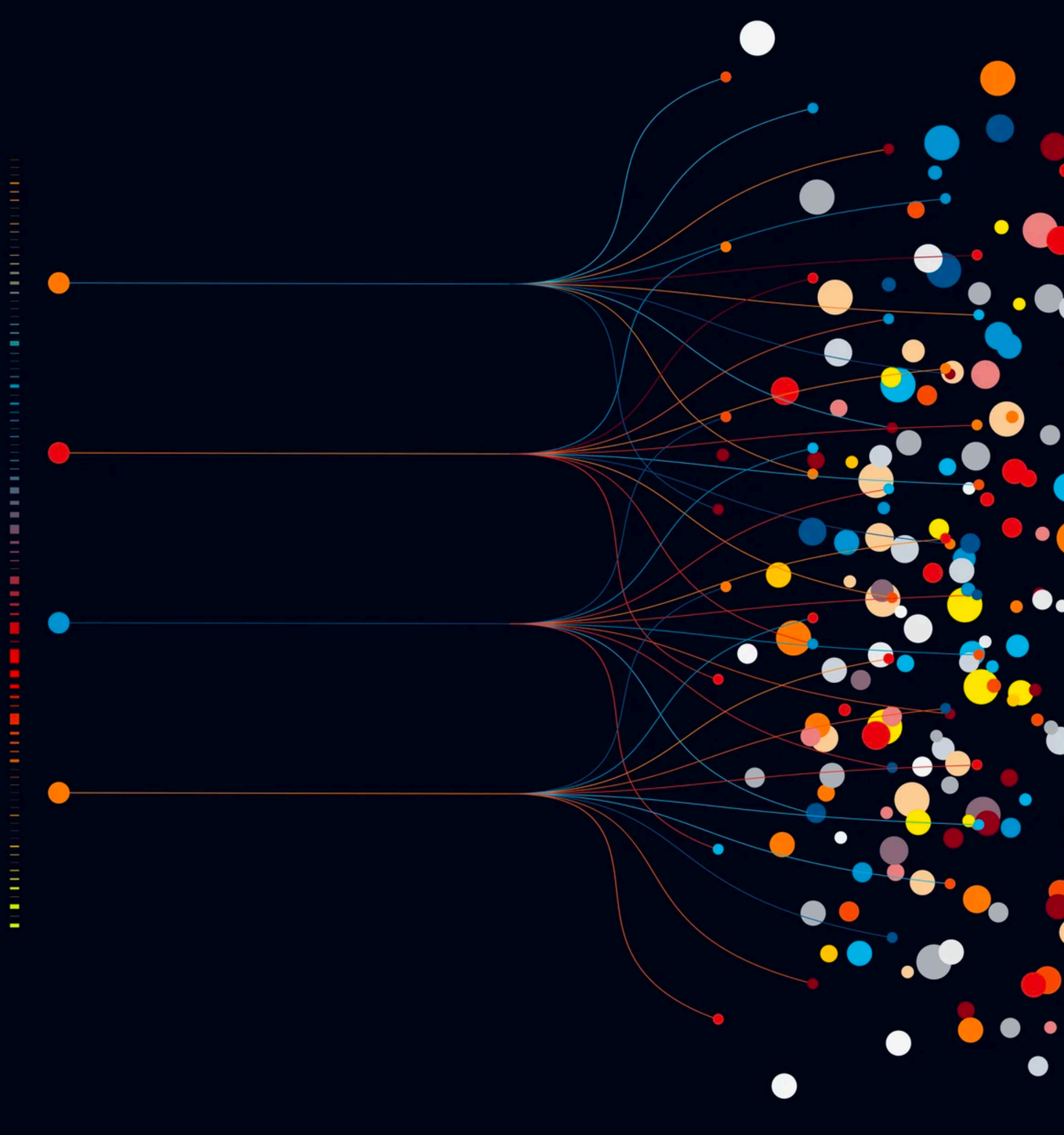
Module 4 - day 1

Reproducible data analysis and workflows

Antonio Fernandez-Guerra
antonio@metagenomics.eu

Fundamentals Data Analysis | March 6 2025 - Copenhagen



Objectives

1. **The Reproducibility Crisis in Science**
   Understanding the Problem and Finding Solutions

2. **Ensuring Reproducibility in Scientific Research**
   Best Practices and Strategies

