

BIOS 259: The Art of Reproducible Science

A Hands-on Approach | A **Stanford** Biosciences Mini-course

Day 06: Documenting and sharing code/data

Aziz Khan <azizk@stanford.edu>

<https://github.com/asntech/bios259-w24/>



Course schedule

Date	Topic	Time	Location
02.26.2024 – Monday	Introduction to reproducibility and setting up	10:00-13:00	Alway Building M218A
02.28.2024 – Wednesday	Version Control (Git/GitHub)	10:00-13:00	M218A
03.01.2024 – Friday	Dependency management (Conda, Mamba, Bioconda)	10:00-13:00	M218A
03.04.2024 – Monday	Containerization (Docker, Singularity)	10:00-13:00	M218A
03.06.2024 – Wednesday	Workflows (Snakemake, Nextflow/nf-core)	10:00-13:00	M218A
03.08.2024 – Friday	Documentation (FAIR data and open code) and wrap up	10:00-13:00	Li Ka Shing LK208

M218A: <https://25live.collegenet.com/pro/stanfordson#~/home/location/1454/details>

LK208: <https://25live.collegenet.com/pro/stanfordson#~/home/location/1149/details>

Learning goals for today

Learn best practices and tools for organizing, sharing, and documenting code and data to facilitate reproducibility and enable others to build upon your work.

- Project organization - directory structure and file naming
- Literate programming using Jupyter notebook, R Markdown, Quarto
- FAIR data principles and long term archiving
- Open and versioned software and long term archiving

Think-pair share exercise

Think: Take a few minutes to reflect on your current project organization methods

- Consider the directory structure you use and how you name your files. What challenges have you faced in organizing your projects effectively?

Pair: Pair up with a partner and discuss your thoughts on project organization

- Share your current practices, any strategies you've found helpful, and any difficulties you've encountered. Compare and contrast your approaches.

Share: Share with the larger group some key insights from your discussion

- Discuss common challenges and solutions related to project organization. Share any new ideas or best practices you've learned from your partner.

Project organization

Plan early about the file names and folder structure

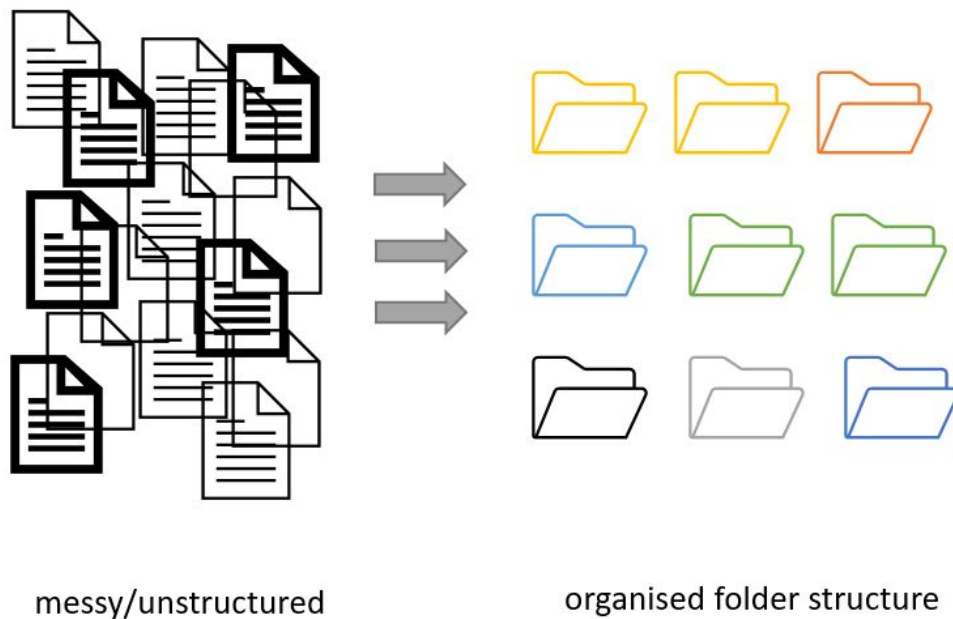


Figure credits: Andrés Romanowski

Reproducibility starts from the day one

Computational **reproducibility starts with** when the **project starts**:

```
> cd projects  
> mkdir <project_name>
```

Not when your PI told you to put the code in a repository **because the Journal and/or the funding agency mandates it.**

Organizing your projects

- A good starting point is to keep all files associated with a project in a **single folder**
- **Different projects** should have **separate folders**
- Use **consistent and informative directory structure**
- Add a **README file** to describe the project and instructions on reproducing the results
- Use **.gitignore**
- Your mileage may vary: it's **not a one-size-fits-all**

```
project_name/  
├── README.md  
├── data/  
│   ├── README.md  
│   └── sub-folder/  
├── processed_data/  
├── manuscript/  
├── results/  
├── src/  
│   ├── LICENSE  
│   ├── requirements.txt  
│   └── ...  
└── doc/  
    ├── index.rst  
    └── ...
```



Boilerplate for reproducible science

Project Structure

```
.
├── AUTHORS.md
├── LICENSE
├── README.md
├── bin
│   └── <- Your compiled model code can be stored here (not tracked by git)
├── config
│   └── <- Configuration files, e.g., for doxygen or for your model if needed
├── data
│   ├── external
│   │   └── <- Data from third party sources.
│   ├── interim
│   │   └── <- Intermediate data that has been transformed.
│   ├── processed
│   │   └── <- The final, canonical data sets for modeling.
│   └── raw
│       └── <- The original, immutable data dump.
├── docs
│   └── <- Documentation, e.g., doxygen or scientific papers (not tracked by git)
├── notebooks
│   └── <- IPython or R notebooks
├── reports
│   └── <- For a manuscript source, e.g., LaTeX, Markdown, etc., or any project report
├── figures
│   └── <- Figures for the manuscript or reports
└── src
    ├── data
    │   └── <- scripts and programs to process data
    ├── external
    │   └── <- Any external source code, e.g., pull other git projects, or external libraries
    ├── models
    │   └── <- Source code for your own model
    ├── tools
    │   └── <- Any helper scripts go here
    └── visualization
        └── <- Scripts for visualisation of your results, e.g., matplotlib, ggplot2 related
```

Cookiecutter is a cross-platform command-line utility that creates projects from project templates, e.g. Python package projects

Install and get the template for reproducible science

> pip install cookiecutter

> cookiecutter

gh:mkrappp/cookiecutter-reproducible-science

<https://github.com/mkrapp/cookiecutter-reproducible-science>

What are your file names look like?

