

Reproducible computational workflow

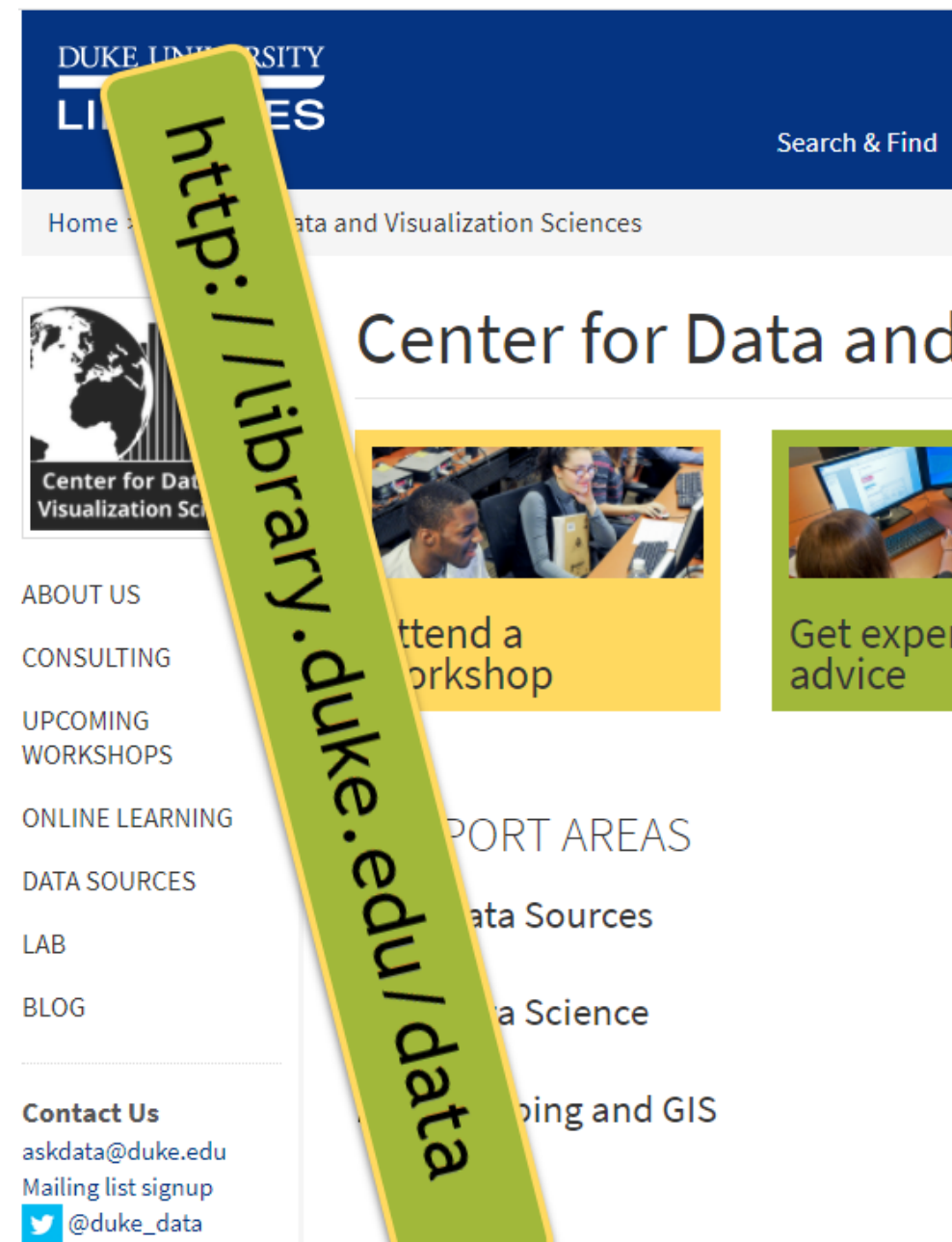
CDVS

John Little

Nov. 5, 2021

Whoami

John Little
Data Science Librarian
Host of Rfun.library.duke.edu
Center for Data & Visualization Sciences



Foundations of reproducible computational research

Data Analysis and Workflow Management

Reproducibility

Reproducibility is about being as lazy as possible

-- Hadley Wickham

So you can recreate your environment as easily as possible

Lazy in a good way

-- Thomas Mock

Outline



- Project Management
- Data Wrangling
- Literate Coding
- Analysis
 - Explanatory/Exploratory
 - Visualizing
 - Modeling
- Report products
- Archiving

Data life cycle

Data \rightsquigarrow Information \rightsquigarrow Publish \rightleftharpoons Archive

Data is *given*. Information is *taken*.

