ETH zürich

IT Services

Reproducible Scientific Computing and Data Analysis

Nadia Marounina, Henry Lütcke
Scientific IT Services, ETH Zurich

March 11, 2026

Slides & Materials: https://siscourses.ethz.ch/reproducible_computing/

Overview of today's workshop

Setting the Scene

Managing your Source Code

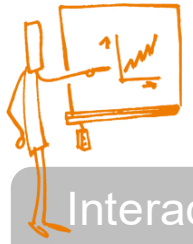
git

Managing Dependencies & Computing Environments

CONDA

Virtualizing Computing Environments

docker



Interactive Computational Notebooks

jupyter

Reproducible Computing Platforms

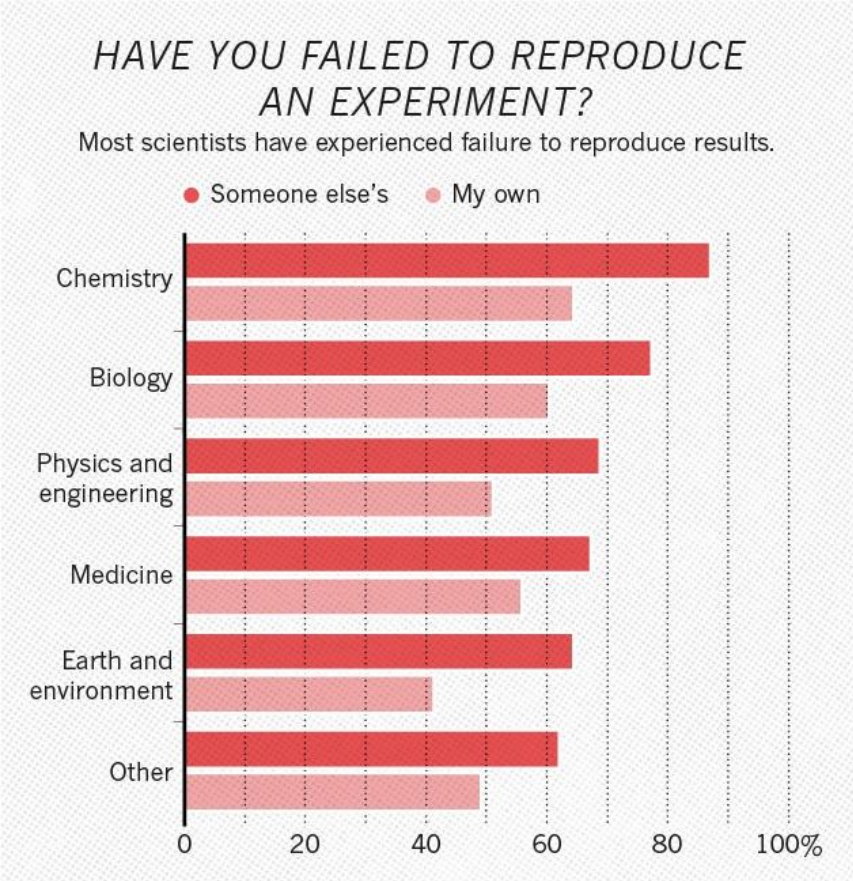
binder renku

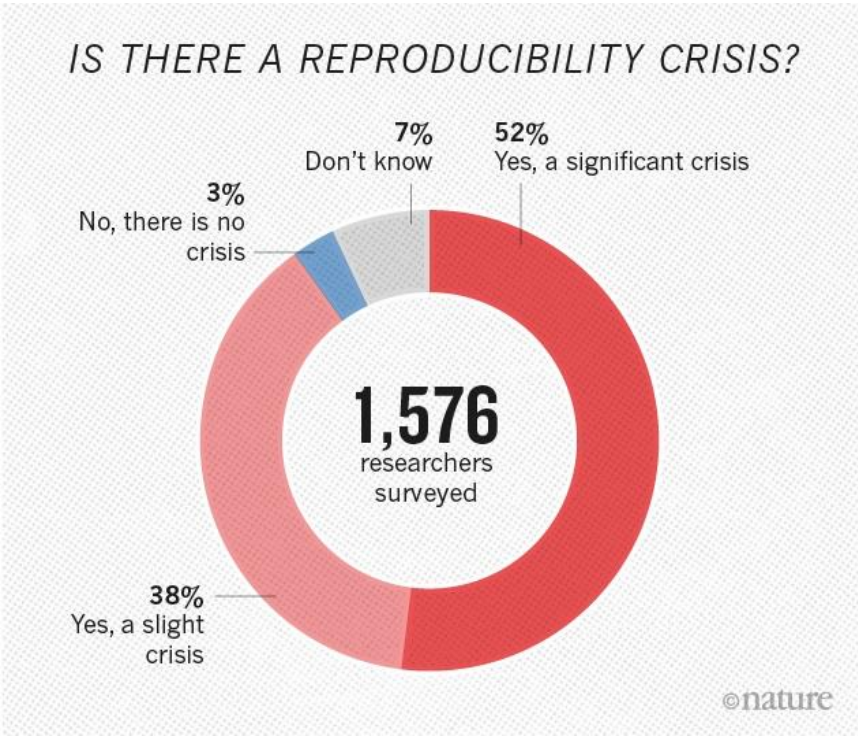
Image credit: Open Science Training Handbook

Setting the Scene

Reproducibility & Replicability in Science

Nature survey on reproducibility across all scientific domains





[Nature 533, 452–454 (26 May 2016) doi:10.1038/533452a]

Reproducibility & Replicability in Science

RESEARCH ARTICLE

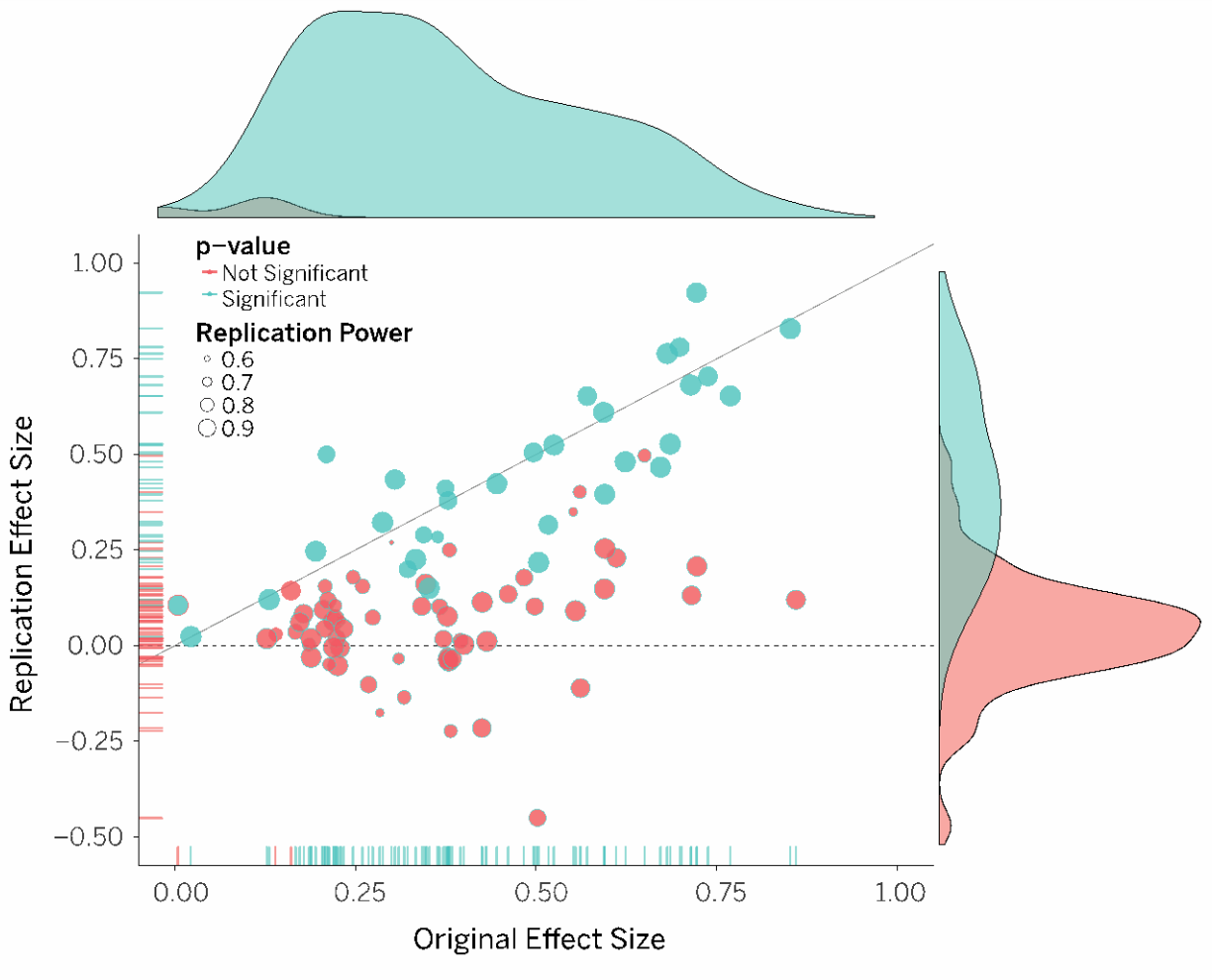
Estimating the reproducibility of psychological science

Open Science Collaboration*,†
+ See all authors and affiliations

Science 28 Aug 2015:
Vol. 349, Issue 6251, aac4716
DOI: 10.1126/science.aac4716

The *Reproducibility project*

- Replicate 100 experiments published in top psychology journals
- One-half to two-thirds of original findings could not be observed in the replication study



Reproducibility & Replicability in Science

RESEARCH ARTICLE

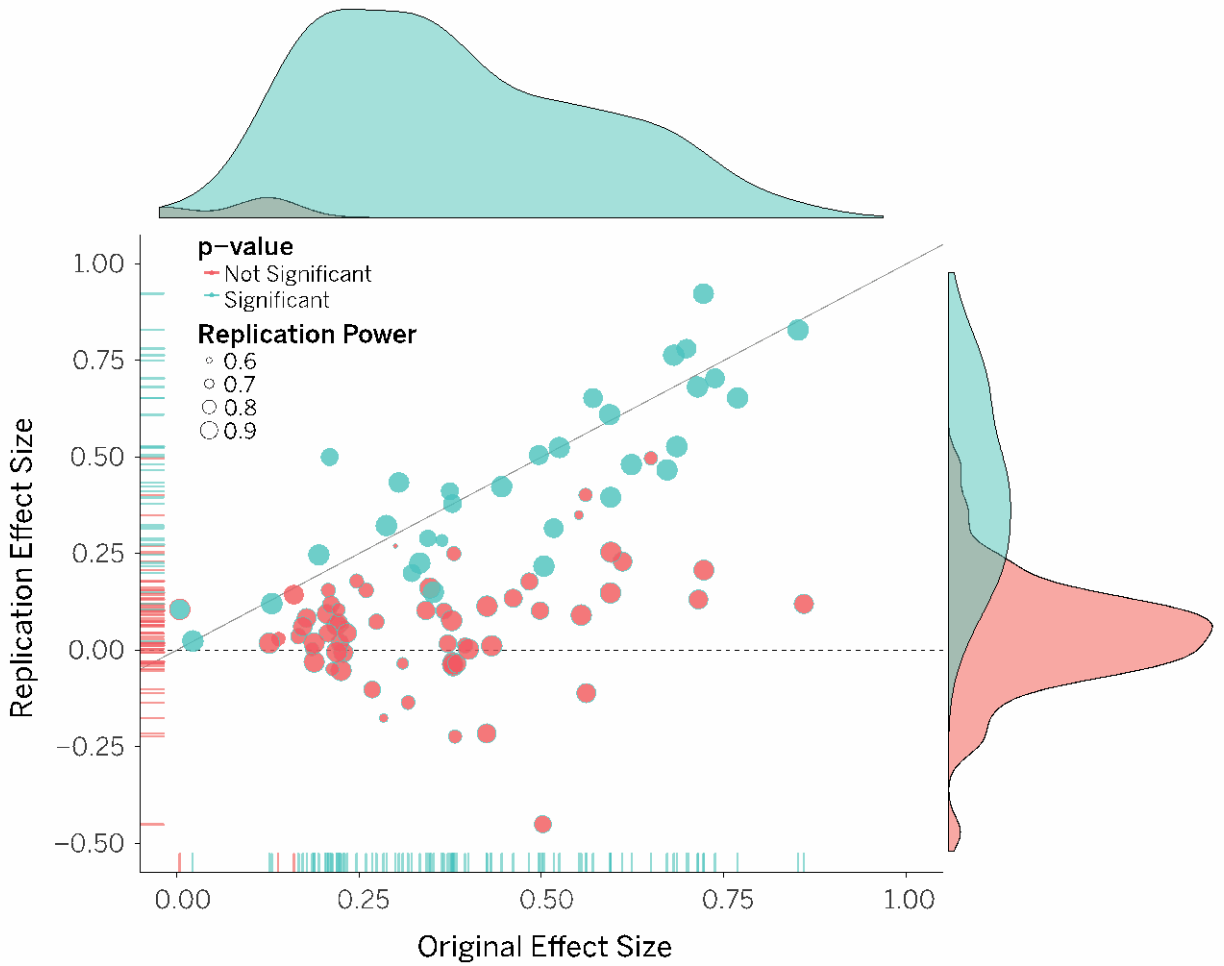
Estimating the reproducibility of psychological science

Open Science Collaboration*,†
+ See all authors and affiliations

Science 28 Aug 2015:
Vol. 349, Issue 6251, aac4716
DOI: 10.1126/science.aac4716

The *Reproducibility project*

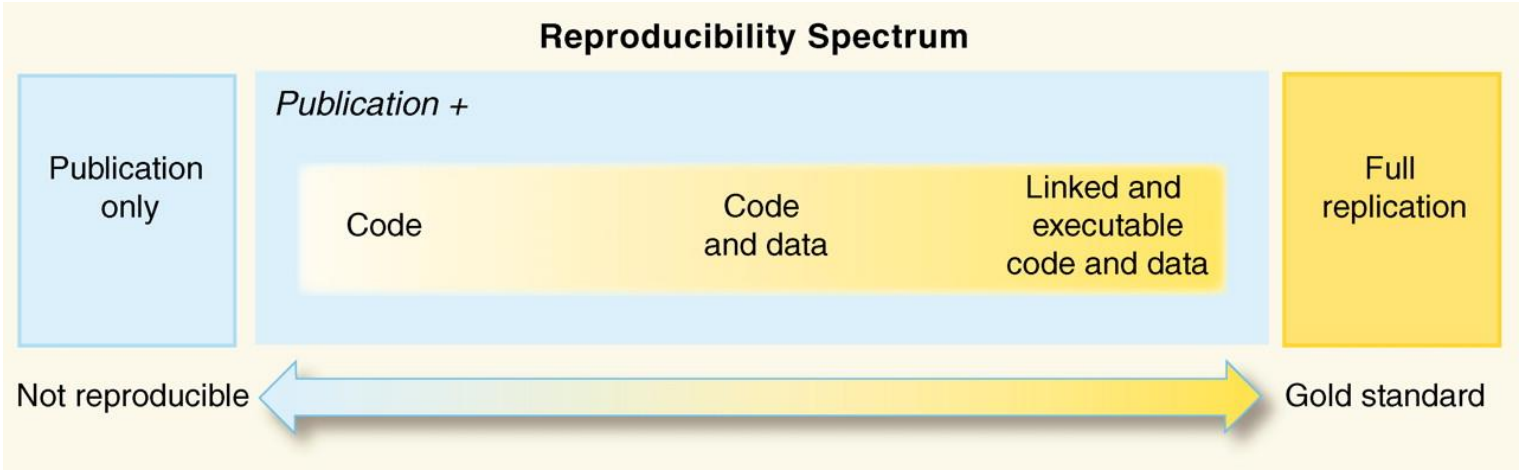
- Replicate 100 experiments published in top psychology journals
- One-half to two-thirds of original findings could not be observed in the replication study