✗ Bad	✓ Good
Myabstract.docx	2020-06-08_abstract-for-sla.docx
Joe's Filenames Use Spaces and Punctuation.xlsx	Joes-filenames-are-getting-better.xlsx
figure 1.png	Fig01_scatterplot-talk-length-vs-interest.png
fig 2.png	Fig02_histogram-talk-attendance.png
JW7d^(2sl@deletethisandyourcareerisoverWx2*.txt	1986-01-28_raw-data-from-challenger-o-rings.txt

<https://the-turing-way.netlify.app/project-design/filenaming>

File names matter

File naming might seem mundane, but **it plays a crucial role in organizing, finding, and sharing research** output effectively

Make your file names readable and orderable

Your file names should be **machine readable**, **human readable** and easily **orderable**.

- No spaces, special characters
- Write descriptive names in a logical order (separated by “_” or “-”)
- Put the date at the start - YYYYMMDD

Use YYYY-MM-DD

Use the ISO 8601 standard for dates YYYY-MM-DD

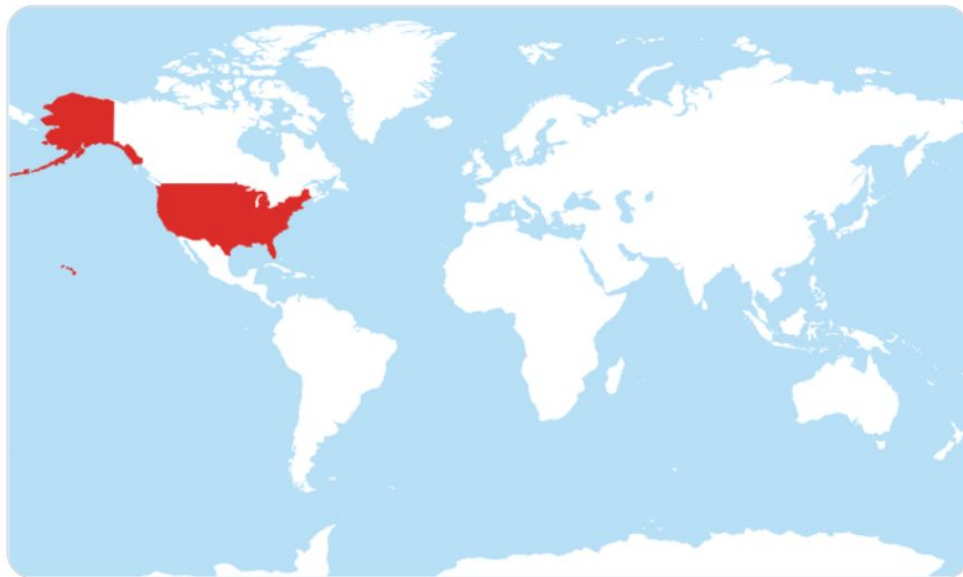


Michael Donohoe

@donohoe

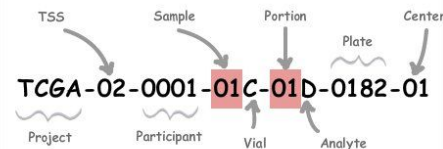


Comprehensive map of all countries in the world that use the MMDDYYYY format



Create informative sample naming

TCGA barcode is a good example

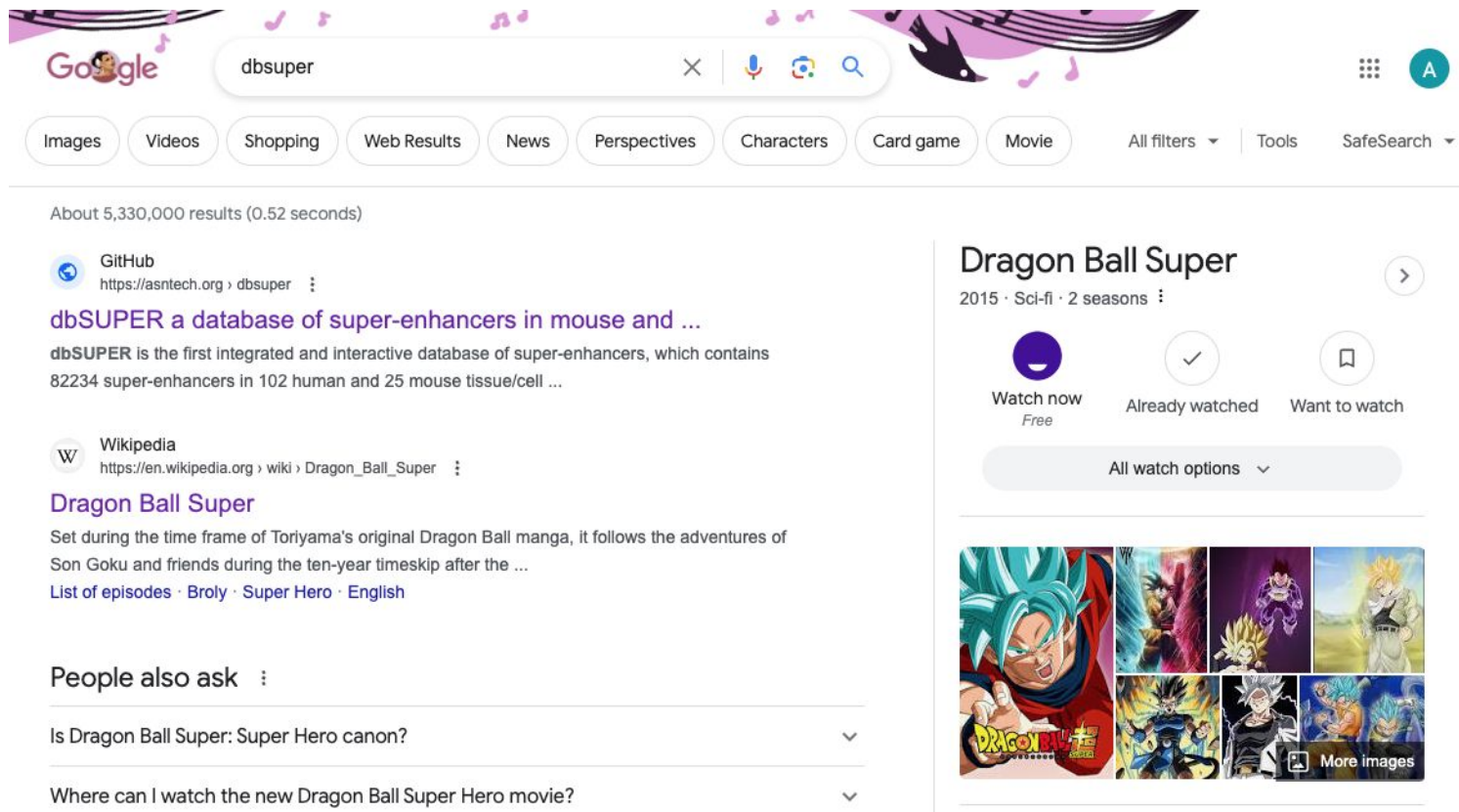


Label	Identifier for	Value	Value Description	Possible Values
Analyte	Molecular type of analyte for analysis	D	The analyte is a DNA sample	See Code Tables Report
Plate	Order of plate in a sequence of 96-well plates	182	The 182nd plate	4-digit alphanumeric value
Portion	Order of portion in a sequence of 100 - 120 mg sample portions	1	The first portion of the sample	01-99
Vial	Order of sample in a sequence of samples	C	The third vial	A to Z
Project	Project name	TCGA	TCGA project	TCGA
Sample	Sample type	1	A solid tumor	Tumor types range from 01 - 09, normal types from 10 - 19 and control samples from 20 - 29. See Code Tables Report for a complete list of sample codes
Center	Sequencing or characterization center that will receive the aliquot for analysis	1	The Broad Institute GCC	See Code Tables Report
Participant	Study participant	1	The first participant from MD Anderson for GBM study	Any alpha-numeric value
TSS	Tissue source site	2	GBM (brain tumor) sample from MD Anderson	See Code Tables Report

Name your software/tool carefully

- Choose a unique, descriptive, and memorable name for your software tool
- Ensure the name accurately reflects the functionality and purpose of your tool
- Search and check if your chosen name is not taken by someone else for thingelse
- Check for trademark availability and potential conflicts before finalizing your software name

dbSUPER - a database of super-enhancers



