is a social process!!

Why Do We Need Reproducibility?

- oise!!
- onfirmation ias
- indsight bias



What do we mean by reproducible "workflow"?

Reproducibility and your code

- Scripts: may make your code reproducible (but not your analysis)
- Commenting and formatting can help!
- Think about future you…

```
```{r}
#| eval: false
## load the packages necessary
library
## read in the data
                <- read_csv "/Users/mattwilliamson/Google Drive/My Drive/TEAC

## How many in each feature class
table                $
```
```

Reproducible scripts

- Comments explain what the code is doing
- Operations are ordered logically
- Only relevant commands are presented
- Useful object and function names
- Script runs without errors (on your machine and someone else's)

Flipping the script

Toward Efficient Reproducible Workflows

- Scripts can document what you did, but not why you did it!
- Scripts separate your analysis products from your report/manuscript

What is literate programming?

- Documentation containing code (not vice versa!)
- Direct connection between code and explanation
- Convey meaning to humans rather than telling computer what to do!

Why literate programming?

- Your analysis scripts **are** computer software
- Integrate math, figures, code, and narrative in one place
- Explaining something helps you learn it

Introducing Quarto

What is Quarto?

- End-to-End process between data and report
- Explicit linkage between each step (including iteration)
- Each step involves trials and choices

What is Quarto?

- A multi-language platform for developing reproducible documents
- A ‘lab notebook’ for your analyses
- Allows transparent, reproducible scientific reports and presentations

Key components

1. Metadata and global options: YAML
2. Text, figures, and tables: Markdown and LaTeX
3. Code: `knitr` (or `jupyter` if you're into that sort of thing)

For this class…

- We'll use headers to outline the analysis
- We'll use code chunks for small, self-contained operations
- We'll create our own functions for repeated operations
- We'll `knit` our documents into a standalone, readable document

Version control, reproducibility, and sanity

Version control in general

- Track changes without version explosion (via `git`)
- Create specific snapshots of a project to facilitate experimentation (via `commit` and `branches`)
- Create centralized backups and ease collaboration (via `GitHub`)

Version control and reproducibility

- Documenting changes to code, manuscripts, figures increases transparency of the scientific process
- Collaboration with other programmers is easier and less risky
- Automates the sharing of code and original data

Version control and sanity

- `commit` early, `commit` often
- use sensible messages to remind yourself where you were
- make sure you always have the most up-to-date version
- **It will take some practice to `git` comfortable**