Reproducibility & Replicability in Science

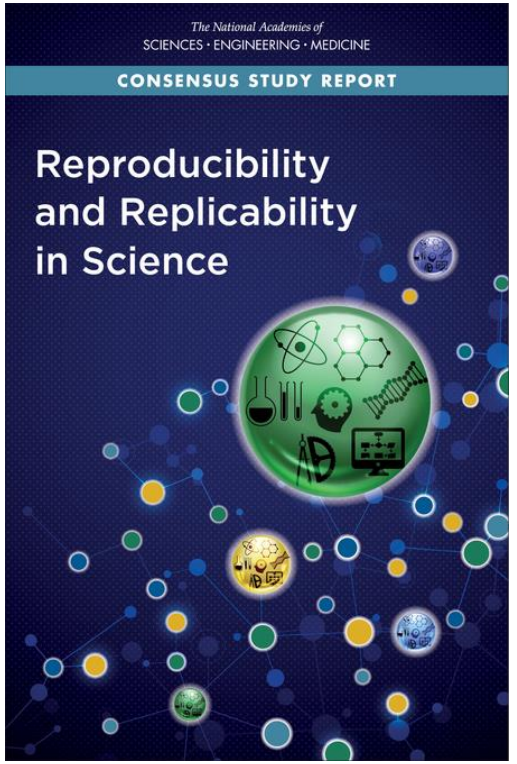
Replication:
new data and / or new method in independent study = same finding

Reproducible research:
same data + same method = same results



Peng (2011). doi:10.1126/science.1213847

Defining the Scope: Computational Reproducibility



«**Reproducibility** is obtaining consistent results using the same input data, computational steps, methods, and code and conditions of analysis. The term is synonymous with *"computational reproducibility"*... »

«To help ensure the reproducibility of computational results, researchers should **convey clear, specific, and complete information about any computational methods and data products that support their published results in order to enable other researchers to repeat the analysis**, unless such information is restricted by non-public data policies. That information should include the data, study methods, and **computational environment**. »

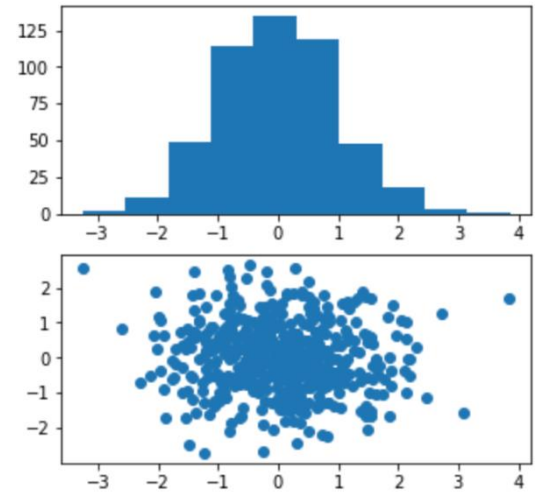
National Academies of Sciences, Engineering, and Medicine (2019). https://doi.org/10.17226/25303

Reproducibility ≠ Correctness

Computational Reproducibility: What can go wrong?

- Code only runs on specific **operating system**
  - Examples: Windows / Linux scripts, special programs (e.g. *SigmaPlot*)
- Code has specific **external dependencies**
  - Example: wget https://zenodo.org/record/1234567/files/dataset.zip
- Code has specific **internal dependencies** (libraries, modules etc.)

```python
import matplotlib.pyplot as plt
import numpy as np

np.random.seed(42)
data = np.random.randn(2, 500)

fig, axs = plt.subplots(2, 1, figsize=(5, 5))
axs[0].hist(data[0])
axs[1].scatter(data[0], data[1])

plt.show()
```



Computational Reproducibility: What can go wrong?

- Code only runs on specific **operating system**
  - Examples: Windows / Linux scripts, special programs (e.g. *SigmaPlot*)
- Code has specific **external dependencies**
  - Example: wget https://zenodo.org/record/1234567/files/dataset.zip
- Code has specific **internal dependencies** (libraries, modules etc.)
- Code has specific **version dependencies**
- Code may rely on availability of specific **software licenses**
  - Example: fastaread function in the MATLAB Bioinformatics Toolbox

```
import numpy as np

print("Using Numpy %s" % np.__version__)

rng = np.random.default_rng(42)
rng.dirichlet((0.04, 0.03), 2)
```

```
Using Numpy 1.18.1
array([[2.10122596e-01, 7.89877404e-01],
       [1.99456813e-22, 1.00000000e+00]])
```

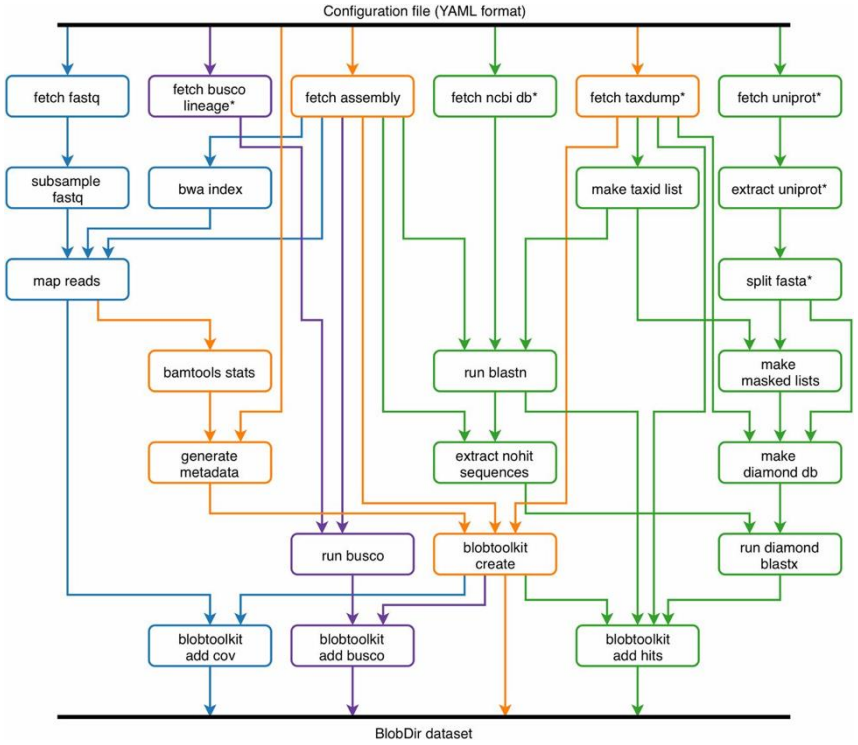
```
import numpy as np

print("Using Numpy %s" % np.__version__)

rng = np.random.default_rng(42)
rng.dirichlet((0.04, 0.03), 2)
```

```
Using Numpy 1.20.2
array([[9.99999999e-01, 7.24826532e-10],
       [9.99726345e-01, 2.73654825e-04]])
```

See *https://numpy.org/doc/stable/release/1.19.0-notes.html#changed-random-variate-stream-from-numpy-random-generator-dirichlet*

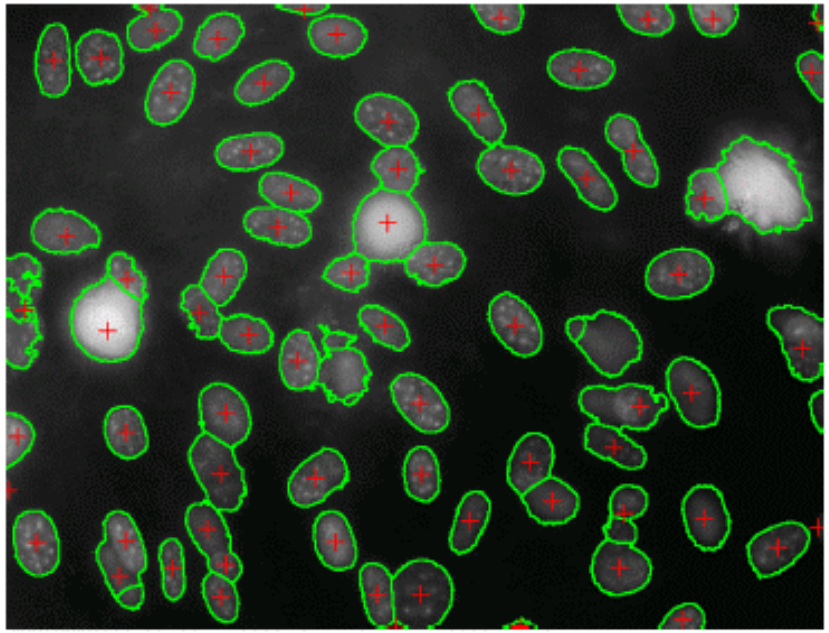
Computational Reproducibility: What can go wrong?

- Code only runs on specific **operating system**
  - Examples: Windows / Linux scripts, special programs (e.g. *SigmaPlot*)
- Code has specific **external dependencies**
  - Example: wget https://zenodo.org/record/1234567/files/dataset.zip
- Code has specific **internal dependencies** (libraries, modules etc.)
- Code has specific **version dependencies**
- Code may rely on availability of specific **software licenses**
  - Example: fastaread function in the MATLAB Bioinformatics Toolbox
- Code may be **incomprehensible** (complex, undocumented workflows)



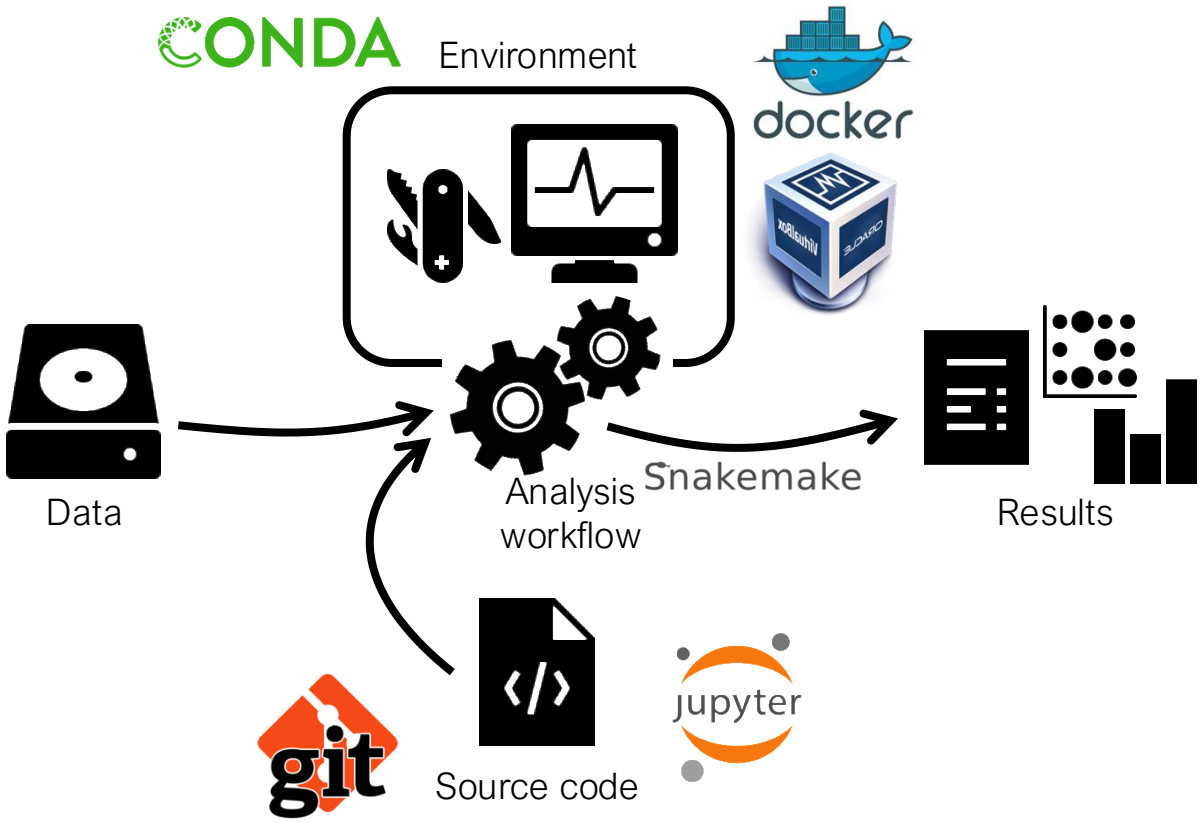
Computational Reproducibility: What can go wrong?

- Code only runs on specific **operating system**
  - Examples: Windows / Linux scripts, special programs (e.g. *SigmaPlot*)
- Code has specific **external dependencies**
  - Example: `wget` https://zenodo.org/record/1234567/files/dataset.zip
- Code has specific **internal dependencies** (libraries, modules etc.)
- Code has specific **version dependencies**
- Code may rely on availability of specific **software licenses**
  - Example: `fastaread` function in the MATLAB Bioinformatics Toolbox
- Code may be **incomprehensible** (complex, undocumented workflows)
- Analysis workflow may rely on **manual steps**



Computational Reproducibility: Pieces of the Puzzle

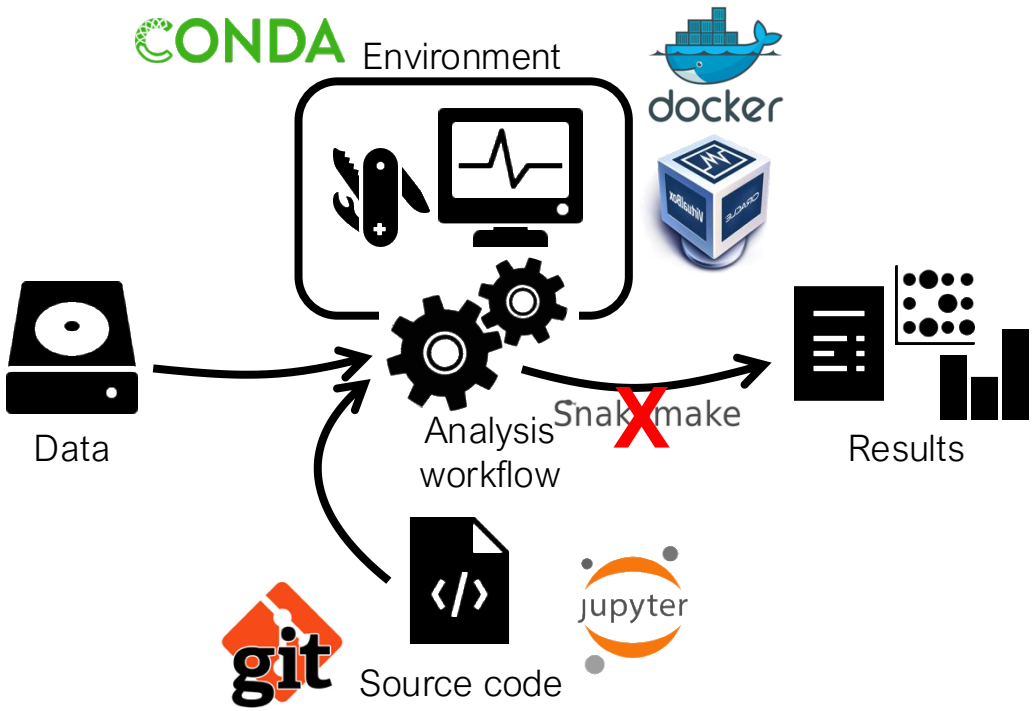
All parts of a computational analysis have to be reproducible!



Slide credit: L. Wigge, R. Ågren & J. Sundh (NBIS & SciLifeLab, Sweden)

Computational Reproducibility: Pieces of the Puzzle

What is covered in today's workshop?