The image is a screenshot of a Google search page. The search bar at the top contains the text 'dbsuper'. Below the search bar, there are tabs for 'Images', 'Videos', 'Shopping', 'Web Results', 'News', 'Perspectives', 'Characters', 'Card game', and 'Movie'. The 'Web Results' tab is selected. The search results show 'About 5,330,000 results (0.52 seconds)'. The first result is from GitHub, titled 'dbSUPER a database of super-enhancers in mouse and ...', with a description: 'dbSUPER is the first integrated and interactive database of super-enhancers, which contains 82234 super-enhancers in 102 human and 25 mouse tissue/cell ...'. The second result is from Wikipedia, titled 'Dragon Ball Super', with a description: 'Set during the time frame of Toriyama's original Dragon Ball manga, it follows the adventures of Son Goku and friends during the ten-year timeskip after the ...'. Below the search results, there is a section 'People also ask' with two questions: 'Is Dragon Ball Super: Super Hero canon?' and 'Where can I watch the new Dragon Ball Super Hero movie?'. On the right side of the page, there is a sidebar for 'Dragon Ball Super', showing the title, year (2015), genre (Sci-fi), and number of seasons (2). It also has buttons for 'Watch now Free', 'Already watched', and 'Want to watch'. Below these buttons is a button for 'All watch options'. At the bottom of the sidebar is a grid of images from the series, with a 'More images' link.

Google

dbsuper

Images Videos Shopping Web Results News Perspectives Characters Card game Movie

All filters Tools SafeSearch

About 5,330,000 results (0.52 seconds)

GitHub
https://asntech.org › dbsuper

dbSUPER a database of super-enhancers in mouse and ...

dbSUPER is the first integrated and interactive database of super-enhancers, which contains 82234 super-enhancers in 102 human and 25 mouse tissue/cell ...

Wikipedia
https://en.wikipedia.org › wiki › Dragon_Ball_Super

Dragon Ball Super

Set during the time frame of Toriyama's original Dragon Ball manga, it follows the adventures of Son Goku and friends during the ten-year timeskip after the ...

[List of episodes](#) · [Broly](#) · [Super Hero](#) · [English](#)

People also ask

Is Dragon Ball Super: Super Hero canon?

Where can I watch the new Dragon Ball Super Hero movie?

Dragon Ball Super

2015 · Sci-fi · 2 seasons

Watch now Free

Already watched

Want to watch

All watch options

More images

Best practices for naming

- Use **descriptive and concise** file names that reflect the content and purpose of the file.
- Be **consistent** in file naming conventions across your research project.
- **Avoid special characters**, spaces, and ambiguous abbreviations in file names.
- Document file naming conventions in project documentation for clarity and reproducibility.

Naming exercise

Objective: The objective of this exercise is to emphasize the importance of naming conventions in biological research and to provide hands-on experience in assigning names to samples with different types of data.

Scenario: You are part of a big project that aims to integrate multi-omics data generated by different centers, including genomics (**WG, WE**), whole transcriptomics (**WT**), epigenomics (DNA Meth), and spatial imaging data, to develop personalized treatment strategies for patients with a complex disease. The patients have multiple matched tumor/normal samples both pre and post treatment. As initial step of new project, your team aims to standardize the naming convention.