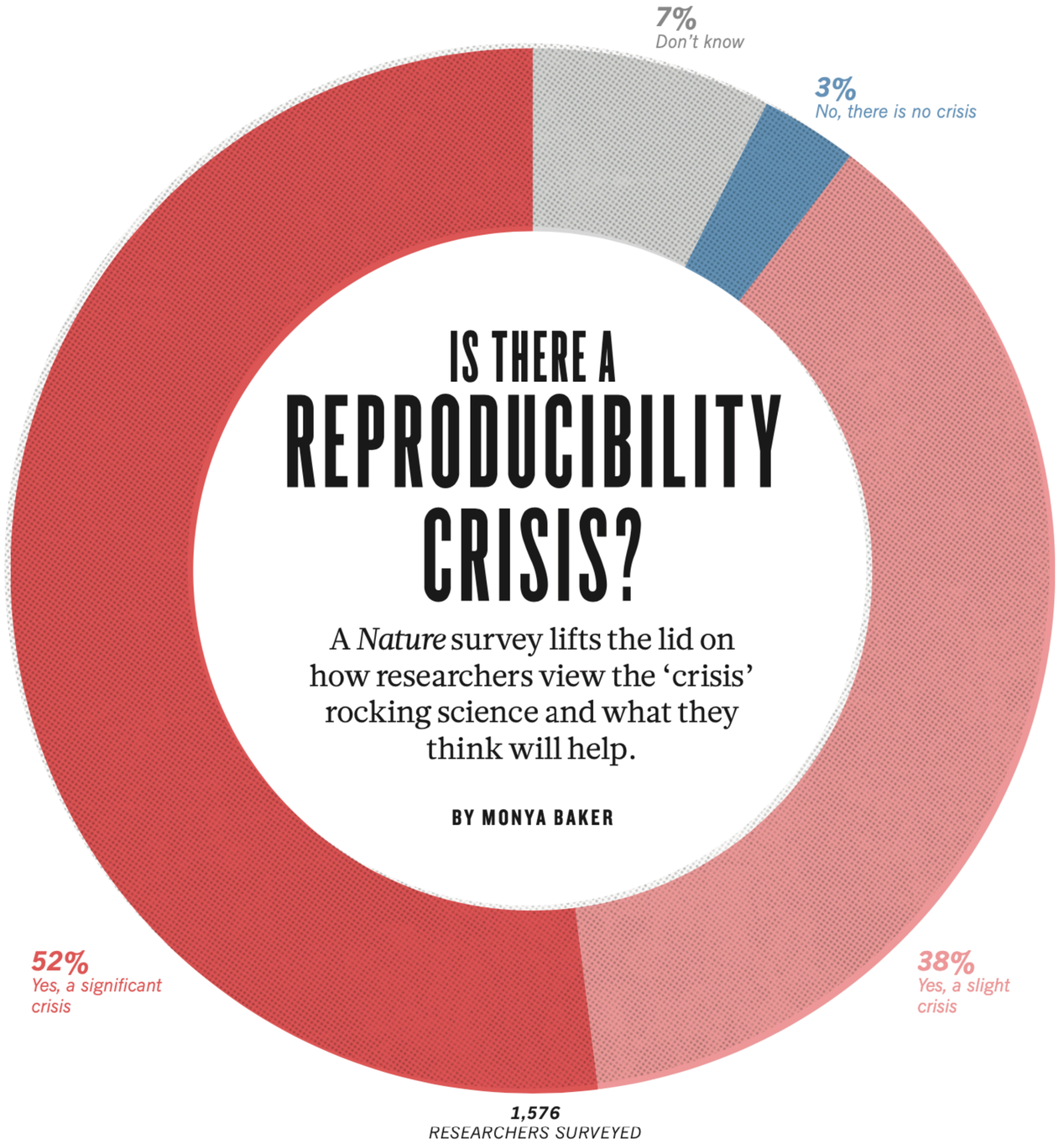
3. **Setting Up a Computational Research Project**
   Tips and Techniques for Success

4. **Git Version Control for Scientific Computing**
   Managing Code and Collaborating Effectively

5. **Introduction to BASH Scripting**
   Automating Tasks and Streamlining Workflows

The Reproducibility Crisis in Science

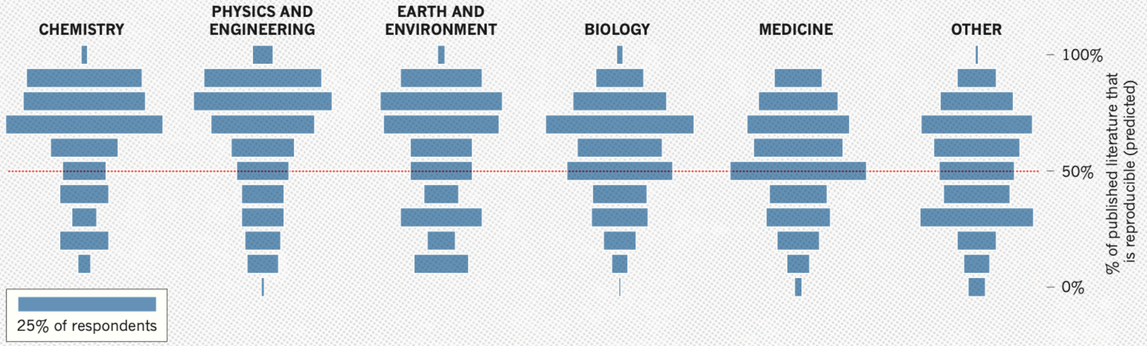
Understanding the Problem and Finding Solutions



https://doi.org/10.1038/d41586-019-00067-3

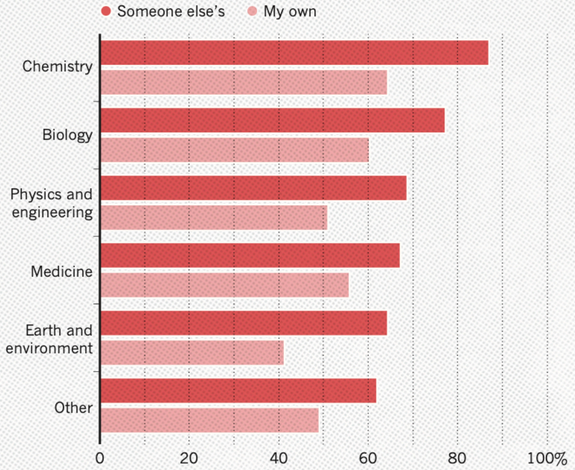
HOW MUCH PUBLISHED WORK IN YOUR FIELD IS REPRODUCIBLE?

Physicists and chemists were most confident in the literature.