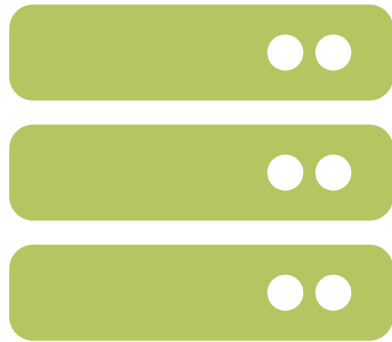
-- Daniel Kaplan. 2011.

- Data: Recorded facts
- Information: a particular form of data well suited to communicate with humans and intended to guide conclusions, beliefs, decision, and action

Finding data

Center for Data & Visualization
Sciences

AskData@Duke.edu



YOUR data

- Raw data sources
- Ancillary data

Citeable, archived, transparent



Data Sharing and Management Snafu in 3 Short Acts



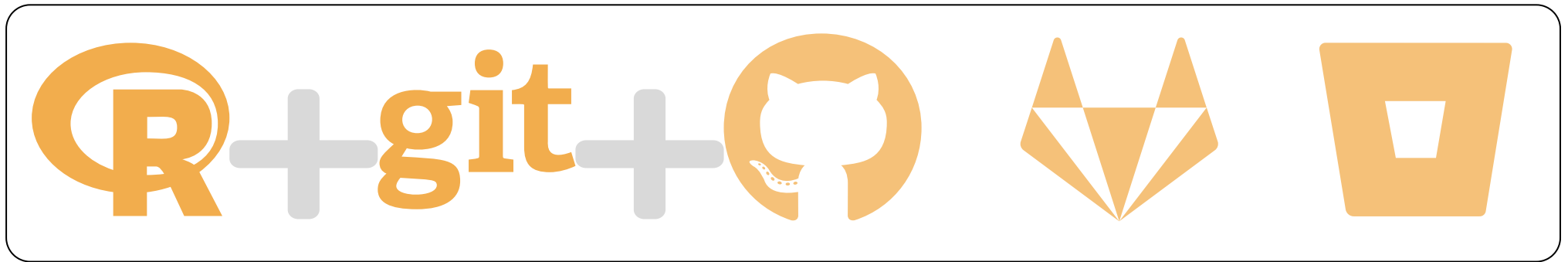
Project Management and file structure

1. Data wrangling
2. Analysis
3. Visualization
4. Reporting
5. Archiving

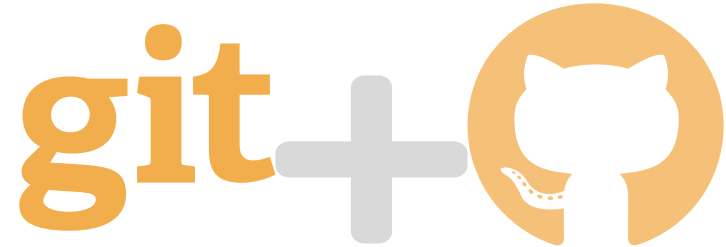


Use source code as the *workflow orchestrator* to manage project elements

Example: RStudio IDE with git version control



Version control and social coding hub



Example project directory structure

ProjectName/

- README.MD
- Data/
 - Raw data/
 - Wrangled and cleaned data/
 - YYYY-MM-DDVersion(s)
 - Use version control ← Better than YYYY-MM-DD
- Code | scripts/
 - Data cleaning/
 - Analysis
 - Models/
- Output and report products/
 - Report products (R Markdown reports, slides, dashboards, MS Word, PDF, LaTeX)
 - Graphs or images/
 - Tables
 - Publications
 - ebooks
 - .bib file

Data Wrangling / Analysis:

- Generate and manage data and analysis with code
- Incorporate found and produced data into the same project
- Data wrangling (normalization & cleaning) as reproducible processes
- Protect personally identifiable information (PRDN)
- Relational database systems bring complexity: great power and administrative responsibility
- ♠♣ Tidy data ♦♥

Some good *reproducible* data wrangling tool options:

OpenRefine • R notebooks • Jupyter notebooks

Tidy data definition

- Every column is a variable
- Every row is an observation
- Every cell is a single value