Divide in two groups and develop a unique naming convention to ensure clarity and consistency throughout the project:

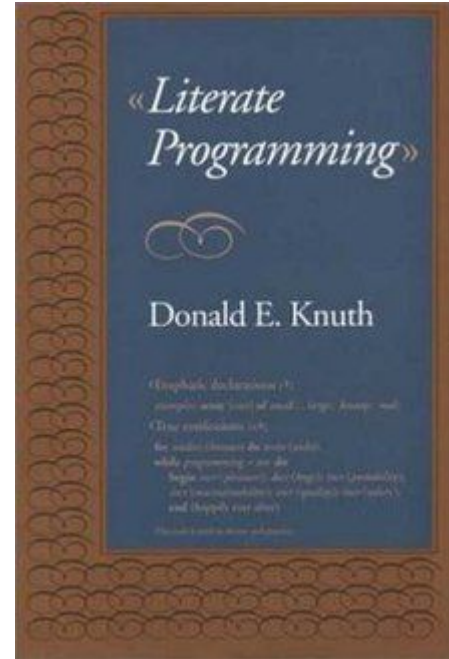
Patient_ID, Sample_ID, Experiment_ID

Literate programming

Use Notebooks - Jupyter, R Markdown,

What is literate programming?

- Literate programming is writing code in a way that is **understandable to humans**, as well as **executable by computers**. In literate programming, the source code is written as an explanation of the program's logic, embedded with documentation, rather than just a series of instructions.
- The concept was introduced by Donald Knuth in 1984
- It is used in scientific computing and in data science routinely for reproducible research



The main components of literate programming

Descriptive text: This is the explanatory text written in natural language, which describes the problem being solved, the algorithm used, and the reasoning behind the code.

Code snippets: These are sections of code written in a programming language (e.g., Python, R, Julia) that perform specific tasks within the program. Code chunks are embedded within the narrative text.

Documentation: Comprehensive documentation is a crucial aspect of literate programming. It includes explanations of code snippets, algorithmic details, and any other relevant information needed to understand the program.

Benefits of literate programming

- Literate programming helps to write code that is **more readable, maintainable, and understandable** by others.
- It promotes clear communication of ideas and facilitates **collaboration** and ensure **reproducibility**.

Tools for literate programming



jupyter.org



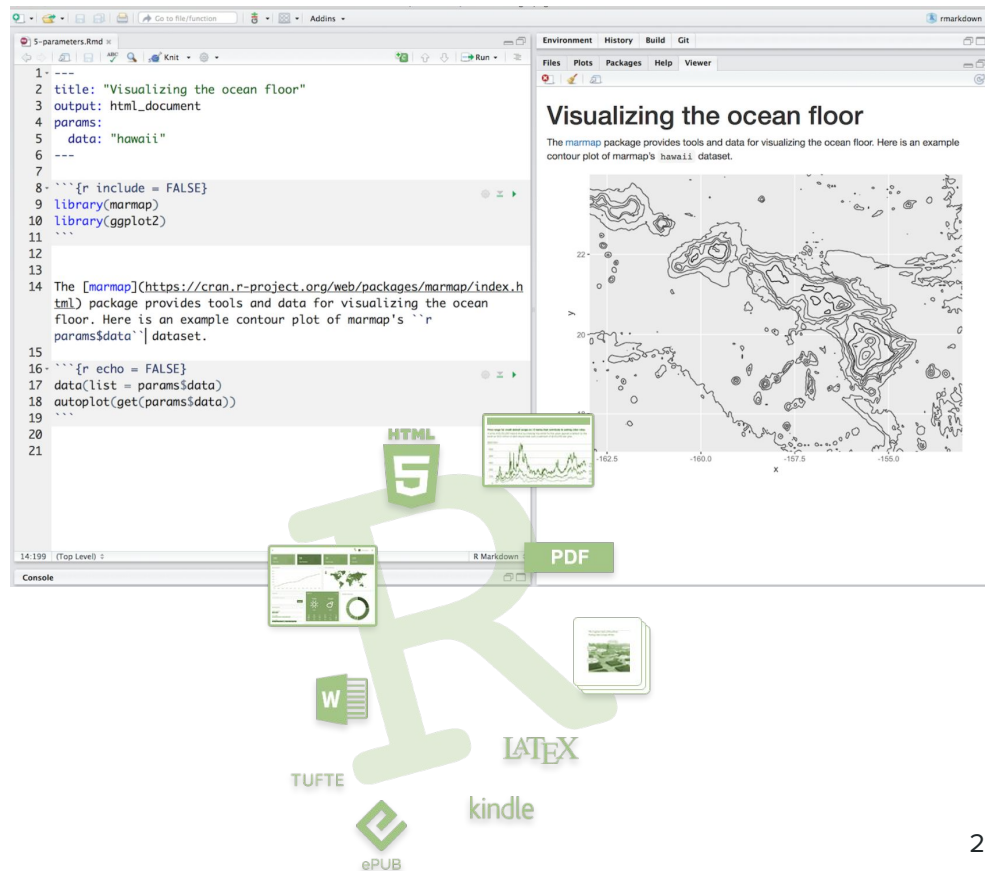
rmarkdown.rstudio.com



quarto.org

Literate programming using **R Markdown**

- R Markdown documents are fully reproducible
- Use notebooks to put together narrative text and code to produce figures
- You can use multiple languages including R, Python, and SQL
- Export the outputs in different readable formats

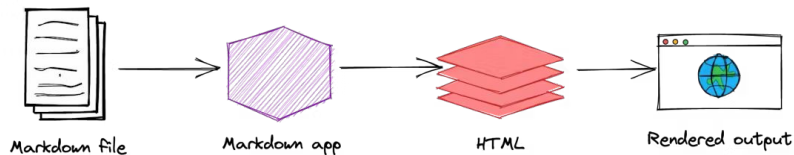


Markdown

Markdown is a **lightweight markup language** that you can use to add formatting elements to plaintext text documents.

Created by John Gruber in 2004

Markdown is now **one of the world's most popular** markup languages.