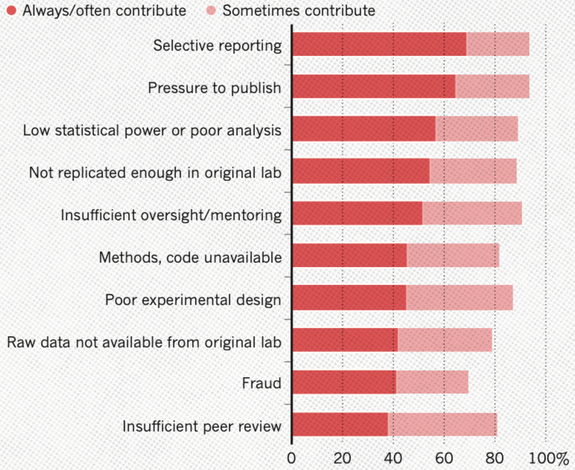
HAVE YOU FAILED TO REPRODUCE AN EXPERIMENT?

Most scientists have experienced failure to reproduce results.



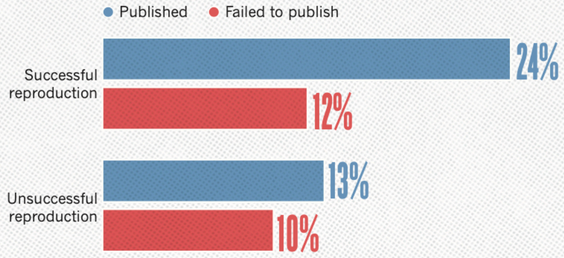
WHAT FACTORS CONTRIBUTE TO IRREPRODUCIBLE RESEARCH?

Many top-rated factors relate to intense competition and time pressure.



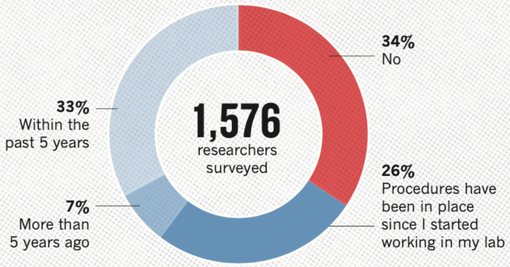
HAVE YOU EVER TRIED TO PUBLISH A REPRODUCTION ATTEMPT?

Although only a small proportion of respondents tried to publish replication attempts, many had their papers accepted.



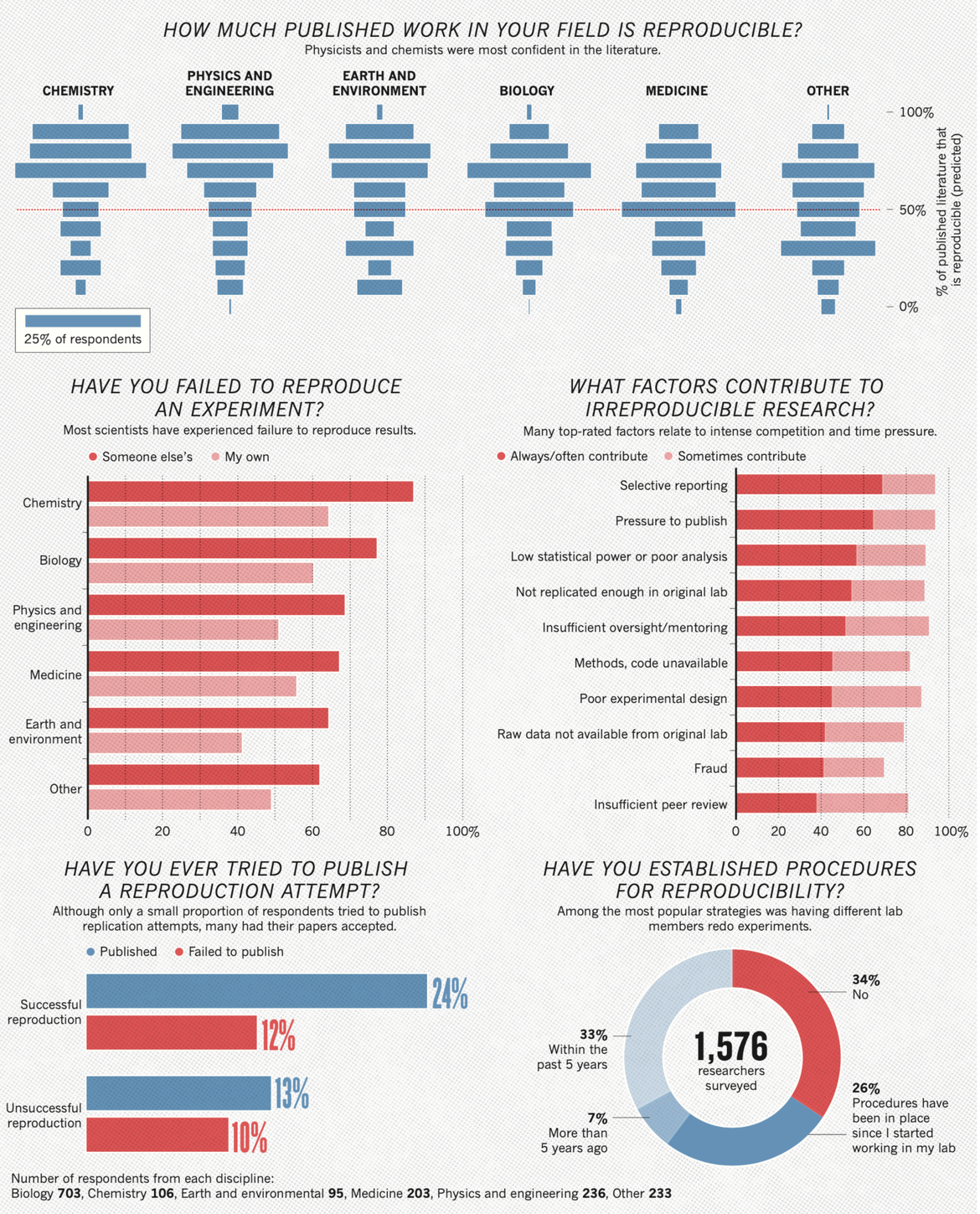
HAVE YOU ESTABLISHED PROCEDURES FOR REPRODUCIBILITY?