Part 1 — **Version control**
Track and backup your project history

Part 2 — **Environment management**
Setup and manage your project environment

Part 3 — **Virtual machines / containers**
Make your project self-contained and distributable

Part 4 — **Notebooks**
Document your exploratory analysis

Part 5 — **Reproducible computing platforms**
Out-of-the-box computational reproducibility

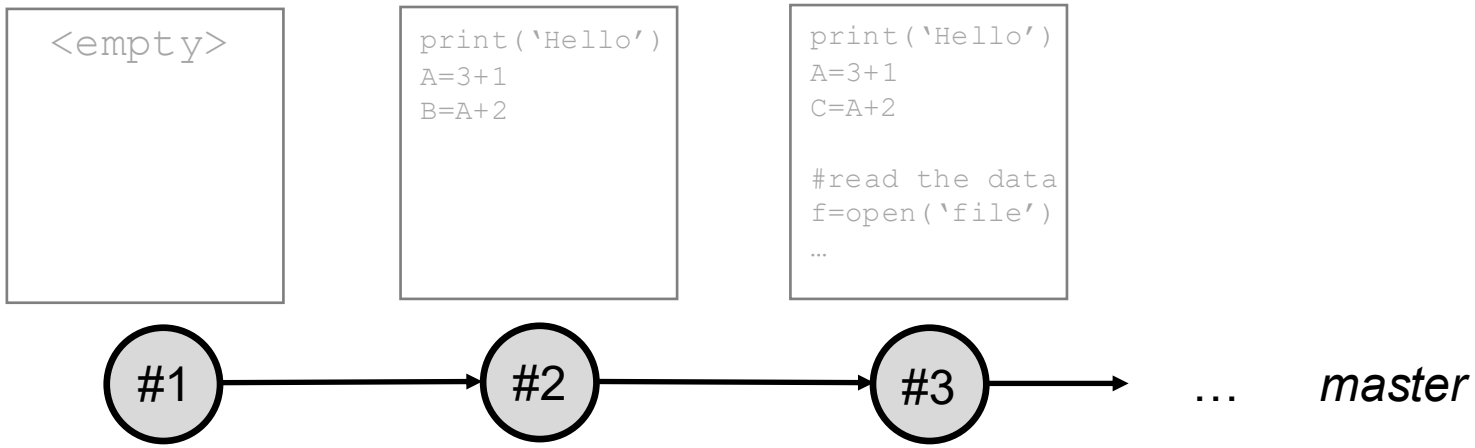
Computational Reproducibility: Questions?

Tell us a bit about yourself

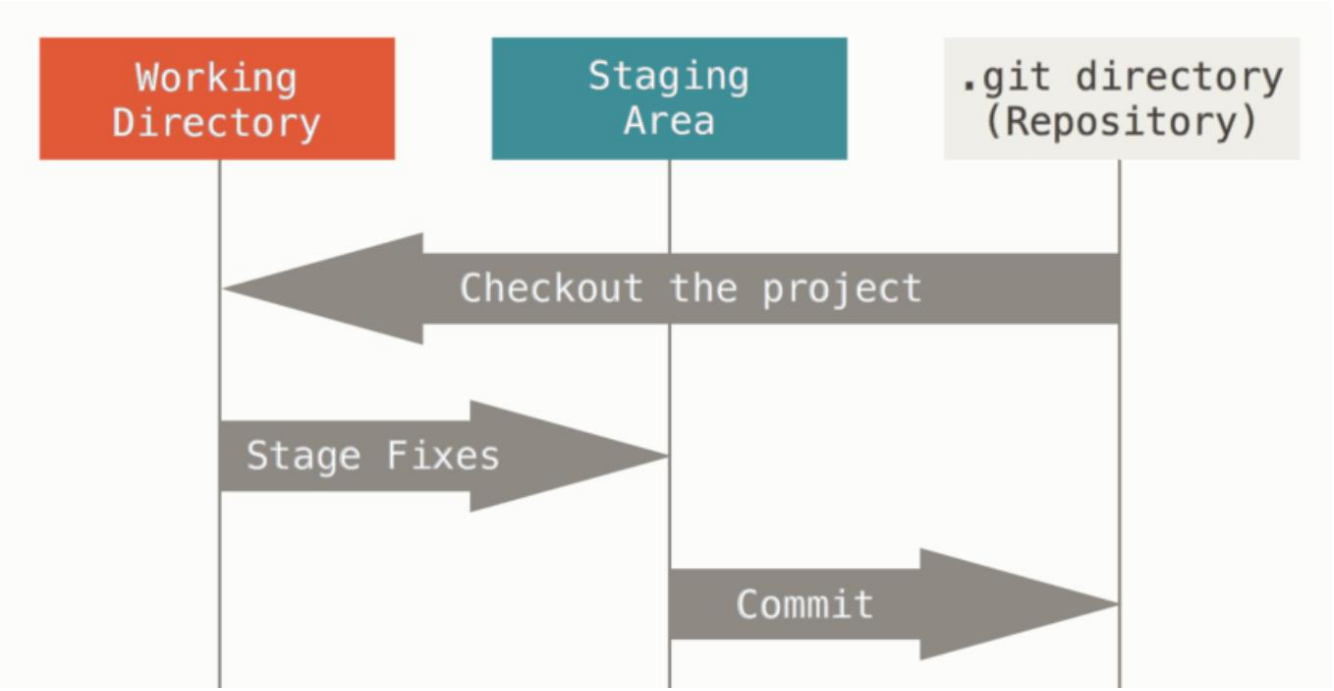
Managing your Source Code

Code Management

- Code management is the process of handling changes in source code
- Proper code management is essential to ensure **reproducible results**
- Professional code management relies on **Version Control Systems** (VCS)
  - Version control: tracking changes made to text files over time
- **Git** is by far the most popular version control system used world-wide in the software community



How do I track the changes in my code with git?



The basic Git workflow

- Modify files in your working directory
- Selectively stage the changes you want to be part of your next commit, adding **`only`** those changes to the staging area
- Make a commit, which takes the files as they are in the staging area and stores that snapshot permanently to your .git directory

[demo]

Test case : a program that takes in three files and print their content. Text_1.txt contains the string “one”, text_2.txt ”two”, etc

```
git_demo 13:58:33 >>ls
total 32
-rw-r-xr-x  1 nmarounina  staff  49 Mar  7 13:57 print_all.sh
-rw-r--r--  1 nmarounina  staff   4 Mar  7 13:54 text_1.txt
-rw-r--r--  1 nmarounina  staff   4 Mar  7 13:54 text_2.txt
-rw-r--r--  1 nmarounina  staff   6 Mar  7 13:54 text_3.txt
git_demo 13:59:00 >>./print_all.sh
one
two
three
git_demo 13:59:02 >>
```

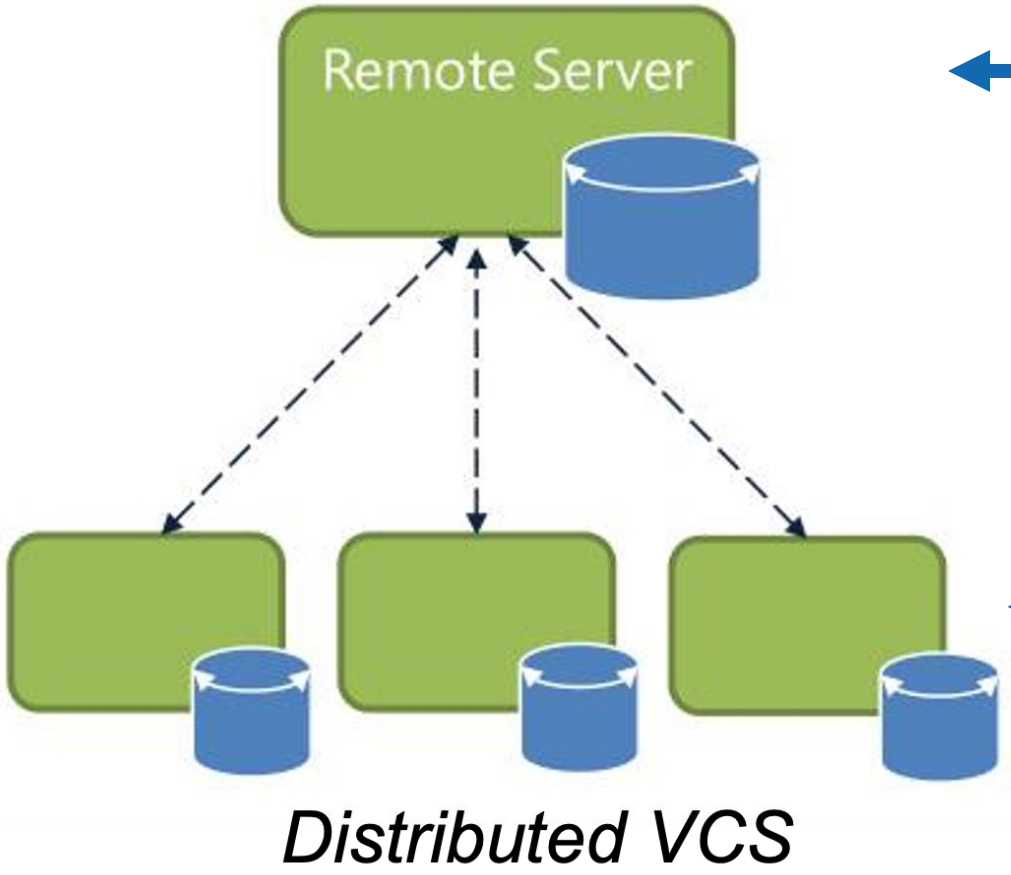
Start with git :

```
git_demo 13:59:20 >>git init #initialises git
Initialized empty Git repository in /Users/nmarounina/Desktop/git_demo/.git/
git_demo 13:59:24 >>
git_demo 13:59:34 >>git add * #adds all files to the staging
git_demo 13:59:40 >>git status #prints information about the current staging area
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   print_all.sh
        new file:   text_1.txt
        new file:   text_2.txt
        new file:   text_3.txt

git_demo 13:59:50 >>
```

First commit :

```
git_demo 13:59:52 >>git commit -m "Initial commit” #creating the first commit/snapshot
[main (root-commit) d5badf3] Initial commit
 4 files changed, 5 insertions(+)
 create mode 100755 print_all.sh
 create mode 100644 text_1.txt
 create mode 100644 text_2.txt
 create mode 100644 text_3.txt
git_demo 14:00:16 >>git log #lists all of the commits for this project
commit d5badf3593de0e511005eee061132d77cdde0823 (HEAD -> main)
Author: Nadia Marounina <nmarounina@ethz.ch>
Date:   Thu Mar 7 14:00:10 2024 +0100

    Initial commit
git_demo 14:00:20 >>
```

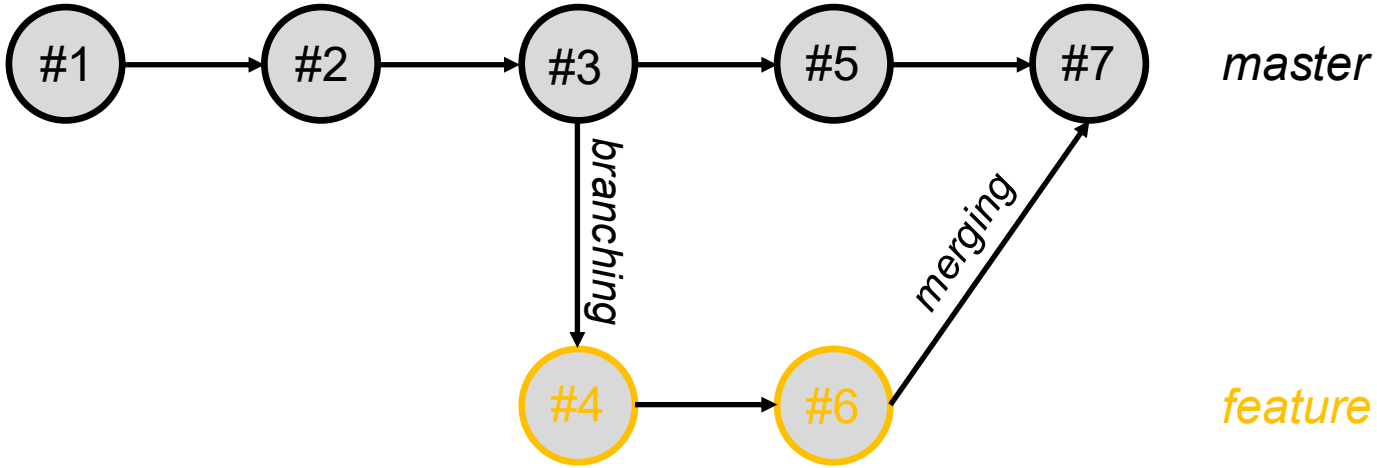
Git : How to share my code with others ?



GitHub, Gitlab, etc..

Personal workstations

Git branching & merging



Git branches & merges

- The initial / default branch is typically called *master* or *main*
- Git manages branches very efficiently
- When merging merging branches, conflicts must be resolved carefully

[demo]

Creating a new branch:

```
git_demo 14:03:15 >>git branch numbers #creates a new branch named ”numbers”
git_demo 14:04:00 >>git status
On branch main
nothing to commit, working tree clean
git_demo 14:04:03 >>git branch #list all branches for the project
* main
  numbers
git_demo 14:04:35 >>git checkout numbers #switch to the new branch
Switched to branch 'numbers'
git_demo 14:04:53 >>
```

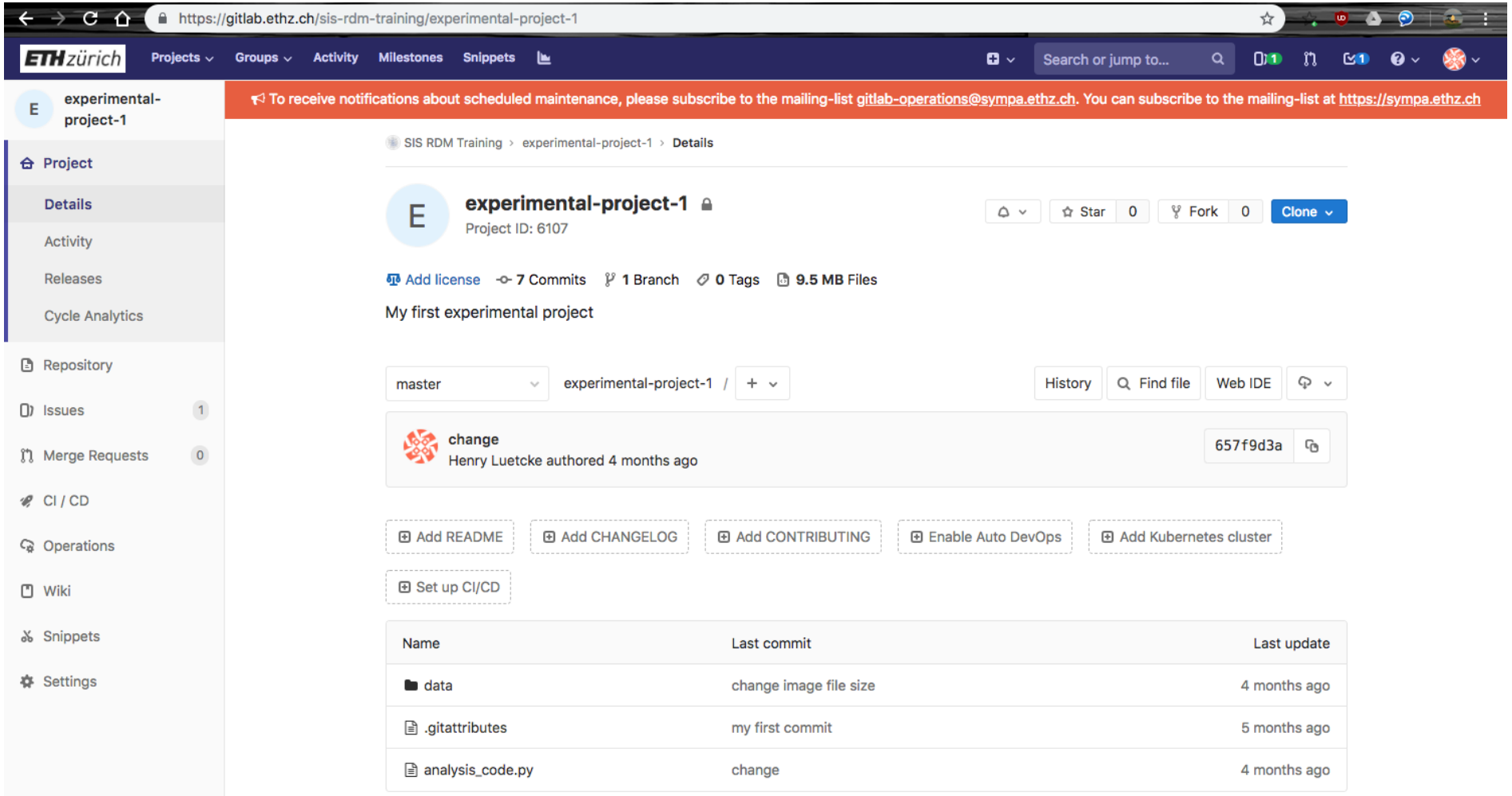
After changing the three text files in the new branch and commiting it again :

```
git_demo 14:04:56 >>vi text_1.txt #vi is a text editor. Here I change ‘one’ to ‘1’…
git_demo 14:05:07 >>vi text_2.txt #... ‘two’ to ‘2’
git_demo 14:05:16 >>vi text_3.txt  #... ‘three’ to ‘3’
git_demo 14:05:29 >>./print_all.sh
1
2
3
git_demo 14:05:37 >>git commit -m "Changed from text to number” #the change has been committed
[… output excluded …]
git_demo 14:05:51 >>
```