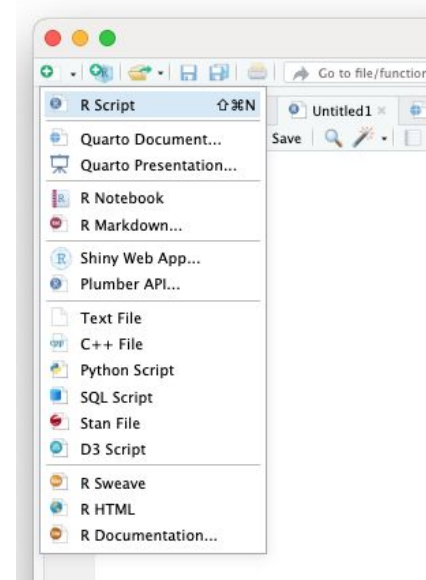
<https://www.markdownguide.org/getting-started/>
<https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf>

Element	Markdown Syntax
Heading	# H1 ## H2 ### H3
Bold	**bold text**
Italic	<i>*italicized text*</i>
Blockquote	> blockquote
Ordered List	1. First item 2. Second item 3. Third item
Unordered List	- First item - Second item - Third item
Code	`code`
Horizontal Rule	---
Link	[title](https://www.example.com)
Image	![alt text](image.jpg)

R Notebook vs. R Markdown

R Markdown makes dynamic documents that incorporate R code and text. It is a lightweight markup language.

R Notebook is an **interactive** notebook interface that allows you to compile R code, visualize output, and form documents in real-time.



Some R/Markdown tips

- Do not hard code absolute paths and metadata in your code
- Keep all code and data in one directory
 - Or have a variable at the top `project_name_path="/path/to/project_name"`
- For anything with randomness, ***set.seed()***
- Use ***renv*** to lock the package versions
- Include a final chunk that use `devtools::session_info()` to session information about your system to help others reproduce

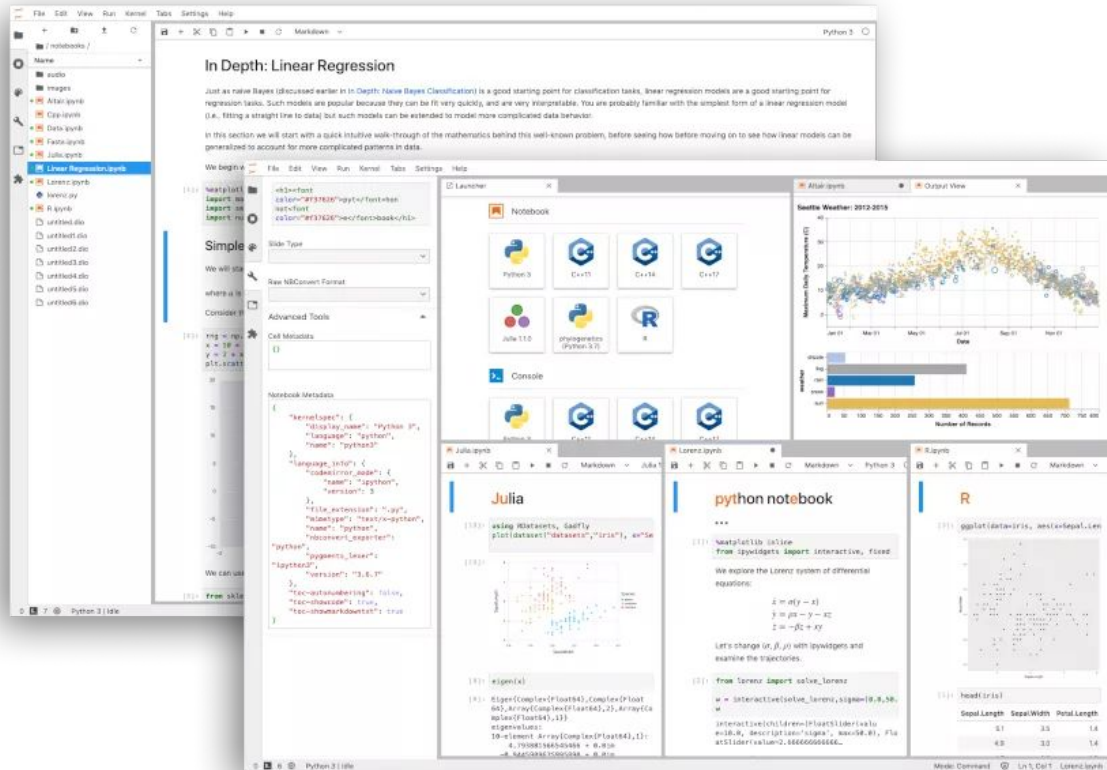
Literate programming using Jupyter Notebook

Jupyter Notebooks are **popular** in data science and scientific computing

Allow users to **combine narrative text, code, and visualizations** in a single document

Can be export the document to portable outputs HTML, PDF, etc

<https://jupyter.org>

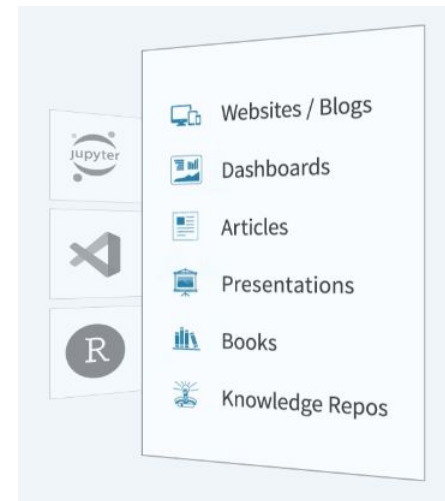


Start a Jupyter Notebook

- Go to your project directory on terminal and run:
 - `$ jupyter notebook`
 - This will open the browser with address <http://localhost:8888/>
-
- The output will be saved as **.ipynb**
 - These are **JSON** (Java Script Object Notation) a plain text and that can be read/created in any programming language

Literate programming using Quarto

- Quarto is a modern, open source and multi-language tool for literate programming by *Posit* – same company behind RStudio
- Can use in Rstudio, VS code and Jupyter
- Create dynamic content with Python, R, Julia, and JavaScript
- Publish reproducible, production quality outputs in HTML, PDF, MS Word, ePub, and more
- Provide advanced layouts and features such as citations, crossrefs, figure panels, and more



<https://quarto.org/>

Quarto R-interface: <https://quarto-dev.github.io/quarto-r/>

Write better code

Write code that **works**, **readable**, and **reproducible**.