Among the most popular strategies was having different lab members redo experiments.



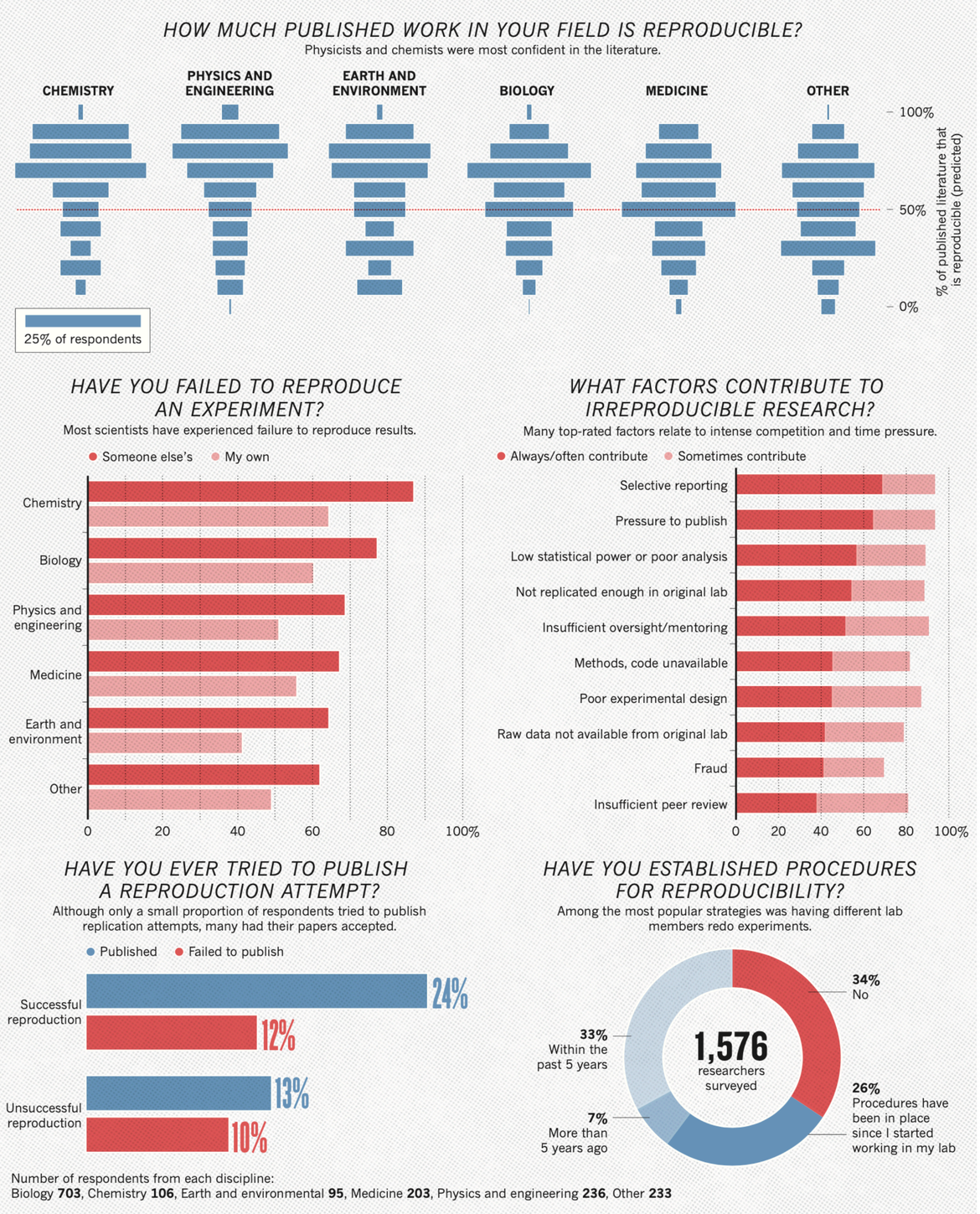
Number of respondents from each discipline:
Biology **703**, Chemistry **106**, Earth and environmental **95**, Medicine **203**, Physics and engineering **236**, Other **233**

https://doi.org/10.1038/d41586-019-00067-3



What is reproducibility in scientific research?

https://doi.org/10.1038/d41586-019-00067-3



I don't think we can define reproducibility across the board

Ferric Fang, University of Washington

Reproducibility is shorthand for a lot of problems

Jon Lorsch, National Institute of General Medical Sciences

https://doi.org/10.1038/d41586-019-00067-3

Repeatability, Replicability, Reproducibility

Repeatability,
Replicability,
Reproducibility,
Methods reproducibility,
Results reproducibility,
Inferential reproducibility,

…

Association for Computing Machinery

Repeatability - Same team, same experimental setup
a researcher can reliably repeat her own computation

Replicability - Different team, same experimental setup
an independent group can obtain the same result using the author's own artifacts.

Reproducibility - Different team, different experimental setup
an independent group can obtain the same result using artifacts which they develop completely independently

https://www.acm.org/publications/policies/artifact-review-badging

ReScience C

Reproducible Science is good. Replicated Science is better.