By switching branches, you change your files in your folder:

```
git_demo 14:06:39 >>git checkout main
Switched to branch 'main'
git_demo 14:07:29 >>./print_all.sh
one
two
three
git_demo 14:07:40 >>git checkout numbers
Switched to branch 'numbers'
git_demo 14:07:45 >>./print_all.sh
1
2
3
git_demo 14:07:46 >>
```

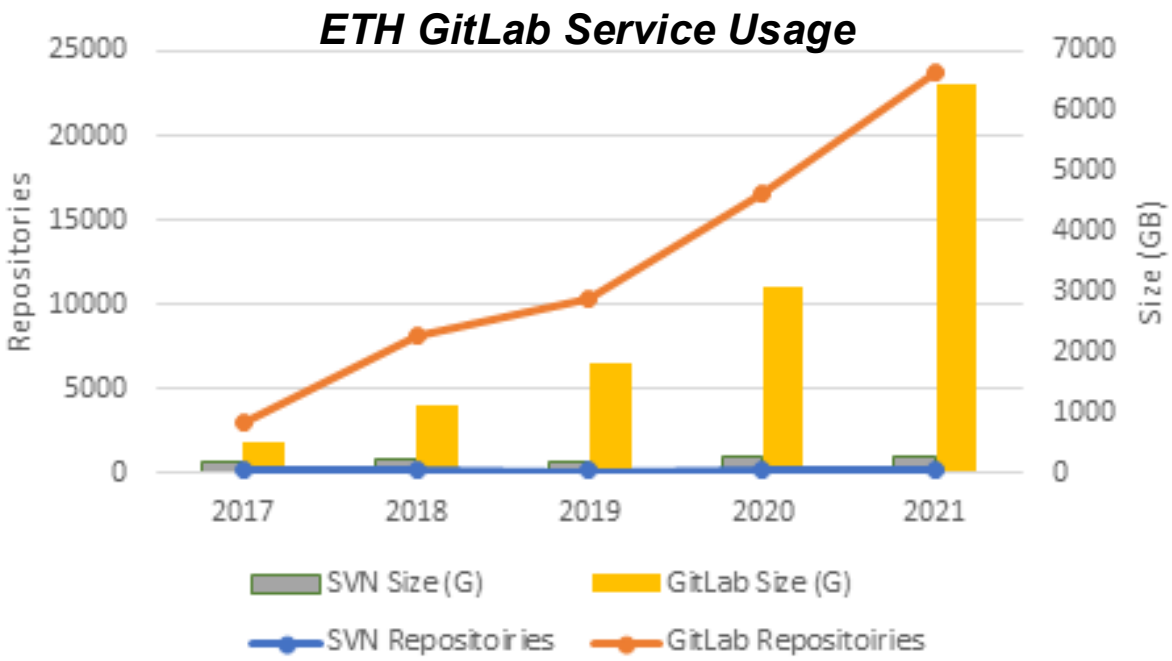
ETH Zurich GitLab Service



https://gitlab.ethz.ch

ETH Zurich GitLab Service

- Integrated file, task and documentation management for individuals and / or groups
- Private, group and public repositories
- Built-in light-weight Wiki (protocols, list of materials etc.)
- Free for small repositories (< 2GB), otherwise yearly price of 250 CHF / TB / year
- Local and remote copies (off-site backup)
- Data can be exported (e.g. to Github)
- Built-in Container registry



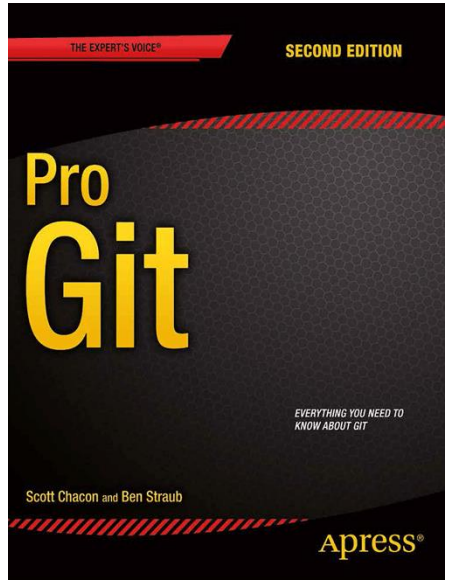
Git – General Recommendations & Resources

Recommendations for working with Git

- Commit early & often
- Provide short but meaningful commit messages
- Do not store large data files in Git repositories
  - e.g. images, movies, binary files
  - Use *.gitignore* file to exclude
  - Or consider tools such as git-lfs or git-annex
- Beware when resolving conflicts during *merge* or *pull* operations
  - A successful merge for Git may not be a successful merge for you

Resources for getting started with Git

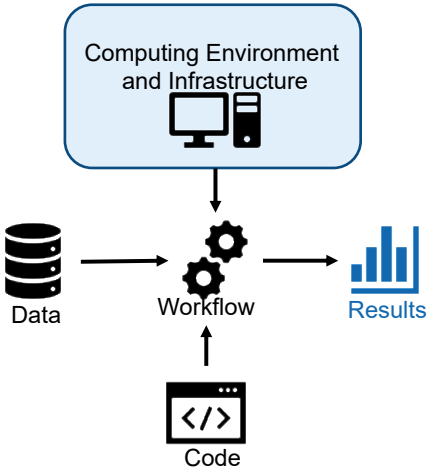
- SIS can provide hands-on Git tutorials / workshops
- Pro Git book by S. Chacon & B. Straub
- Numerous tutorials available on the web / YouTube
  - W3Schools Git tutorial
  - Software Carpentry Git course
  - Git tutorial for scientists
- List of Git GUI clients



Management of source code: Questions?

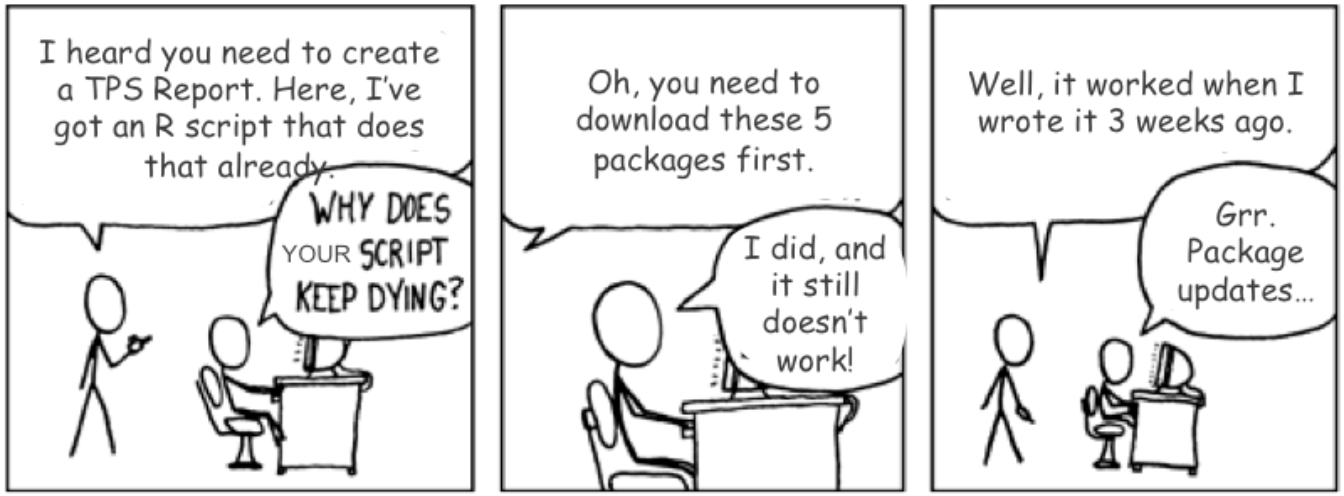
Managing Dependencies & Computing Environments

Reproducible Computing Environment

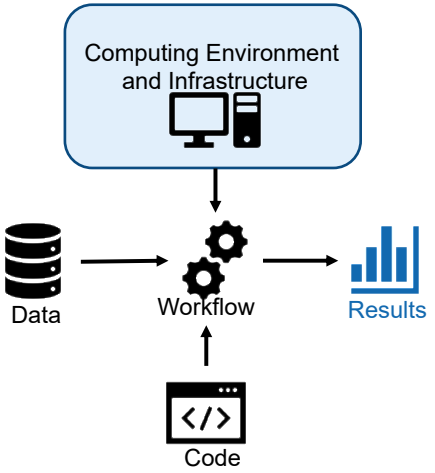


Problem:

Full reproducibility requires the possibility to recreate the system that was originally used to generate the results



Reproducible Computing Environment



Problem:

Full reproducibility requires the possibility to recreate the system that was originally used to generate the results

Solution:

Bundle your application and all dependencies

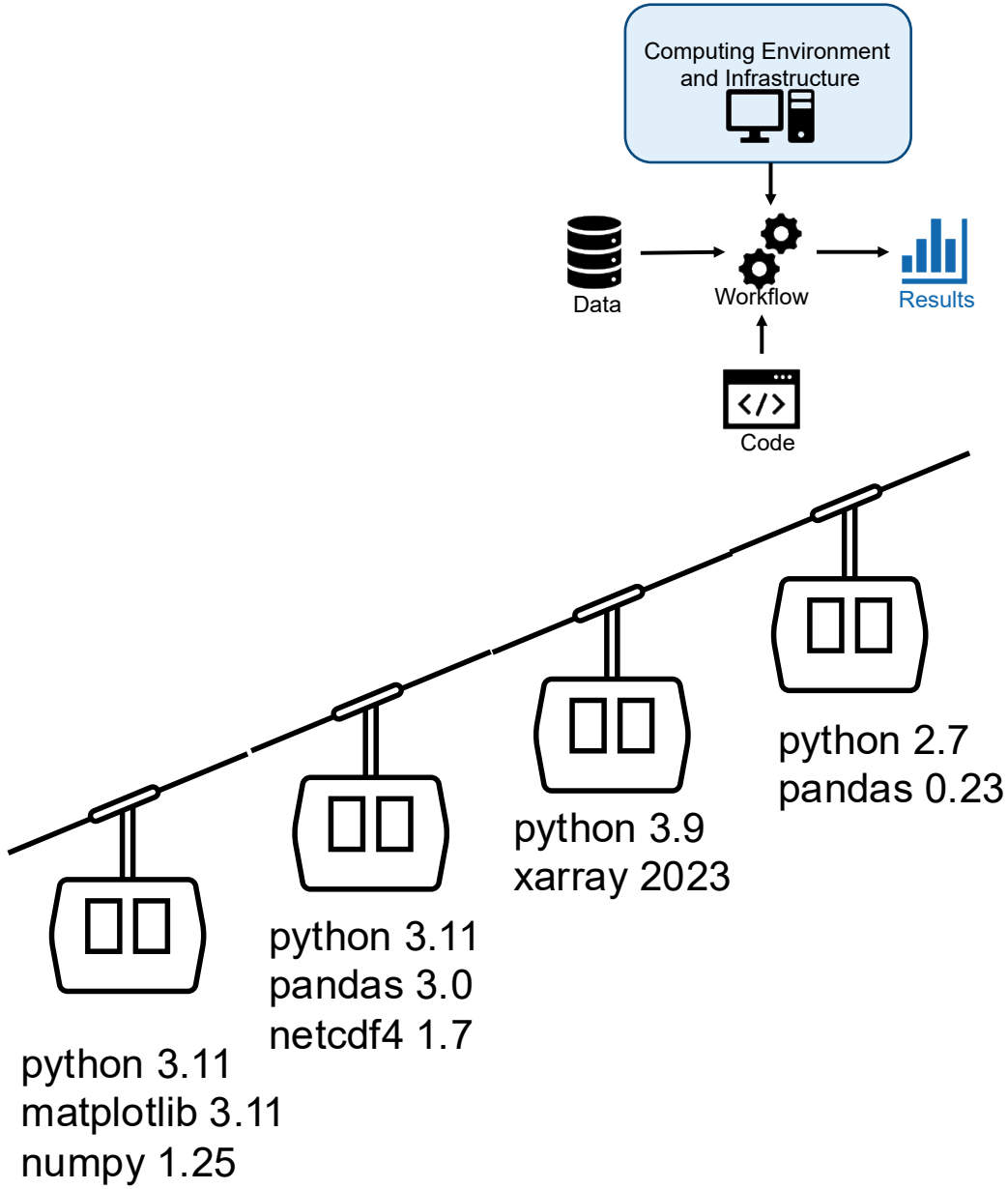
→ Environment Isolation & Dependency management

Tools:

- Application / software level: Conda, pip, virtualenv, renv, Devbox
- Containerization: Docker
- Virtualization (Virtual Machine, VM): VirtualBox, Vmware, Parallels

Virtual Computing Environments

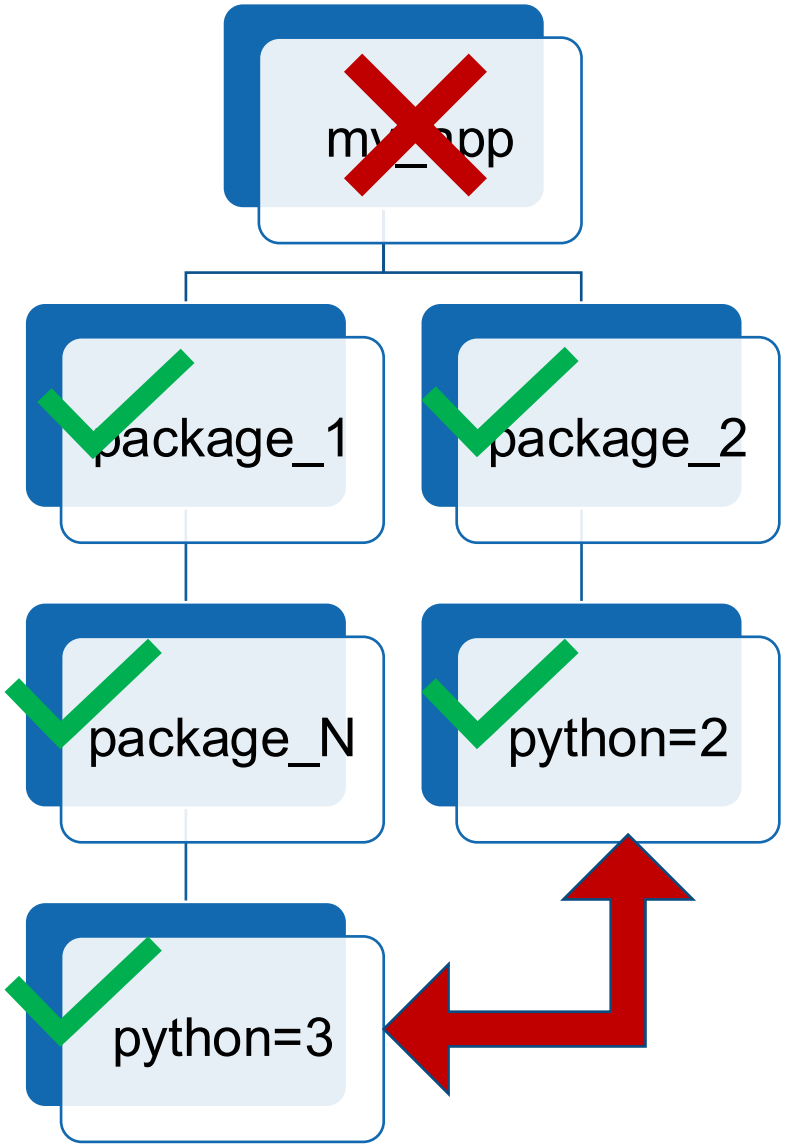
- Separated, self-contained builds of your coding language of choice (e.g. python, R, Julia, etc.)
- Great for projects that are solely in your coding language
- Some tools also work for projects with dependencies in multiple languages
- We recommend one virtual environment for each coding project