Only comment code that's hard to understand.

Try not to write code that's hard to understand.

Hands-on exercise

Notebooks - 30 mins

<https://github.com/asntech/bios259-w24/tree/main/06-docs-sharing>

Make your code open and data FAIR-er

FAIR data and long-term archiving

- Publishing and sharing data is good for science
- But making it **open, reproducible** and **FAIR** is best for science and society



Juan Nunez-Iglesias

@jnuneziglesias



sigh why is this still a thing in 2021?

Data availability

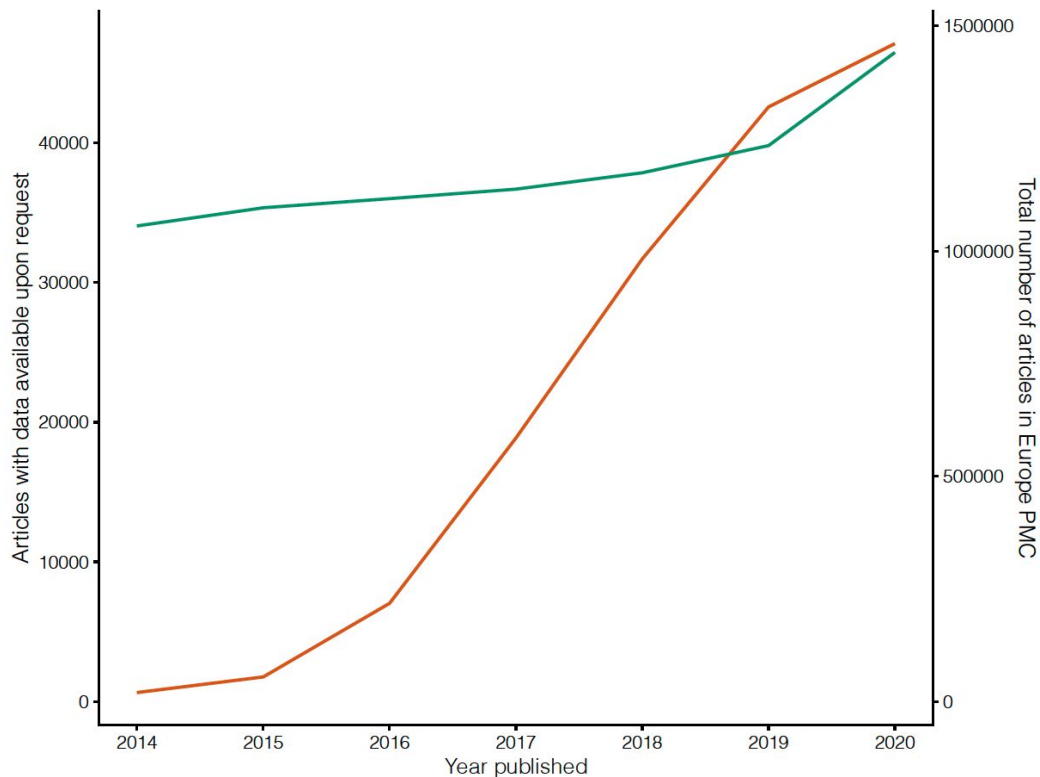
All data that support the findings of this study are available from the corresponding author upon reasonable request. [Source data](#) are provided with this paper.

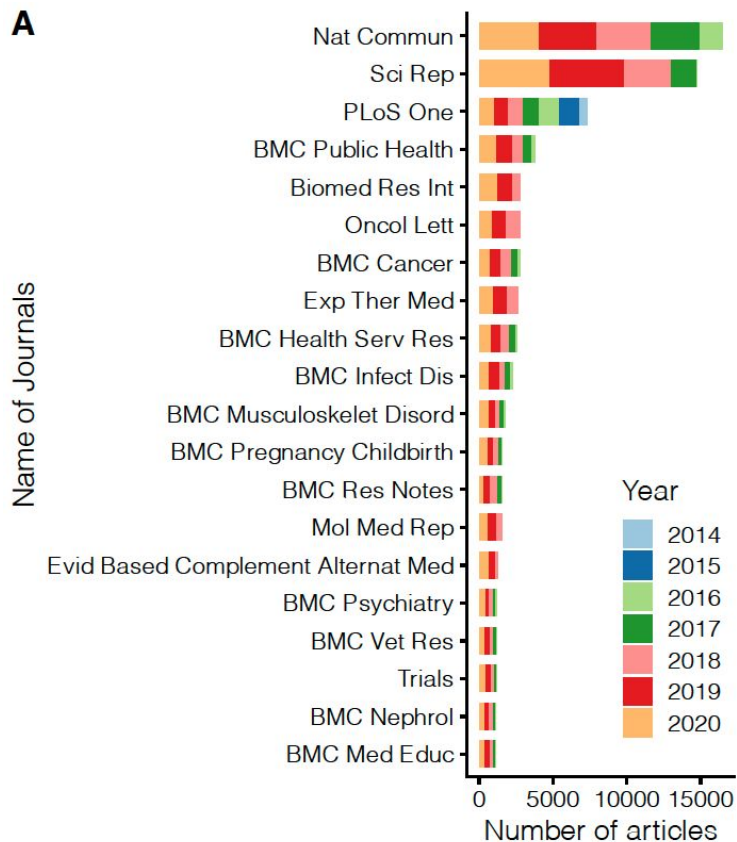
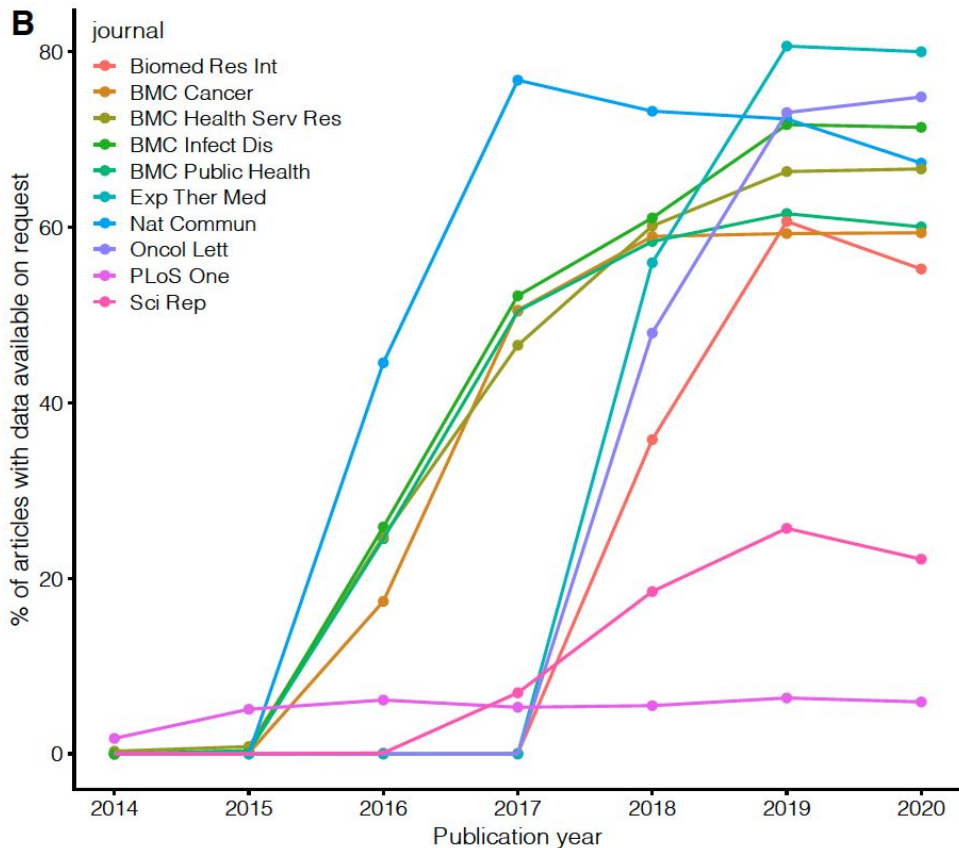
Code availability