Reproducibility - Different team, same experimental setup

as the ability to recompute data analytic results given an observed dataset and knowledge of the data analysis pipeline.

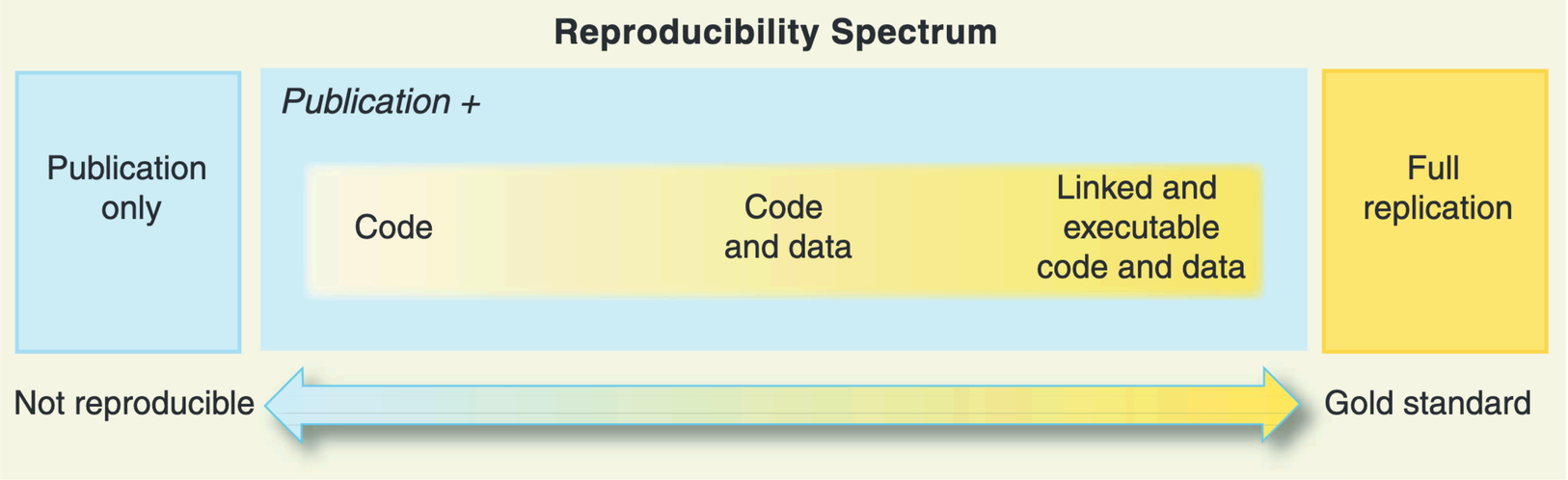
Replicability - Different team, different experimental setup

the chance that an independent experiment targeting the same scientific question will produce a consistent result