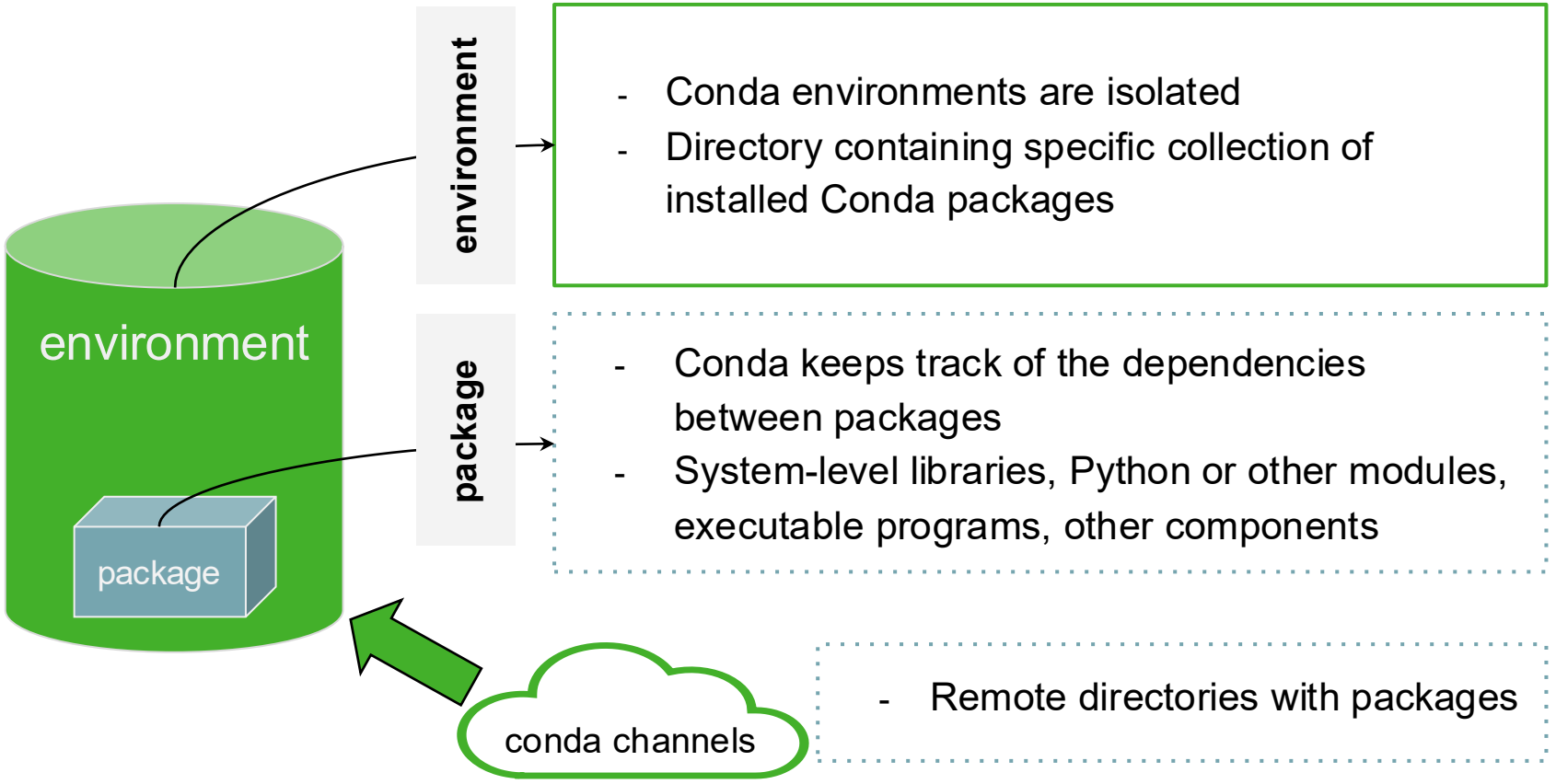
Slide credit: E. Knappe (Lib4RI)

Reproducible Environment for R and Python

- Open source: Anaconda, Miniconda, Miniforge
- Commercial support: Anaconda Enterprise
  - ***Note:*** *certain functionality requires a paid license outside education / academia*
- Multi-platform: Windows, macOS, Linux
- Environment Management System
  - Isolated computing environments on the same system
  - Documentation of the computing environment
- Package Management System
  - Supported programming Languages: Python, R, …
  - System libraries shipped in binary format
  - Resolve dependencies & conflicts between packages



Conda in a Nutshell



environment.yml

```
channels:
- conda-forge

dependencies:
- python=3.8
- jupyterlab
```

Conda automatically creates an environment file with packages and dependencies

Slide credit: L. Wigge, R. Ågren & J. Sundh (NBIS & SciLifeLab, Sweden)

Environment and Package Management Systems

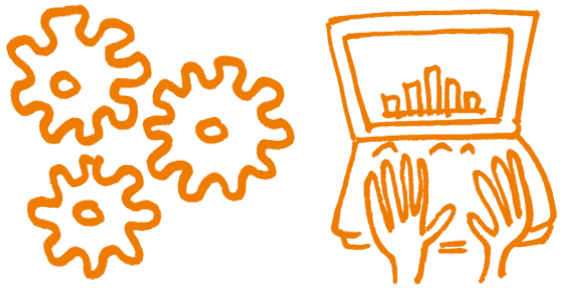
Language	Environment Management	Package Management	Comments
Python 2 (not supported)	virtualenv, conda	pip, conda	
Python 3	venv, virtualenv, pipenv poetry, uv, conda	pip, pipenv, poetry, uv, conda	Not all can install different Python versions
R	renv, conda	renv, conda	only conda can install different R versions
Julia	Pkg, conda	Pkg, conda	conda provides outdated Julia versions
Matlab	N/A	Add-on manager, Matlab Package Manager (unofficial)	Matlab search path determines dependencies

Alternatives to Conda are emerging!

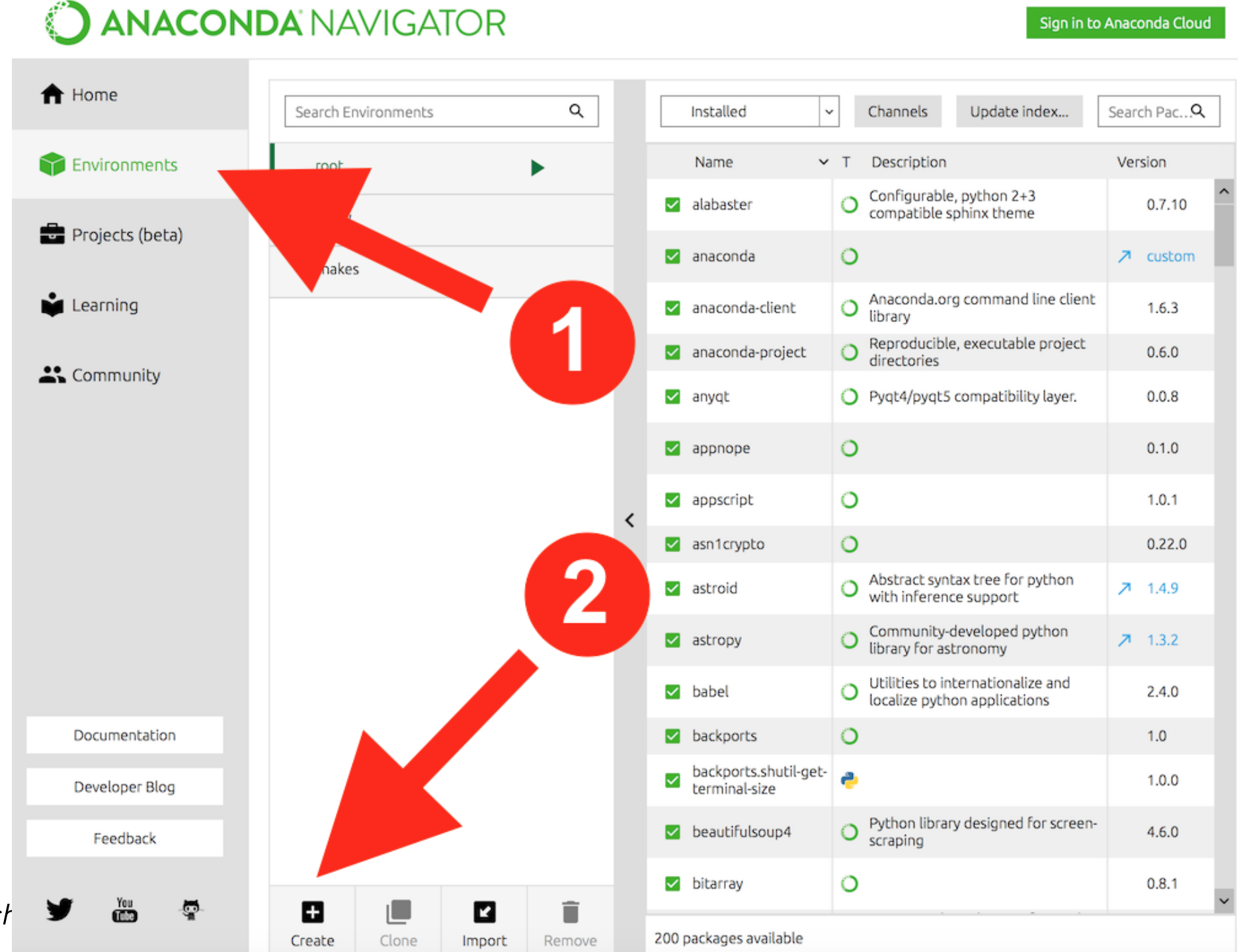
pixi

Devbox

Conda Hands-on Session



https://siscourses.ethz.ch/reproducible_computing/Conda.slidy.html


ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Projects (beta)
Learning
Community
Documentation
Developer Blog
Feedback
Search Environments
root
nakes
Create
Clone
Import
Remove
Installed
Channels
Update index...
Search Pac...
Name
T
Description
Version
alabaster
Configurable, python 2+3 compatible sphinx theme
0.7.10
anaconda
custom
anaconda-client
Anaconda.org command line client library
1.6.3
anaconda-project
Reproducible, executable project directories
0.6.0
anyqt
Pyqt4/pyqt5 compatibility layer.
0.0.8
appnope
0.1.0
appscript
1.0.1
asn1crypto
0.22.0
astroid
Abstract syntax tree for python with inference support
1.4.9
astropy
Community-developed python library for astronomy
1.3.2
babel
Utilities to internationalize and localize python applications
2.4.0
backports
1.0
backports.shutil-get-terminal-size
1.0.0
beautifulsoup4
Python library designed for screen-scraping
4.6.0
bitarray
0.8.1
200 packages available
1
2

Conda - What can go wrong?

- The package metadata (dependency list) is updated (not very likely)
- The package is deleted by the owner
- The package is not available under another platform
- There is no conda package for what you are looking for
- Complex dependencies may fail or take a long time to resolve

Virtualizing Computing Environments

Conda - What can go wrong?

- The package metadata (dependency list) is updated (not very likely)
- **The package is deleted by the owner**
- **The package is not available under another platform**
- **There is no conda package for what you are looking for**
- Complex dependencies may fail or take a long time to resolve

Reproducible Environment

Problem:

Full reproducibility requires the possibility to recreate the system that was originally used to generate the results

Solution:

Bundle your application and all dependencies

→ Environment Isolation & Dependency management

Tools:

- Application / software level: Conda, pip, virtualenv, renv
- Containerization: Docker
- Virtualization (Virtual Machine, VM): VirtualBox, VMware, Parallels

Reproducible Environment – Virtual Machines

- A virtual machine (VM) is an operating system (“guest”) that runs inside another computing environment (“host”).

- **Advantages:**
  - Allows multiple OS environments on a single physical computer
  - VMs are widely available and are easy to manage, maintain and distribute
  - Offers application provisioning and disaster recovery options

- **Drawbacks:**
  - They are not as efficient as a physical computer because the hardware resources are distributed in an indirect way.
  - Multiple VMs running on a single physical machine can deliver unstable performance

Source: https://searchservervirtualization.techtarget.com/definition/virtual-machine

Reproducible Environment – Containers

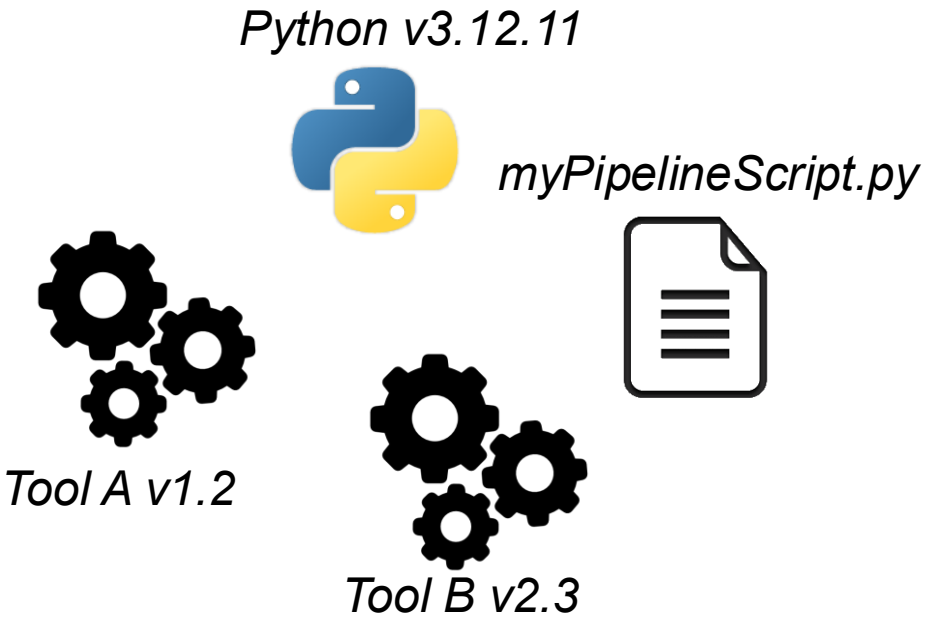
- A mini, more flexible, virtual machine
- Great for projects that require specific system requirements / applications / tools / OS
- Larger than a virtual environment but even better for long-term reproducibility
- Docker / Podman / Colima / Apptainer are programs that help you create, run and deliver containers