The code used to analyze the data is available from the corresponding author upon reasonable request.

Data available upon request is increasing

The growth of biomedical research articles with associated data available upon request



A**B**

14% responded and only ~7% shared the requested data

ORIGINAL ARTICLE | VOLUME 150, P33-41, OCTOBER 2022



Many researchers were not compliant with their published data sharing statement: a mixed-methods study

Mirko Gabelica • Ružica Bojčić • Livia Puljak  

Published: May 29, 2022 • DOI: <https://doi.org/10.1016/j.jclinepi.2022.05.019> •



Data availability

Metadata and cellranger outputs are available at Zenodo (<https://doi.org/10.5281/zenodo.6401895>). Expressed cellular barcodes (ECB) sequencing data are available at bioProject ID ([PRJNA838456](https://www.ncbi.nlm.nih.gov/bioproject/PRJNA838456)). Genomic sequencing and scRNA-seq data are available at dbGAP under accession no. phs003249.v1. [Source data](#) are provided with this paper.

Code availability

The computational methods, procedures and analyses summarized above are implemented in custom R and python, and bash scripts are available via the Curtis Lab: https://github.com/cancersysbio/gastric_organoid_evolution.

Practical challenges for researchers in data sharing

Springer Nature have published the results of a survey of >7,700 researchers worldwide, looking at data sharing during publication



Main challenge to data sharing is **organising data in a presentable and useful way**

Almost half of all respondents (46%) said that **organising data** was a challenge, followed by **confusion around copyright** (37%) and **not knowing where to share data** (33%)