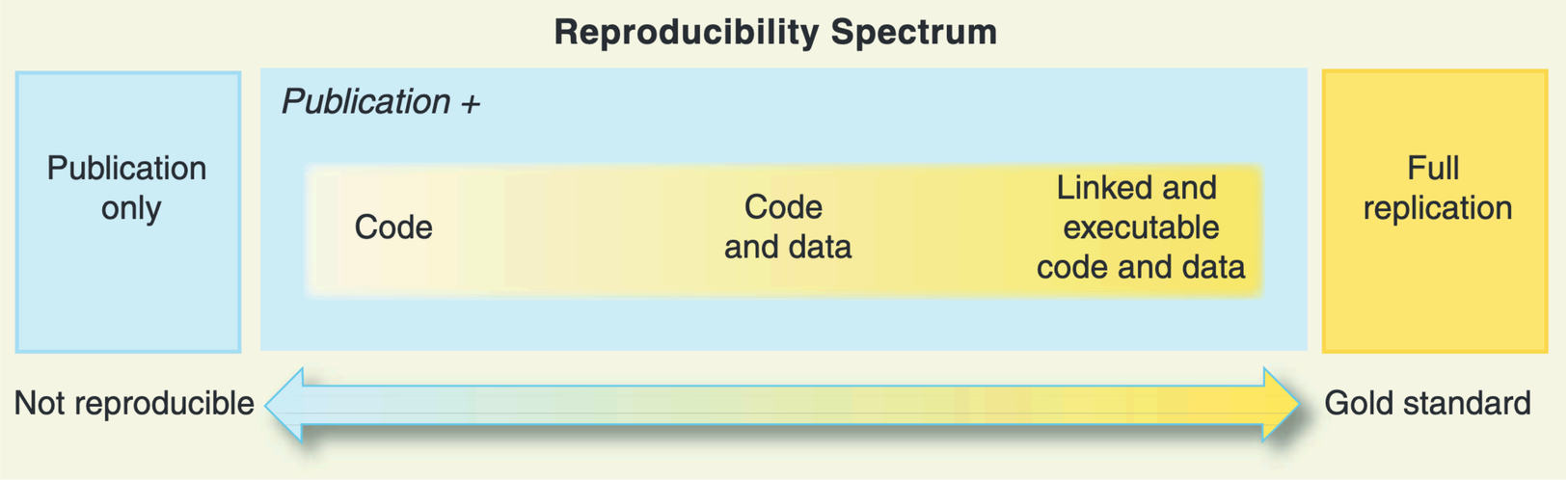
https://doi.org/10.7717/peerj-cs.142

Ensuring Reproducibility in Scientific Research

Best Practices and Strategies



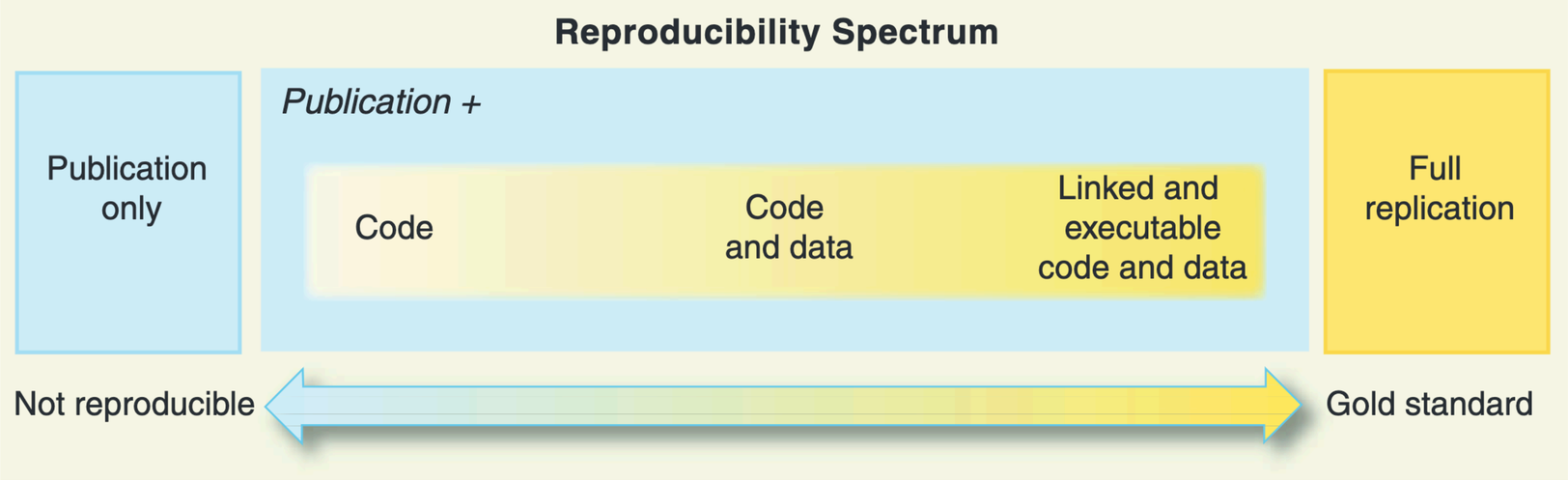
https://doi.org/10.1126/science.1213847



Before data analysis: data storage and organization

is difficult to reproduce research when data are disorganized or missing

https://doi.org/10.1002/ bes2.1801



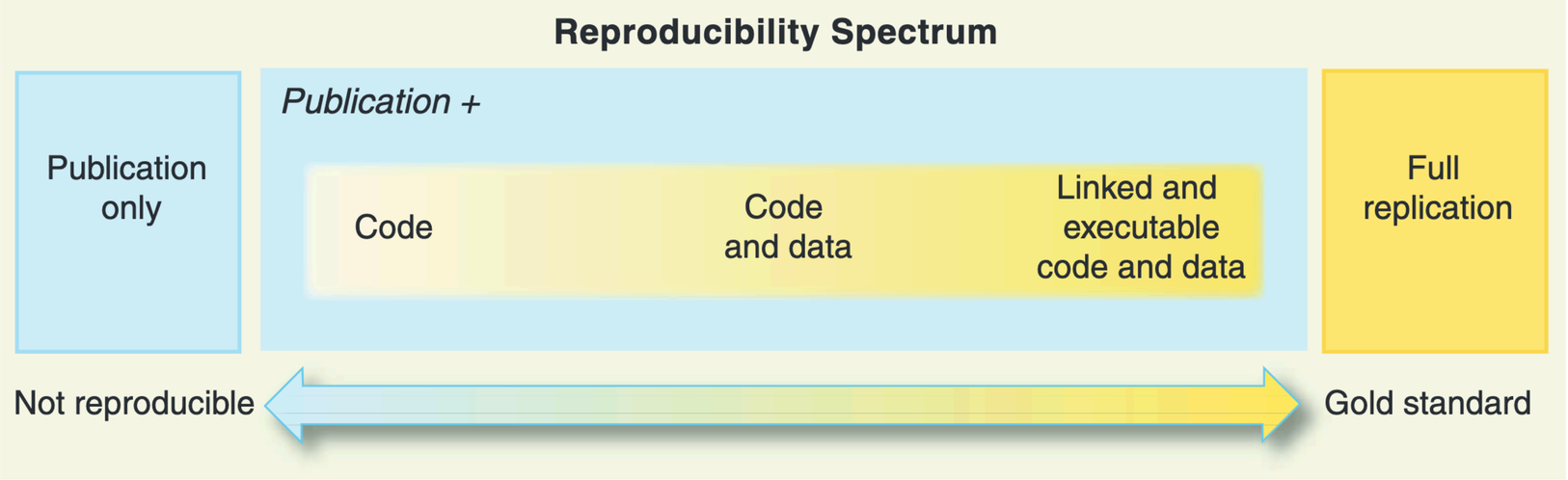
Before data analysis: data storage and organization