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	17206362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	216766	128042583

variables

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	17206362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	216766	128042583

observations

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	17206362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	216766	128042583

values

<https://tidyr.tidyverse.org/articles/tidy-data.html#tidy-data> • <https://r4ds.had.co.nz/tidy-data.html#tidy-data-1>

Example Untidy data

religion	<\$10k	\$10-20k	\$20-30k	\$30-40k	\$40-50k	\$50-75k	\$75-100k	\$100-150k	>150k	Don't know/refused
Agnostic	27	34	60	81	76	137	122	109	84	96
Atheist	12	27	37	52	35	70	73	59	74	76
Buddhist	27	21	30	34	33	58	62	39	53	54
Catholic	418	617	732	670	638	1116	949	792	633	1489
Don't know/refused	15	14	15	11	10	35	21	17	18	116
Evangelical Prot	575	869	1064	982	881	1486	949	723	414	1529

Problem?

Column headers are values, not variable names

Example Tidy data

Religious income		
religion	income	value
Agnostic	\$10-20k	34
Atheist	\$10-20k	27
Buddhist	\$10-20k	21
Catholic	\$10-20k	617
Don't know/refused	\$10-20k	14
Evangelical Prot	\$10-20k	869
tidyr::relig_incom Downloaded from https://www.pewforum.org/religious-landscape-study/ (downloaded November 2009)		

Literate coding

Combine code with prose and visualizations

- Use prose to explain analysis
 - Structure your analysis and documentation
 - Markdown | R Markdown
- Analysis AND report writing
 - Data code-books are part of the project
 - Render reports from code (*report products*)

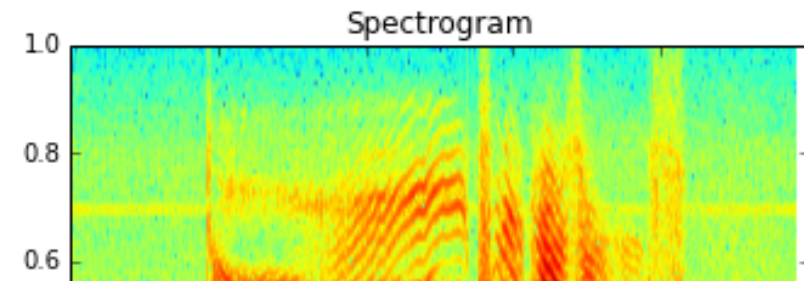
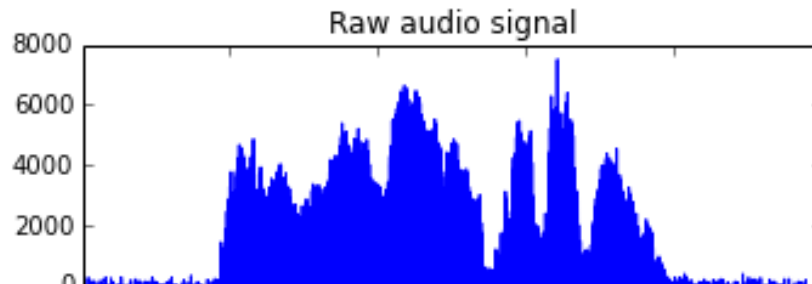
$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \quad k = 0, \dots, N-1$$

We begin by loading a datafile using SciPy's audio file support:

```
In [1]: from scipy.io import wavfile
rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib's builtin specgram routine:

```
In [2]: %matplotlib inline
from matplotlib import pyplot as plt
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
ax1.plot(x); ax1.set_title('Raw audio signal')
ax2.specgram(x); ax2.set_title('Spectrogram');
```

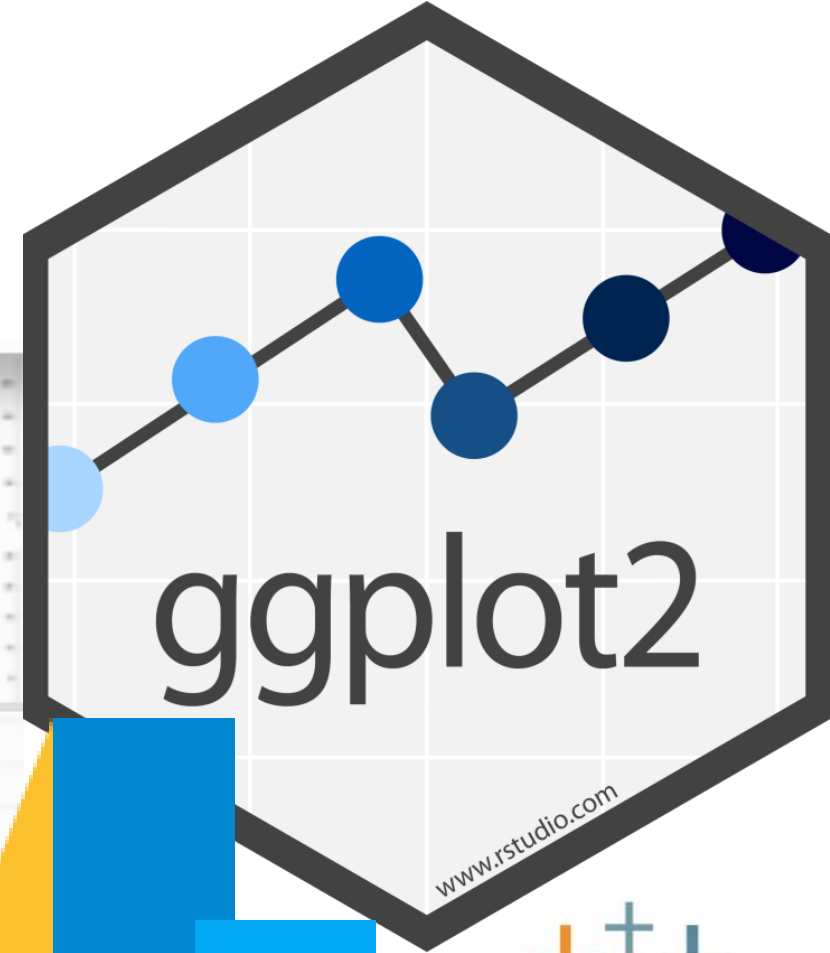
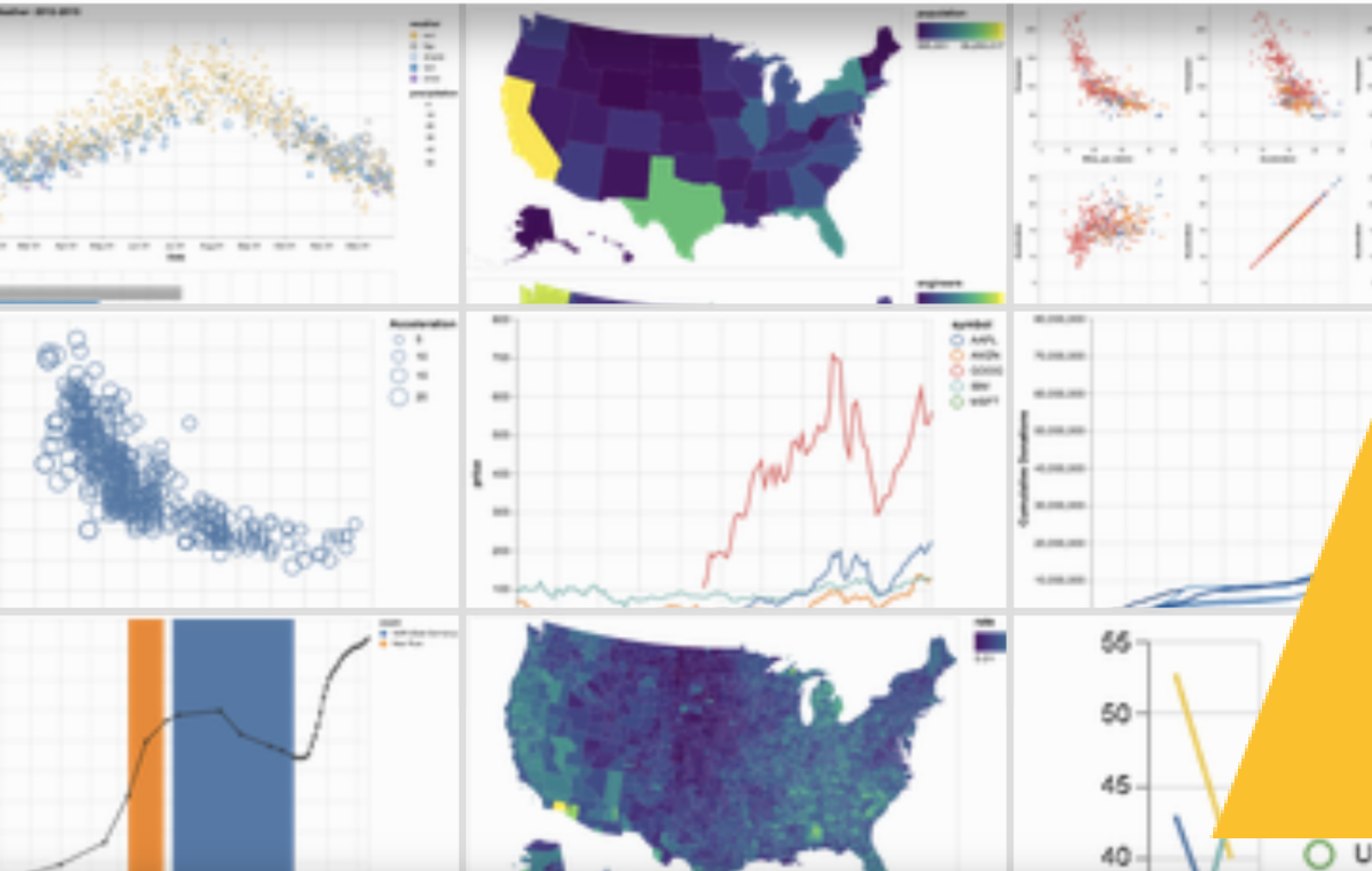


↑ Example jupyter notebook ↑

<https://arogozhnikov.github.io/2016/09/10/jupyter-features.html>

Visualize & spatial analysis

Use code-based tools or tools with reproducible features



Report products

Leverage your literate coding

- Use rendered notebooks to *show your work* without requiring the reader to reproduce the exact compute environment
 - Document all versions and session information
- *Render (derived) reports* from literate code documents
 - Generate slides, dashboards, documents, visualizations, books, PDFs, LaTeX, etc. from the same source code

Static and interactive documents work well with R Markdown