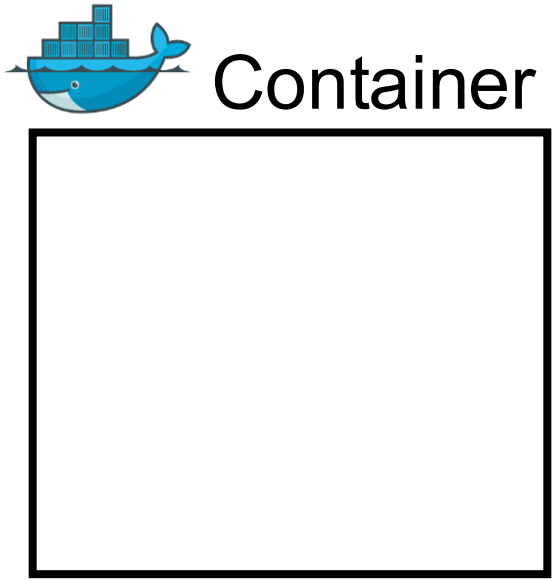
podman

Colima

Reproducible Environment – Containers

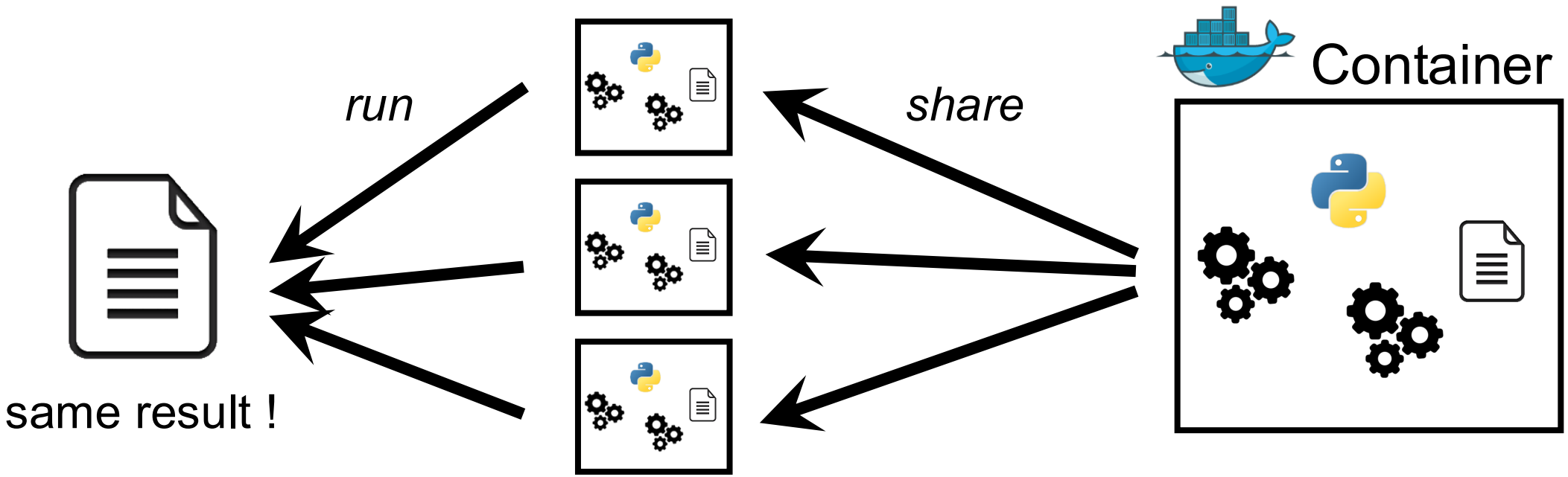
- **Containers** allow you to **package** your software / pipeline with the **dependencies** inside a **reproducible**, easy to **share**, **runnable** file
- Example: **Docker containers**





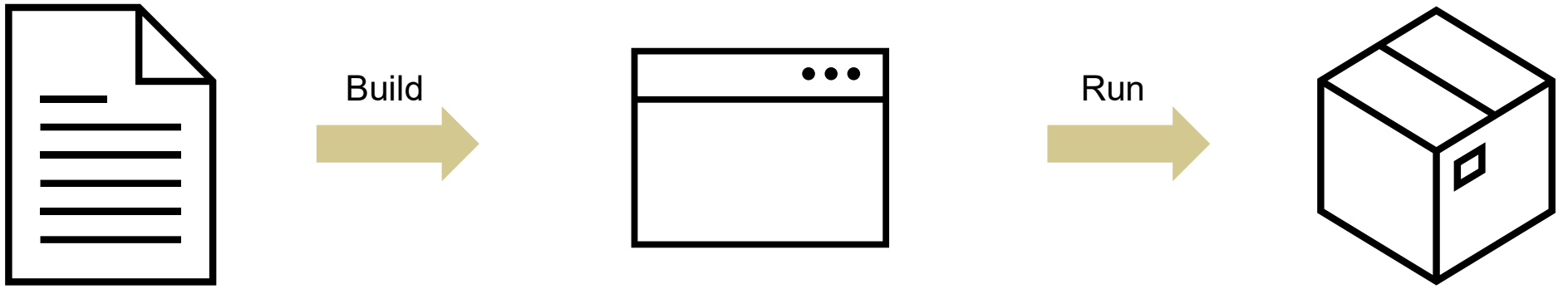
Reproducible Environment – Containers

- **Containers** allow you to **package** your software / pipeline with the **dependencies** inside a **reproducible**, easy to **share**, **runnable** file
- Example: **Docker containers**



Reproducible Environment – Containers

Pull back the curtain on containerization



Dockerfile

Docker image

Docker container

Small text file with instructions

Packaged snapshot of everything
(OS, packages, code) ready to be used

A running instance of your image,
Package of software

*an instance because you can run
multiple containers from one image

Slide credit: E. Knappe (Lib4RI)

Reproducible Environment – Containers

Pull back the curtain on containerization

- Write or use an existing Dockerfile

```
FROM python:3

WORKDIR /usr/src/app

COPY requirements.txt ./
RUN pip install --no-cache-dir -r requirements.txt

COPY . .

CMD [ "python", ".scripts/wind_speed_at_dorthys_house.py" ]
```


Slide credit: E. Knappe (Lib4RI)

Reproducible Environment – Containers

Pull back the curtain on containerization

- Write or use an existing Dockerfile

Dockerfile for pflotran

https://groundwater.usu.edu/

Slide credit: E. Knappe (Lib4RI)

Reproducible Environment – Containers

Pull back the curtain on containerization

- Write or use an existing Dockerfile
- Build the image
  - Or use an existing one: https://hub.docker.com/search?badges=official

```
docker build -t tornado_analysis
```


Slide credit: E. Knappe (Lib4RI)

Reproducible Environment – Containers

Pull back the curtain on containerization

- Write or use an existing Dockerfile
- Build the image
  - Or use an existing one: https://hub.docker.com/search?badges=official
- Run the container

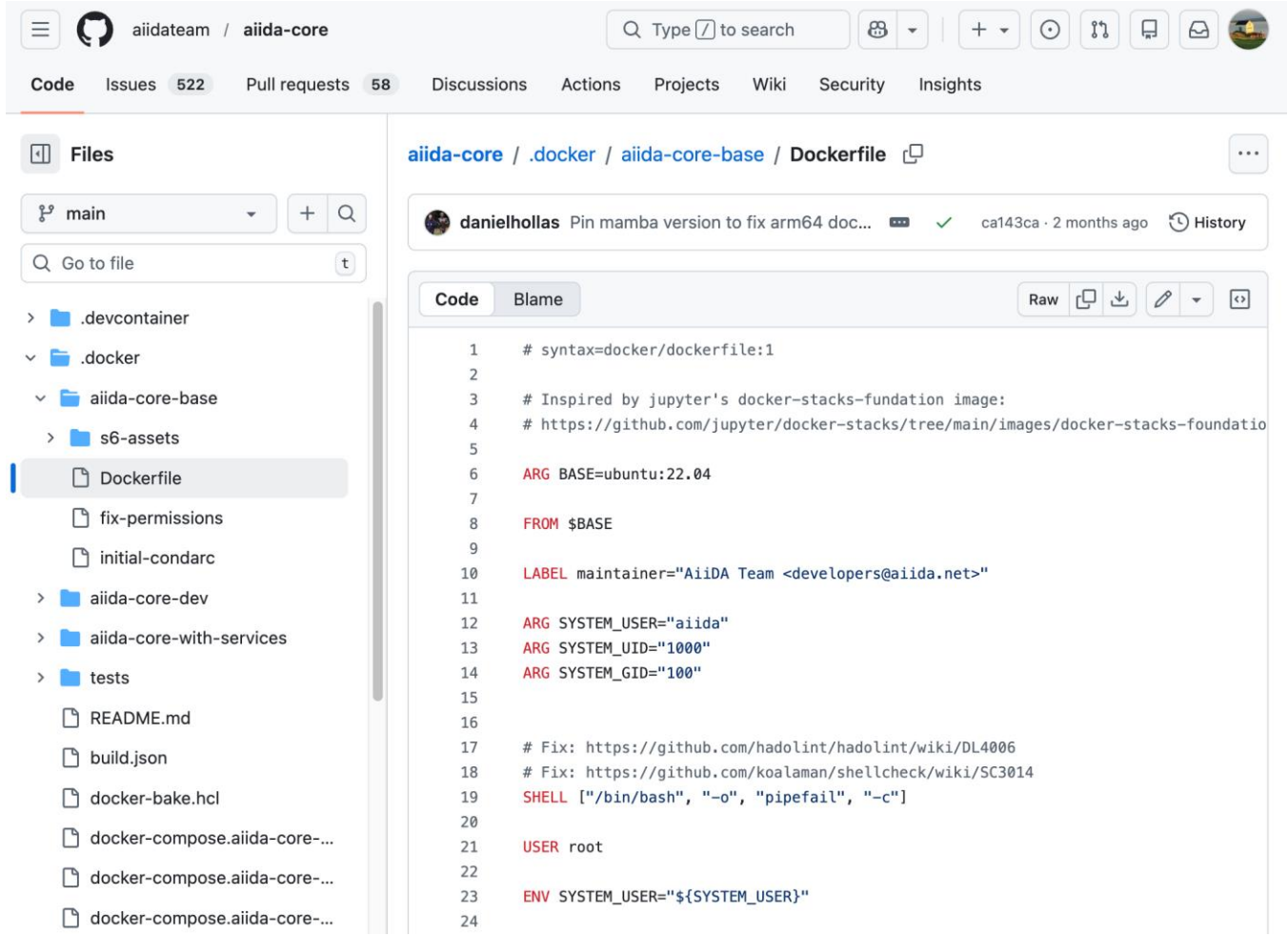
```
docker run tornado_analysis
```


Slide credit: E. Knappe (Lib4RI)

Reproducible Environment – Containers

Pull back the curtain on containerization

- Write or use an existing Dockerfile
- Build the image
  - Or use an existing one: https://hub.docker.com/search?badges=official
- Run the container
- Share your Dockerfile when you share your code

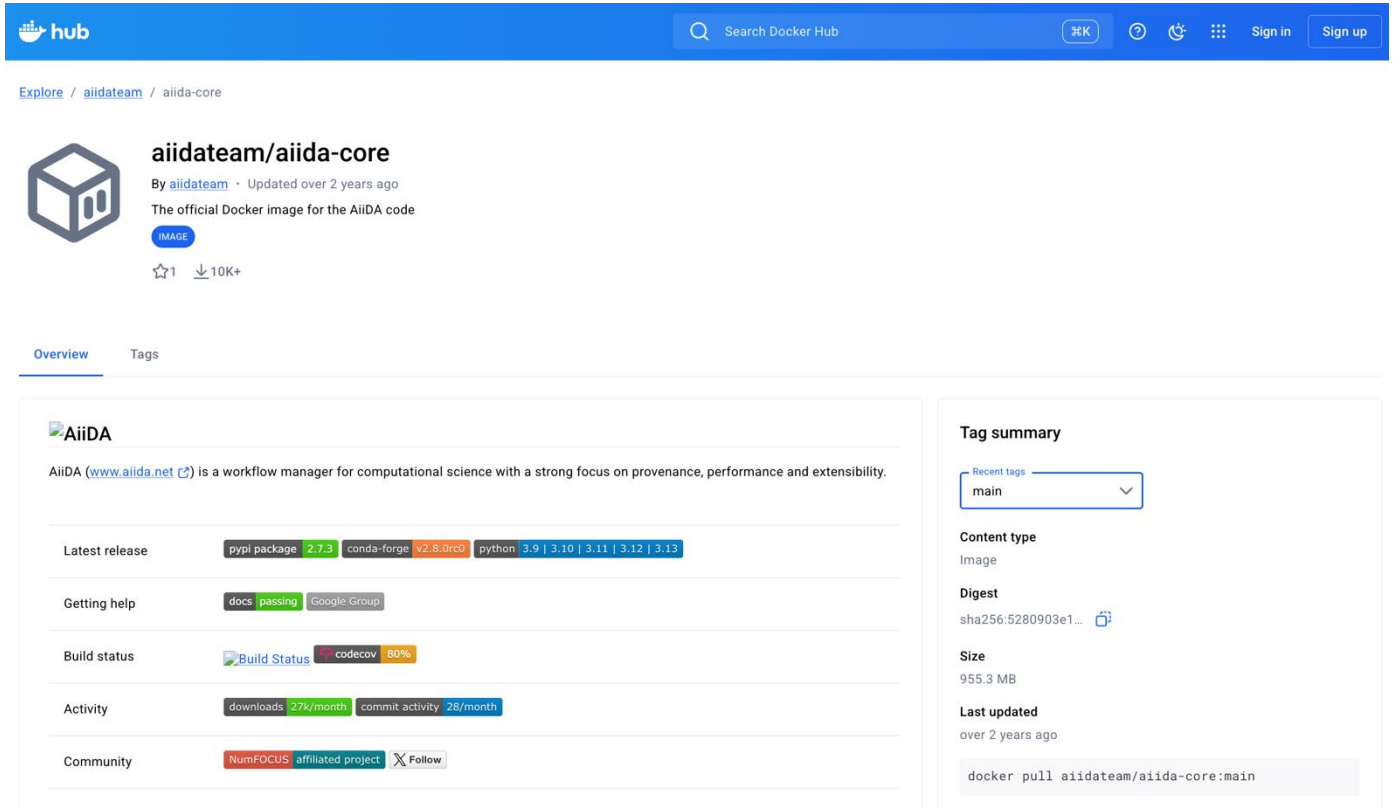


Dockerfile for AiiDA workflow manager on Github

Reproducible Environment – Containers

Pull back the curtain on containerization

- Write or use an existing Dockerfile
- Build the image
  - Or use an existing one: https://hub.docker.com/search?badges=official
- Run the container
- Share your Dockerfile when you share your code
- Archive your Docker **image** for long-term preservation