is difficult to reproduce research when data are disorganized or missing

During analysis: best coding practices

code available and open for re-use

https://doi.org/10.1002/ bes2.1801



Before data analysis: data storage and organization

is difficult to reproduce research when data are disorganized or missing

During analysis: best coding practices

code available and open for re-use

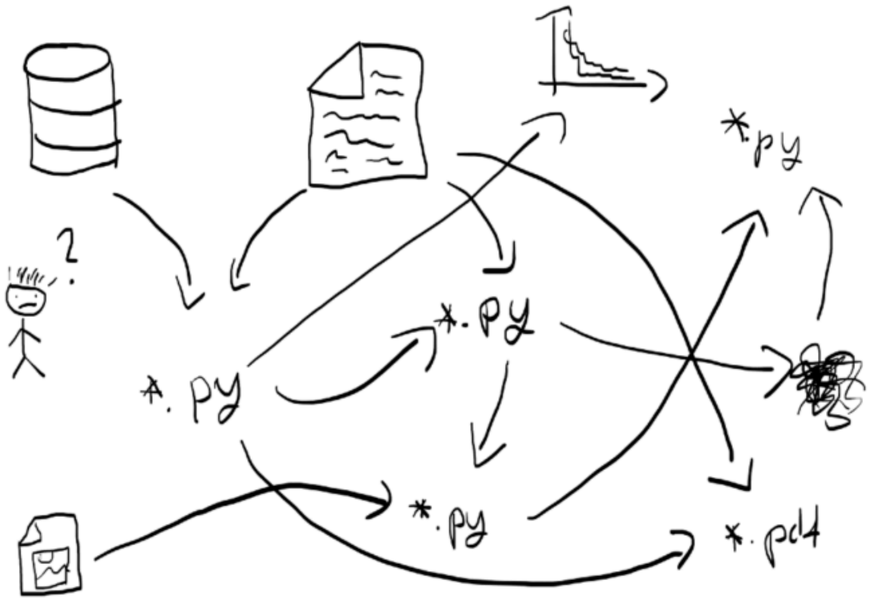
After data analysis: finalizing results and sharing

data, code, software, and products of a research project must be accessible

https://doi.org/10.1002/ bes2.1801

Setting Up a Computational Research Project

Tips and Techniques for Success



https://mitcommlab.mit.edu/be/commkit/file-structure/

Before data analysis: data storage and organization

- think twice about the data/file structure of your project
- **raw data is immutable**
- separate out its evolution
  - small projects get separate files
  - large projects get separate folders
- rename files
- keep your intermediate data organized (tidy data)
- finalized data should be clearly separated
- avoid **final_final_**

https://mitcommlab.mit.edu/be/commkit/file-structure/

File structure

small project

flat structure

```
PROJECT/
├── figures/                                          <- plots.m saves figures in here
├── main.m                                            <- does something fancy, calls plots.m
├── plots.m
├── raw2wind.m                                        <- script to clean .csv data to .mat
├── readme.txt
├── windSpeed_MITGreenBuilding.mat                    <- cleaned data
├── weatherStation_MITGreenBuilding_2019_07_01.csv    <- raw data
├── weatherStation_MITGreenBuilding_2019_07_02.csv
├── weatherStation_MITGreenBuilding_2019_07_03.csv
├── weatherStation_MITGreenBuilding_2019_07_04.csv
├── weatherStation_MITGreenBuilding_2019_07_05.csv
├── weatherStation_MITGreenBuilding_2019_07_06.csv
└── weatherStation_MITGreenBuilding_2019_07_07.csv
```

https://mitcommlab.mit.edu/be/commkit/file-structure/

File structure

medium project
simple hierarchy

```
PROJECT/
├── bin/            <- compiled binaries.
├── data/
│   ├── raw/
│   └── clean/
│
├── figures/        <- figures used in place of a "results" folder.
├── scripts/
│   ├── process/    <- scripts to maniuplate data between raw, cleaned, final stages.
│   └── plot/         <- intermediate plotting.
│
├── src
│   ├── model1/     <- various experimental models.
│   ├── model2/
│   └── model3/
│
├── LICENSE
├── Makefile
└── readme.md
```

https://mitcommlab.mit.edu/be/commkit/file-structure/

File structure

large project
complex hierarchy

```
PROJECT/
├── LICENSE
├── Makefile           <- Makefile with commands like `make data` or `make train`
├── README.md          <- The top-level README for developers using this project.
├── 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/              <- A default Sphinx project; see sphinx-doc.org for details
│
├── models/            <- Trained and serialized models, model predictions, or model summaries
│
├── notebooks/         <- Jupyter notebooks. Naming convention is a number (for ordering),
│                         the creator's initials, and a short `-` delimited description, e.g.
│                         `1.0-jqp-initial-data-exploration`.
│
├── references/        <- Data dictionaries, manuals, and all other explanatory materials.
│
├── reports/           <- Generated analysis as HTML, PDF, LaTeX, etc.
│   └── figures/       <- Generated graphics and figures to be used in reporting
│
├── requirements.txt   <- The requirements file for reproducing the analysis environment, e.g.
│                         generated with `pip freeze > requirements.txt`
│
├── setup.py           <- Make this project pip installable with `pip install -e`
├── src/               <- Source code for use in this project.
│   ├── __init__.py    <- Makes src a Python module
│   │
│   ├── data/          <- Scripts to download or generate data
│   │   └── make_dataset.py
│   │
│   ├── features/      <- Scripts to turn raw data into features for modeling
│   │   └── build_features.py
│   │
│   ├── models/        <- Scripts to train models and then use trained models to make
│   │   │                 predictions
│   │   ├── predict_model.py
│   │   └── train_model.py
│   │
│   └── visualization/ <- Scripts to create exploratory and results oriented visualizations
│       └── visualize.py
│
└── tox.ini            <- tox file with settings for running tox; see tox.testrun.org
```

https://mitcommlab.mit.edu/be/commkit/file-structure/

During analysis: best coding practices

- use version control system and make it available
  - git, mercurial, svn
- use scripts, or even better a workflow management system
  - nextflow, snakemake
- use environments and make them available and documented
  - conda, docker, singularity
- provide code samples and test data