literate coding

Archiving

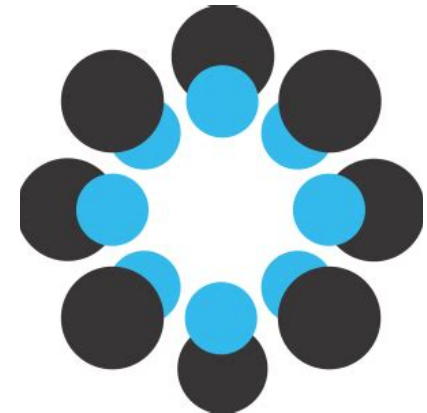
- *Version Control* = (Git) + Social Coding
 - Social Coding Hubs = collaboration + self-documenting your project's story within a **code repository** GitHub • GitLab • BitBucket
- **Archival repositories**
 - disciplinary v institutional
 - data repositories
 - DOIs for output: Article, Code, etc. (i.e. Publishing)
 - link to your ORCID
- *Containers*: computational and archival representations of your project at various publishable milestones
 - Zero-install environment: does not require a *reader* to set up a replica compute environment

OSF

Open Science Framework

OSF is a free and open source project management tool that supports researchers throughout their entire project lifecycle.

<https://OSF.io>



Related topics


DOIs

DOI [10.5281/zenodo.4908855](https://doi.org/10.5281/zenodo.4908855) (Latest Version Release)

Containers