Docker images for AiiDA workflow manager on Docker Hub

Reproducible Environment – Containers

Difference between Dockerfile, Image and Container

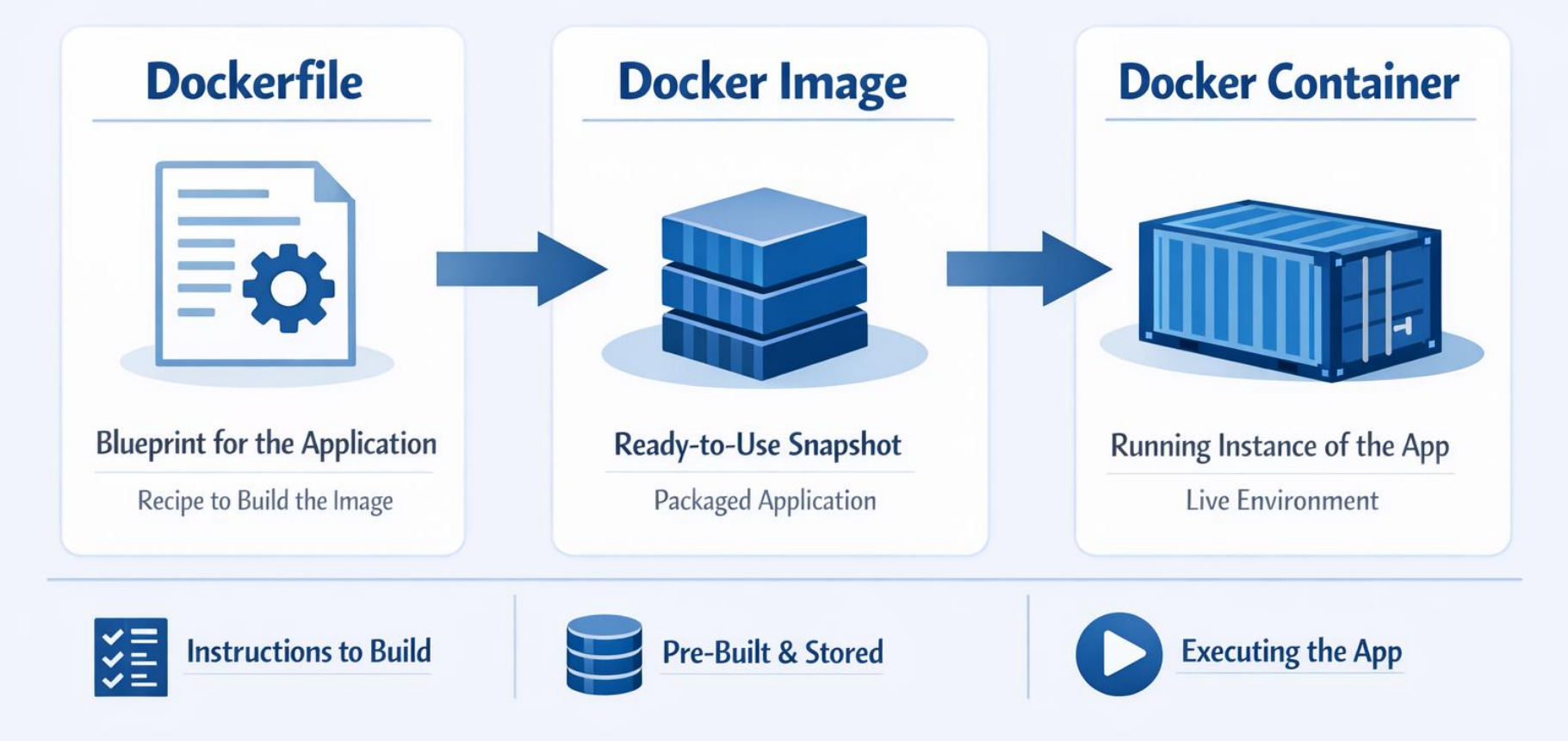
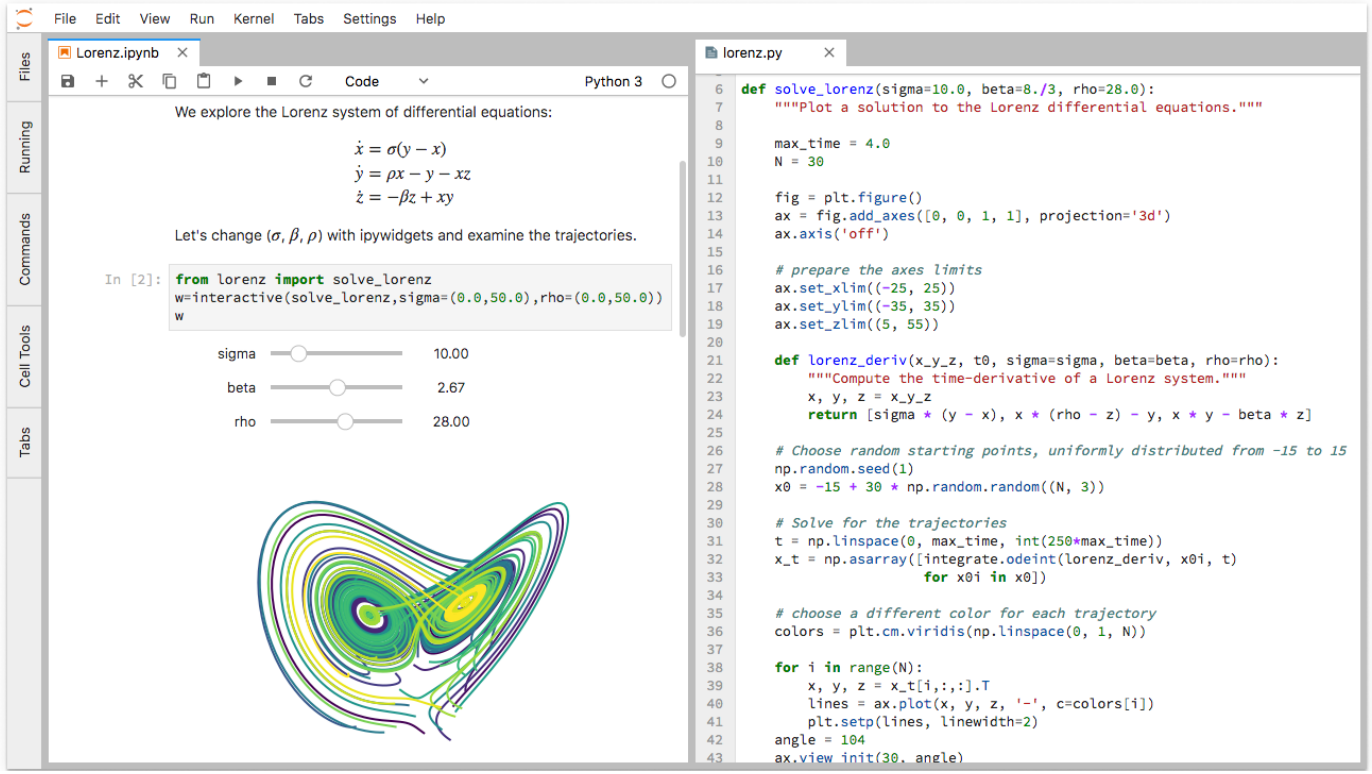


Image generated with DALL·E (OpenAI), March 2026.

Virtual Machines vs Containers

	VMs (Virtual Box)	Containers (Docker)
Use case	Complex Apps (GUI, …)	Data Analysis Scripts, Simple Apps, Microservices, Continuous Integration
Virtualization	Hardware-level	OS-level
Size	GB	MB
Startup time	Minutes	Seconds
Guest OS	Windows, macOS, Linux	Primarily Linux-based
Host OS	Windows, macOS, Linux	Linux, Windows 10 / macOS with hypervisor
Overhead (RAM, CPU)	High - reduced performance	Low - close to native performance
Security	Better (fully isolated)	Poorer (shared kernel)
How to use	Easy if you know to install OS	New things to learn
Getting started	www.virtualbox.org/manual/ch01.html	https://docs.docker.com/get-started/

Reproducible computational environment: Questions?



jupyter

MATLAB Live Editor

RStudio®

WolframAlpha NOTEBOOK EDITION™

Interactive Computational Notebooks

Interactive Notebooks

- Applications that combine documentation, code, input and output generated by the code, e.g. graphs, plots (*Nature 515, 151–152*)
- Useful for exploratory data analysis, sharing, and reproducibility

RStudio

- Open source + commercial edition
- Mainly for development in R but other languages supported

jupyter

- Open source
- > 40 languages supported (Python, R, Julia, Matlab, IDL, etc.)

WolframAlpha NOTEBOOK EDITION™

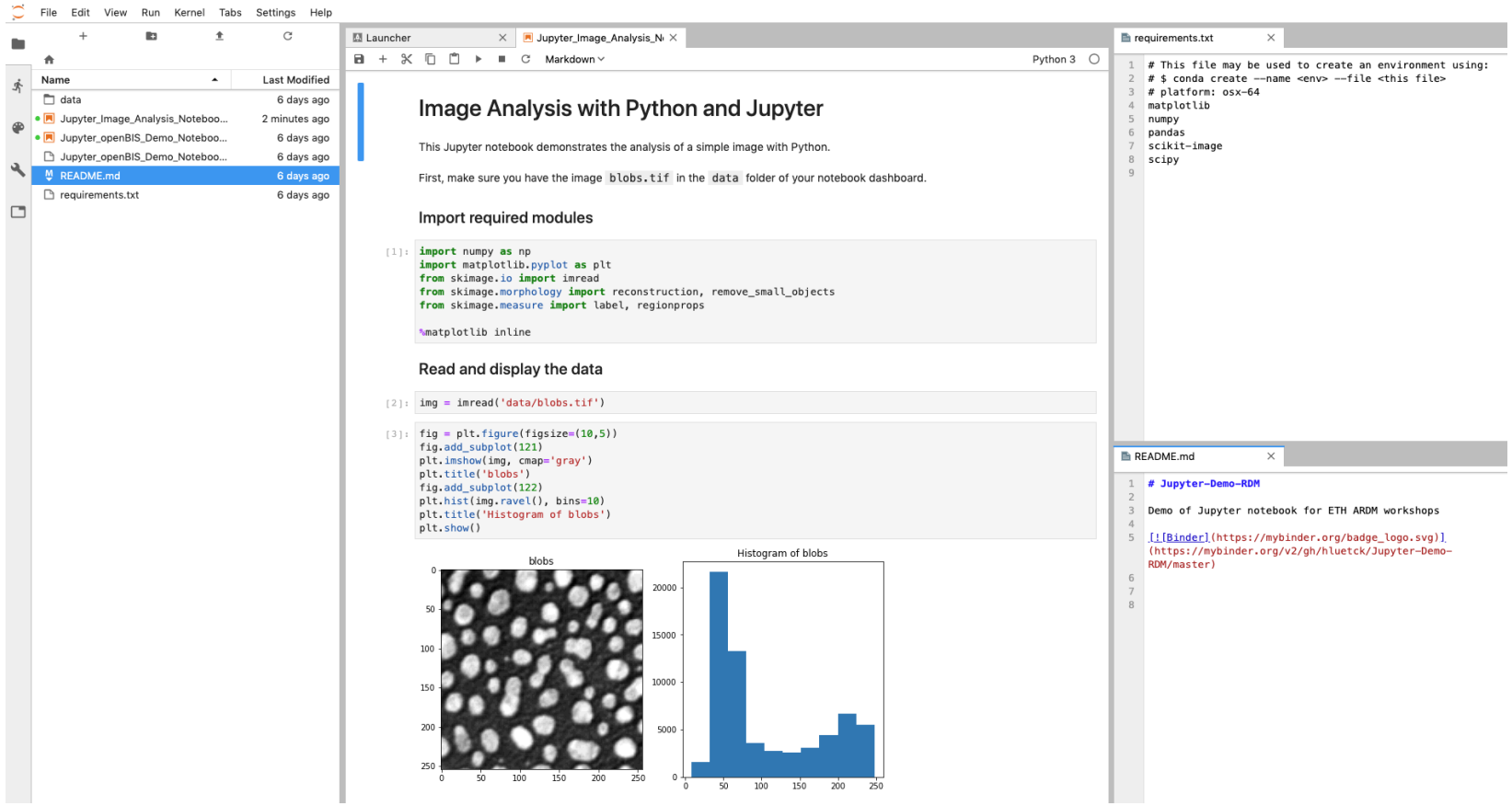
- Commercial
- Used in mathematical fields

MATLAB

- Commercial
- Used in scientific, engineering, mathematical fields

Interactive Notebooks: Jupyter

- **Jupyter notebook:** web-based interactive computational environment
- **JupyterLab:** web-based interactive development environment for notebooks, code, and data



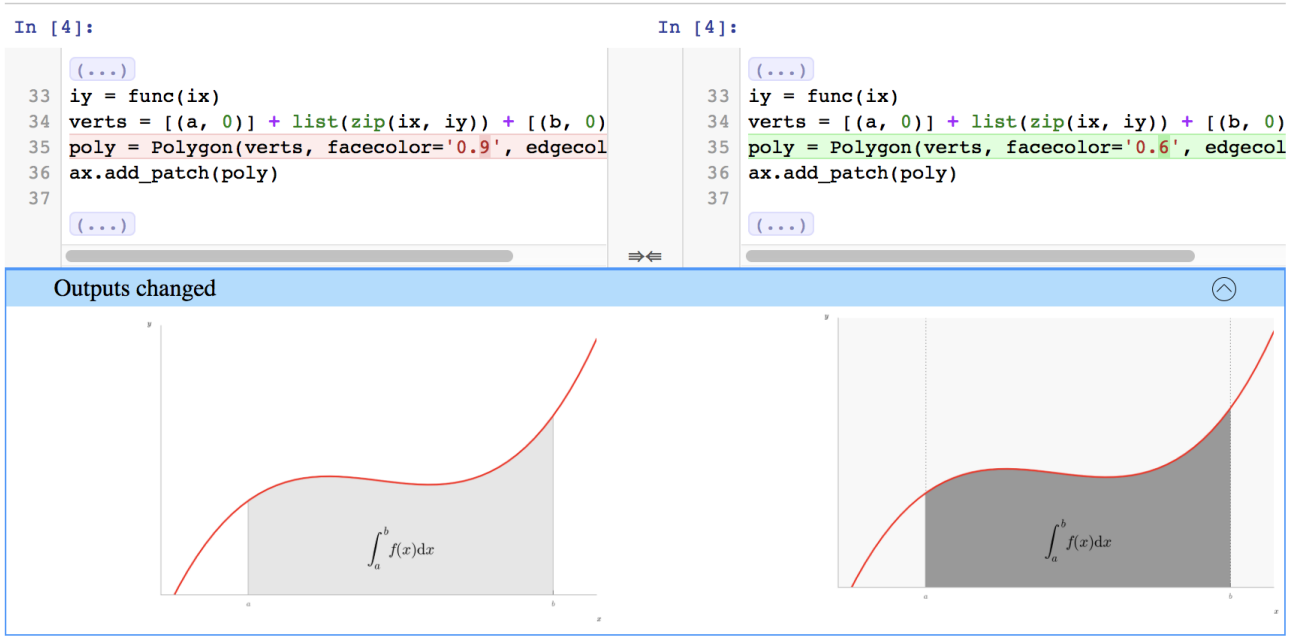
Interactive Notebooks – what can go wrong?

- **Versioning**
  - Version control of even moderately complex NBs is challenging
  - Tracking NB history is harder than for traditional source code
  - Some tools may help (e.g. *nbdime*, *Jupytext*, nbstripout, nbconvert)

```
$ diff a.ipynb b.ipynb
76,77d75
<     "plt.rc('axes', grid=False)\n",
<     "plt.rc('axes', facecolor='white')\n",
90c88
<       "image/png": "iVBORw0KGgoAAAANSUhEUgAABLkAAAMQCAYAAADLj7dlAAAABHNCSVQICAgIfAhki
AAAAAlwSFlz\nAAAWJQAAFiUBSVIk8AAAIABJREFUeJzsvXeYZFd57b12h0maPNJII2lGOaCAkEBCFgozIxkBAp
lY\n1waDyDZg8MX+zMU2F4Mx1x8PwWAwxmBjg4yNi2BfQMa20iiAQFkIjXKWRtJIE3tSz3TXuX+8vV2n\nqyucv
N+9z/o9zzynprvq1D6nqqtqr1prbRNFEQghhBBCCCGEEEII8Zkh1wMghBBCCCGEEEIIISQv\nFLkIIYQQQgghhB
BCiPdQ5CKEEEIIIYQQQggh3kORixBCCCGEEEIIIYR4D0UuQgghBBCCCGEEOI9\nFLkIIYQQQgghBBCiPdQ5CK
EEEIIIYQQQggh3kORixBCCCGEEEIIIYR4D0UuQgghBBCCCGEEOI9\nFLkIIYQQQgghBBCiPdQ5CKEEEIIIYQQ
Qggh3kORixBCCCGEEEIIIYR4D0UuQgghBBCCCGEEOI9\nFLkIIYQQQjzEGHOJMaZljPmo67EkZWq8D7keByGEE
ELChCIXIYQQQirDGPOmKaFj3BhzkMNx/H/G\nmG3GmP/pagwFEbkeQJUYY75gjNljmD67EQQghRB8UuQghhB
BSJe+DCDMjAH7L4TjeAmA+gLc5\nHEMRGNcDqJi3AVgI4DddD4QQghJBKMMacCuBMAFsg4sy7jTH
DjobzZwBuBvBxR/dP\nsvERADcC+LTrgRBCCCFEHxS5CCGEEFIVH4C4uP4SIlQcBOD1LgYSRVEziqXR1H0fRf3
T7IRRdFf\nRlH0K1EUXe96LIQQQgjRB0UuQgghhJSOMWYpgP8BoAXg7wH8HcTN9Tsux0nCwbaLBI
```

Interactive Notebooks – what can go wrong?

- **Versioning**
  - Version control of even moderately complex NBs is challenging
  - Tracking NB history is harder than for traditional source code
  - Some tools may help (e.g. *nbdime*, *Jupytext*, nbstripout, nbconvert)



Chattopadhyay et al. *(2020). What's Wrong with Computational Notebooks?* *doi:10.1145/3313831.3376729*

Interactive Notebooks – what can go wrong?