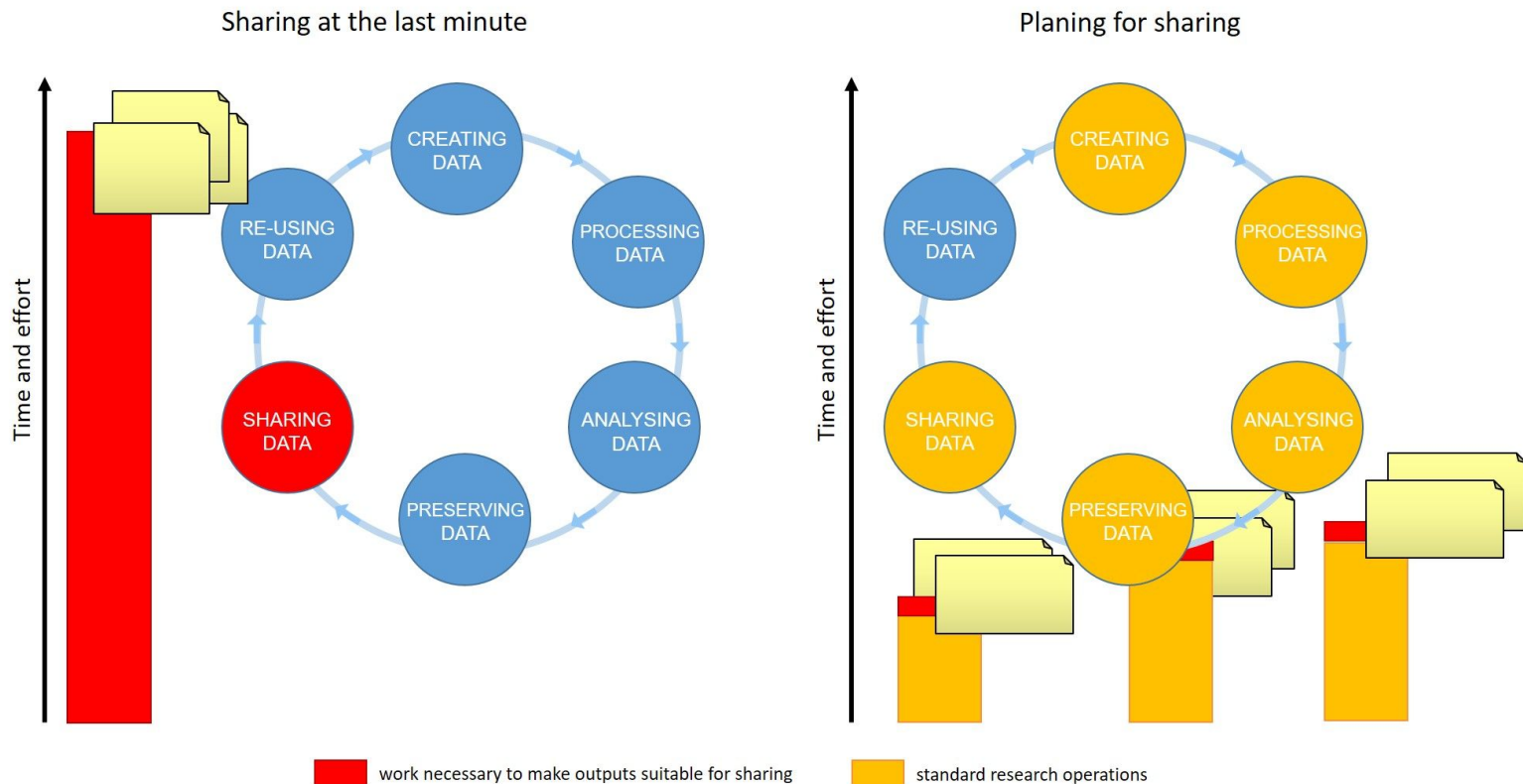


Small datasets are the **least likely to be shared**

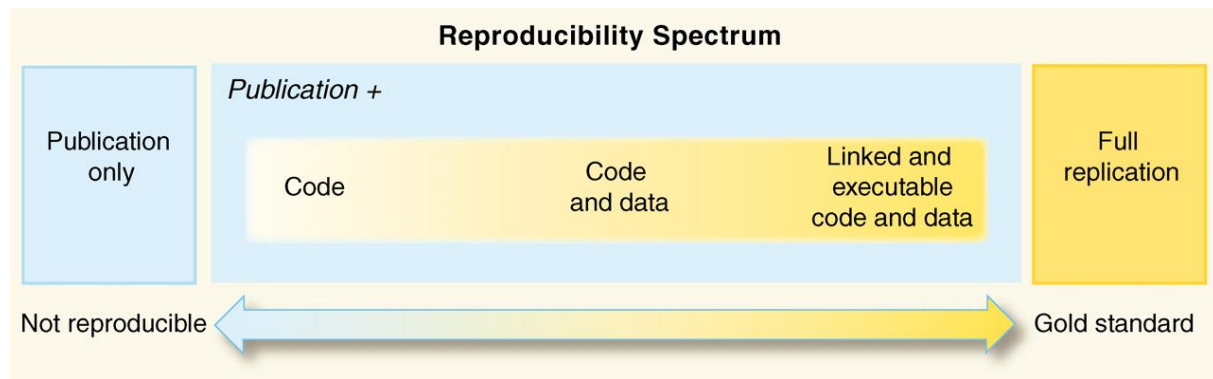
Researchers that generated the smallest sized data files (<20MB; n = 2,036) had the highest **proportion of data** that were neither shared as supplementary information nor deposited in a repository (**42%**)

<https://doi.org/10.6084/m9.figshare.5975011>

Data management is a continuous process



FAIR data and code sharing are vital for reproducibility



Poger Peng (2021) - doi: 10.1126/science.1213847



Source - <https://www.nlm.nih.gov/oet/ed/cde/tutorial/02-200.html>

2016 paper - <https://www.nature.com/articles/sdata201618>
<https://www.go-fair.org/fair-principles/>

Findable

For data to be findable there must be **sufficient metadata**; there must be a **unique and persistent identifier**; and the data must be registered or indexed in a searchable resource.

- The data has a DOI

Accessible

To be accessible, metadata and data should be **readable** by **humans** and by **machines**, and it must reside in a trusted repository.

- Data can be downloaded and read by human and also computers

Interoperable

Data must share a **common structure**, and metadata must use recognized, formal terminologies for description.

- Always use .csv or .tsv files for numerical data. Never share data tables as word or pdf
- Where possible make data available through REST APIs

Reusable

Data and collections must have **clear usage licenses** and clear provenance, and meet relevant community standards for the domain.

- Add license to mention conditions and terms by which data and software can be reused
- Add README file describing the data
- Use descriptive column headers for the data tables

Data repositories

There are general “data agnostic” repositories, for example:

- Dryad
- Zenodo
- FigShare
- Dataverse

Or domain specific, for example:

- UniProt protein data,
- GenBank sequence data,
- MetaboLights metabolomics data
- GitHub for code.

FAIR data exercise (5 min)

Have a look at the dataset record with COVID-19 data:

<https://doi.org/10.5281/zenodo.6339631>

Identify how each of **FAIR** (**F**indable, **A**ccessible, **I**nteroperable, **R**eusable) principles has been met

Hint: Check the linked github repo to easily read the README file

Hands-on exercise - long-term archiving using Zenodo

1. Open - <https://zenodo.org/>
2. Login or sign up with GitHub
3. Go to settings and click GitHub or use the link
<https://zenodo.org/account/settings/github/>
4. Click “Sync Now”
5. Click the OFF button and refresh the page
6. Now you will be able to see the repo under Enabled Repositories
7. But this will not let you publish - you need to create a release on GitHub first
8. Let's go to GitHub to check/add LICENSE.md file and create our first release of project code
9. Get back to Zenodo and publish the repo and get the DOI

Takeaways from the course

1. Reproducibility is **good for you first**, then to science and society – **day 1**
2. Use *Git* **version control** for code and GitHub to collaborate – **day 2**
3. Record/freeze **software versions** and dependencies – **day 3**
4. **Containerize** compute environments (Docker, Singularity) – **day 4**
5. **Automate** your analysis pipelines (e.g. Nextflow, Snakemake,..) – **day 5**
6. Use *Jupyter/R/Quarto* **Notebooks** for literate programming – **day 6**
7. Share code through **persistent repositories** and **make data FAIR** – **day 6**

Resources

- https://rstudio-pubs-static.s3.amazonaws.com/180546_e2d5bf84795745ebb5cd3be3dab71fca.html
- <https://carpentries-incubator.github.io/fair-bio-practice/index.html>