[launch](#) [binder](#)

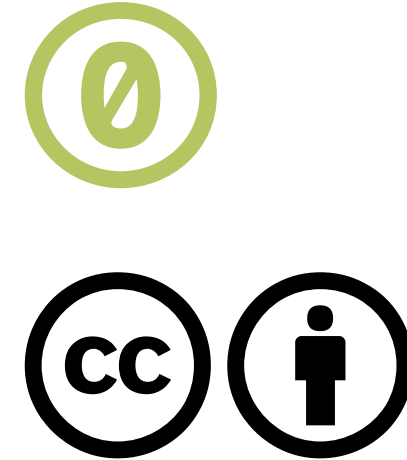
Author ID

 ORCID [0000-0002-3600-0972](https://orcid.org/0000-0002-3600-0972)



Licensing

- MIT for Software
- CC-BY for documents
- CC0 for data



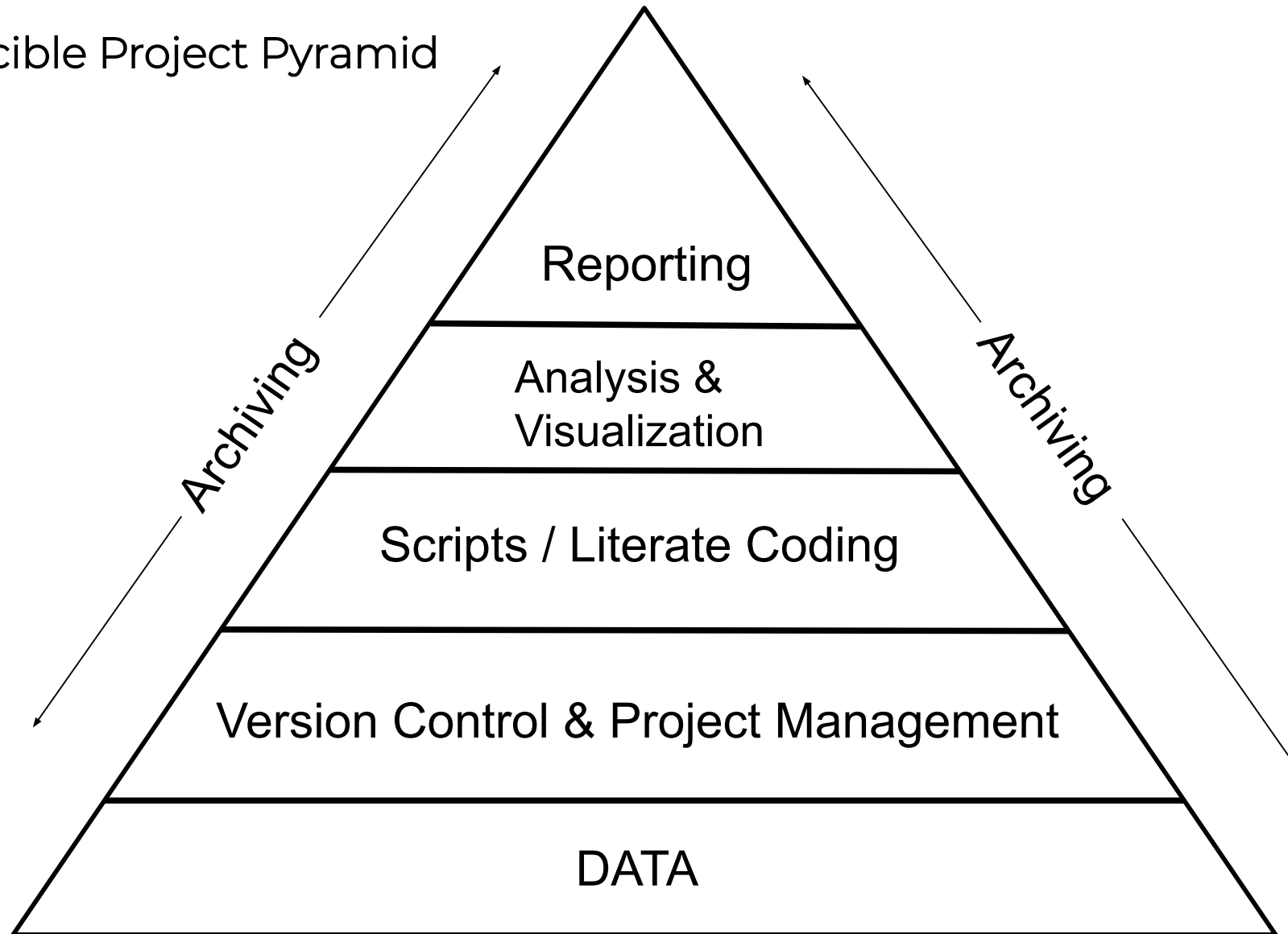
Information about licensing

<https://exygy.com/blog/which-license-should-i-use-mit-vs-apache-vs-gpl/>
<https://arstechnica.com/gadgets/2020/02/how-to-choose-an-open-source-license/>
<https://creativecommons.org/>

Office of Copyright and Scholarly Communications

<https://library.duke.edu/about/depts/scholcomm>

Reproducible Project Pyramid

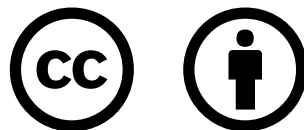




John R Little

Data Science Librarian
Center for Data & Visualization Sciences
Duke University Libraries

<https://johnlittle.info>
<https://Rfun.library.duke.edu>
<https://library.duke.edu/data>



Creative Commons: Attribution 4.0

<https://creativecommons.org/licenses/by-nc/4.0>