After data analysis: finalizing results and sharing

- data and code archiving in public repositories (DOI)
  - zenodo, figshare, dryad, OSF
- prepare in advance data submission
- depending on the size of the project create **research compendiums**
  - data, code, software, and products of a research project are archived together

Git Version Control for Scientific Computing

Managing Code and Collaborating Effectively

- It was created by Linus Torvalds in 2005 to manage the development of the Linux kernel

- It was created by Linus Torvalds in 2005 to manage the development of the Linux kernel.
- Git is a **distributed version control system** used for **tracking changes in files and coordinating work** on those files among multiple people

- It was created by Linus Torvalds in 2005 to manage the development of the Linux kernel.
- Git is a **distributed version control system** used for **tracking changes in files and coordinating work** on those files among multiple people.
- Git **allows users to create, commit, and merge changes to files**, and **keep track of different versions** of the same codebase

- It was created by Linus Torvalds in 2005 to manage the development of the Linux kernel.
- Git is a **distributed version control system** used for **tracking changes in files and coordinating work** on those files among multiple people.
- Git **allows users to create, commit, and merge changes to files**, and **keep track of different versions** of the same codebase.
- It **provides a way to collaborate with others** and maintain a **complete history of all changes** made to a project over time

- It was created by Linus Torvalds in 2005 to manage the development of the Linux kernel.
- Git is a **distributed version control system** used for **tracking changes in files and coordinating work** on those files among multiple people.
- Git **allows users to create, commit, and merge changes to files**, and **keep track of different versions** of the same codebase
- It **provides a way to collaborate with others** and maintain a **complete history of all changes** made to a project over time
- Git is widely used in software development, but can be used for **any type of file-based project**

- Git **is not** GitHub

- Git **is not** GitHub

genomewalker / FunDatA Public

Pin | Unwatch 1 | Fork 0 | Star 0

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

main | 2 branches | 0 tags | Go to file | Add file | Code

Your main branch isn't protected
Protect this branch from force pushing or deletion, or require status checks before merging. Learn more

Protect this branch

genomewalker Minor — ba7a838 — 13 hours ago — **16** commits

data	Added wrapper wf	14 hours ago
day1	Renamed folders	17 hours ago
day2	Minor	13 hours ago
.gitignore	Initial commit	yesterday
LICENSE	Create LICENSE	13 hours ago
README.md	Minor	13 hours ago

README.md

Fundamentals in Computational Analysis of Large-Scale Datasets

Welcome to the GitHub repository for Reproducible data analysis and workflows module of the "Fundamentals in Computational Analysis of Large-Scale Datasets" course. This repository contains all the materials needed for the course, including two days of tasks and a wiki with detailed explanations and examples.

About

A repository for the Fundamentals Data Analysis course

Readme
MIT license
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

Python 54.0% Shell 46.0%

- Git **is not** GitHub

Home

Edit | New page

genomewalker edited this page 6 minutes ago · 7 revisions

Welcome to the wiki for **Reproducible** data analysis and workflows module of the **Fundamentals in Computational Analysis of Large-Scale Datasets** course. The wiki contains all the materials needed for the course, including two days of tasks with detailed explanations and examples.

- Day 1
  - Setting up the environment
  - Introduction to Git
  - Task 1: Writing a simple BASH script
  - Task 2: Writing a More Complex BASH script
- Day 2
  - Task 1: Writing a simple workflow
  - Task 2: Writing a More Complex Workflow
  - Task3: Mapping reads using Bowtie2
- Resources
  - Reproducible Research
  - Setting up a Project

+ Add a custom footer

Pages 12

- Day 1
  - Setting up the environment
  - Introduduction to Git
  - Task 1: Writing a simple BASH script
  - Task 2: Writing a More Complex BASH script
- Day 2
  - Task 1: Writing a simple workflow
  - Task 2: Writing a More Complex Workflow
  - Task3: Mapping reads using Bowtie2
- Resources
  - Reproducible Research
  - Setting up a Project

Clone this wiki locally

https://github.com/genomewalker

Introduction to BASH Scripting

Automating Tasks and Streamlining Workflows

- can **automate repetitive tasks** in scientific research, such as **data preprocessing**, **analysis**, and **visualization**

- can **automate repetitive tasks** in scientific research, such as **data preprocessing**, **analysis**, and **visualization**
- provides a way to **efficiently process large amounts of data** and **perform complex computations**

- can **automate repetitive tasks** in scientific research, such as **data preprocessing**, **analysis**, and **visualization**
- provides a way to **efficiently process large amounts of data** and **perform complex computations**
- can be used to **automate the execution of software tools** and **pipelines**, and to integrate multiple tools and workflows

- can **automate repetitive tasks** in scientific research, such as **data preprocessing**, **analysis**, and **visualization**
- provides a way to **efficiently process large amounts of data** and **perform complex computations**
- can be used to **automate the execution of software tools** and **pipelines**, and to integrate multiple tools and workflows
- can also be used to **create reproducible research workflows**, by documenting the steps taken and the commands executed in a script.

- can **automate repetitive tasks** in scientific research, such as **data preprocessing**, **analysis**, and **visualization**
- provides a way to **efficiently process large amounts of data** and **perform complex computations**
- can be used to **automate the execution of software tools** and **pipelines**, and to integrate multiple tools and workflows
- can also be used to **create reproducible research workflows**, by documenting the steps taken and the commands executed in a script.
- is a **valuable skill** for researchers in fields such as bioinformatics, neuroscience, and physics, where data processing and analysis are essential

Hands-on session

https://github.com/GeoGenetics/data-analysis-2025/reproducible-data-analysis