- **Versioning**
  - Version control of even moderately complex NBs is challenging
  - Tracking NB history is harder than for traditional source code, especially with “classical” git
  - Some jupyter-targeted tools may help

- **Reproducibility**
  - Interactive working mode can result in hard-to-reproduce notebooks
  - Discipline is needed! Regular pruning & refactoring; *“Restart kernel & Run all”* is your friend

- **Security**
  - Data confidentiality & access controls may be problematic

Chattopadhyay et al. *(2020). What's Wrong with Computational Notebooks?* doi:10.1145/3313831.3376729

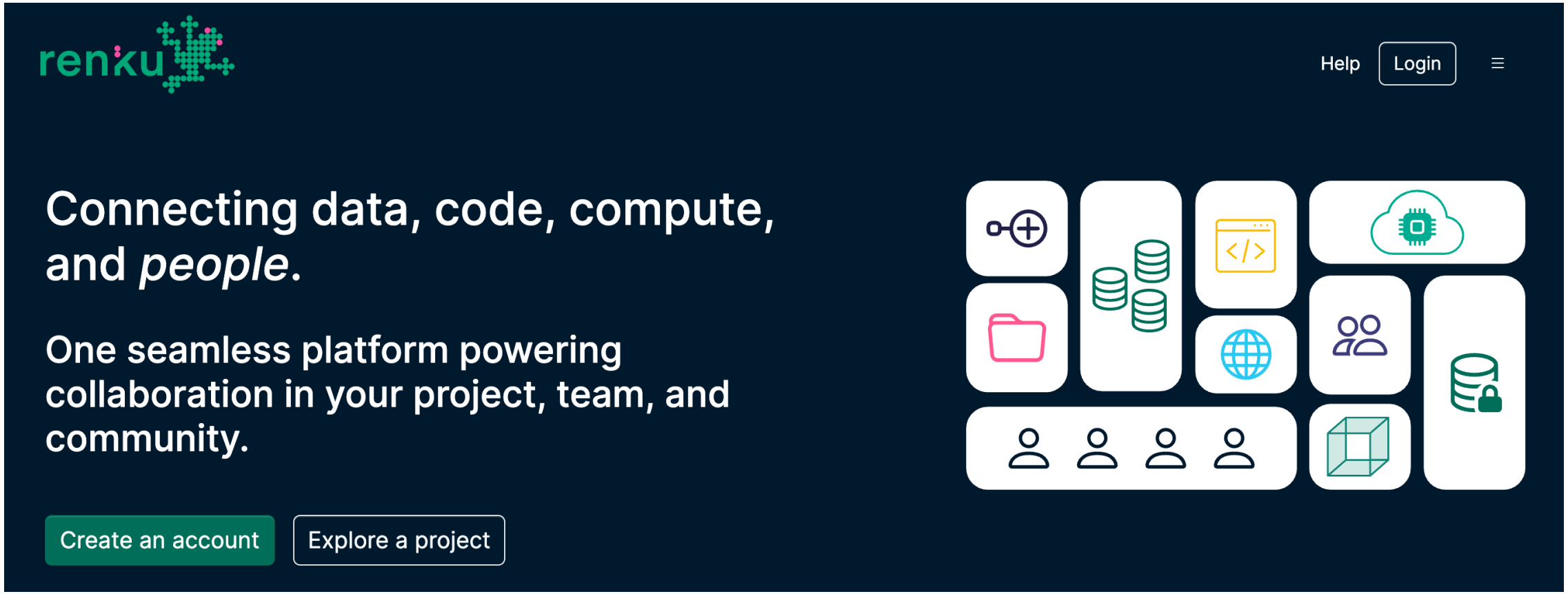
Reproducible Computing Platforms

Reproducible Computing Platforms

- Integrated, **web-based** solutions for **reproducible** and **collaborative** data analysis and **computing**
- Usually built upon **proven open-source technologies** (Git, Conda, Docker etc.)
- Technical **complexity hidden** from user (or made easily accessible)
- Platforms provide **low entry barrier** access to fully reproducible computing
- **Commercial platforms**
  - Examples: Code Ocean, Google Colaboratory, …
  - Costs are incurred by usage of underlying cloud infrastructure (storage, compute, data transfer!)
  - Beware of data ownership, licensing issues and general T&Cs
- **Community platforms**
  - Examples: mybinder, Renkulab.io
  - Usually free of charge but resources are limited

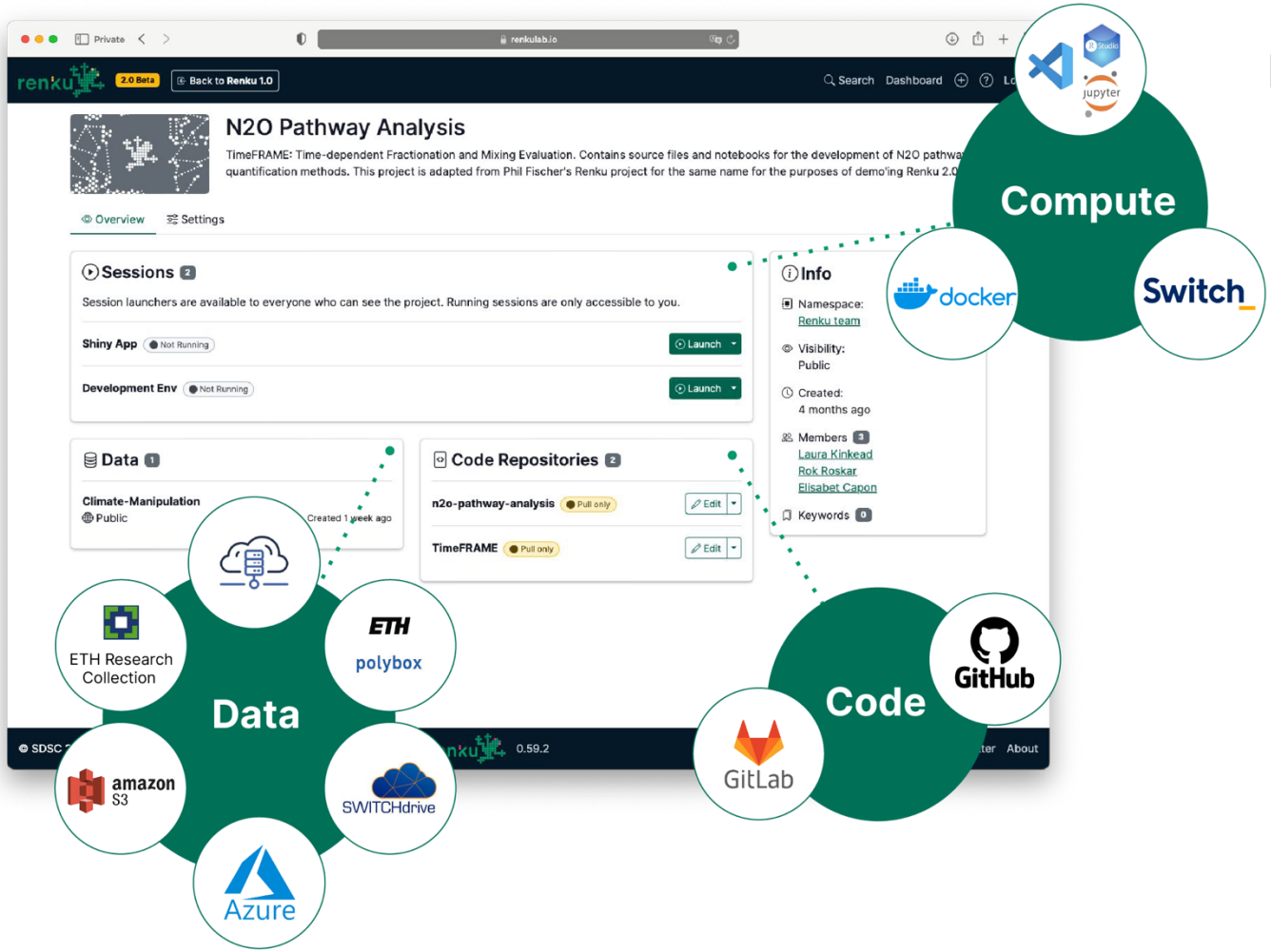
Reproducible Computing Platforms: *renkulab.io*

- Renkulab is a **platform for collaborative data science** from the Swiss Data Science Center (SDSC)



Reproducible Computing Platforms: *renkulab.io*

- Renkulab is a **platform for collaborative data science** from the Swiss Data Science Center (SDSC)

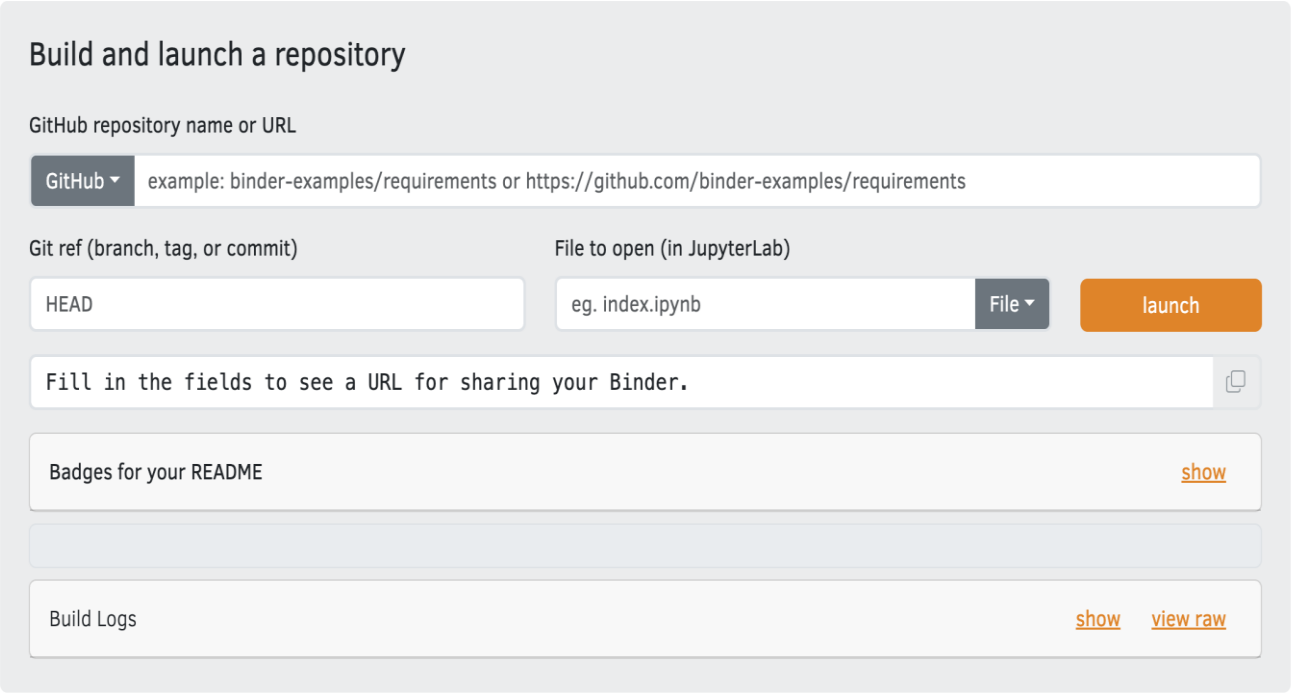


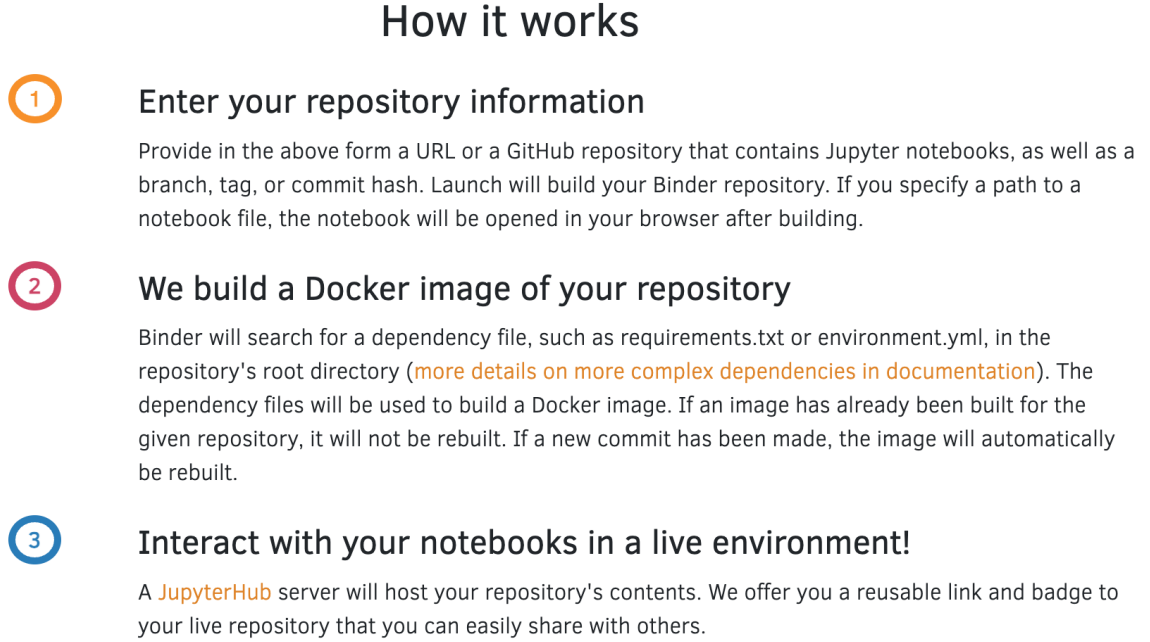
Learning more about Renkulab

- Login with your Switch edu-ID (or create a new account)
- Getting Started Tutorial
- More Renku Tutorials
- Renku How-To Guides

Reproducible Computing Platforms: *mybinder.org*

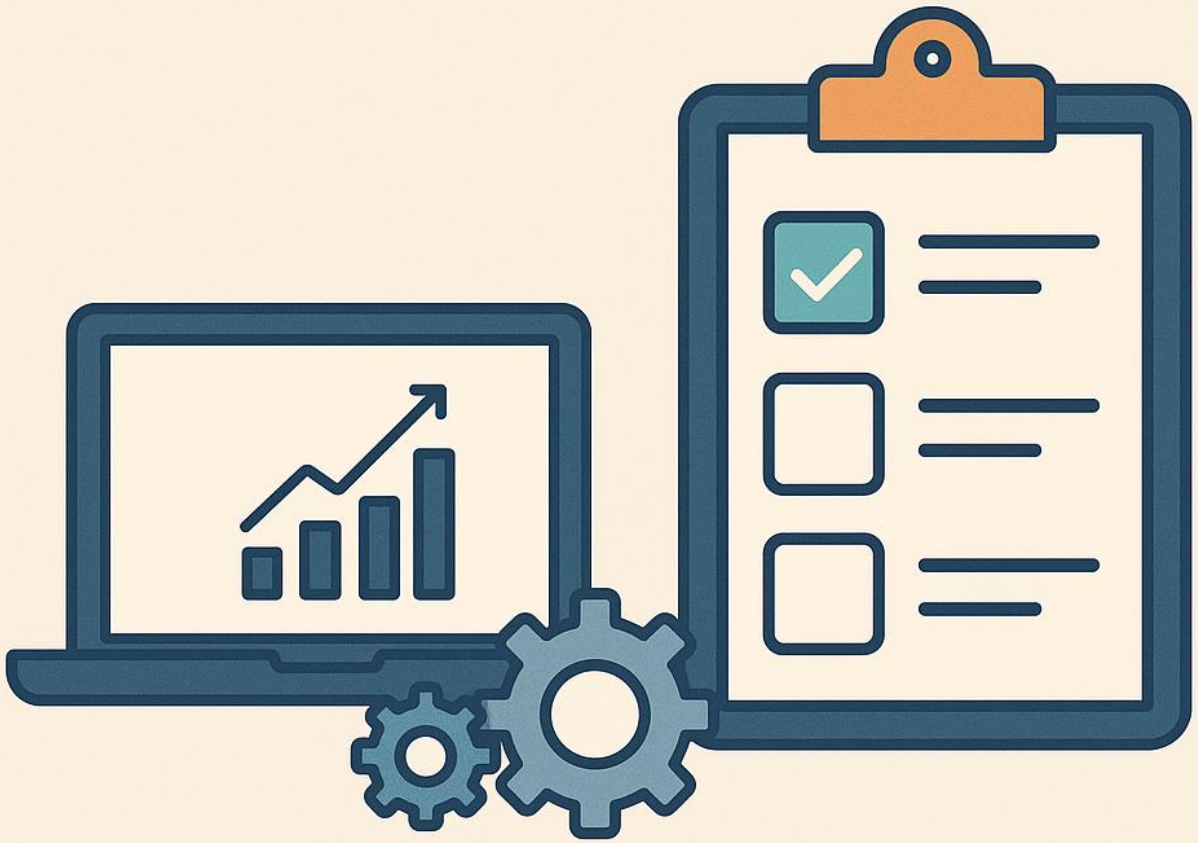
- Binder converts a Git repository into a collection of interactive notebooks
- Notebooks open in an executable environment → code becomes reproducible by anybody, anywhere



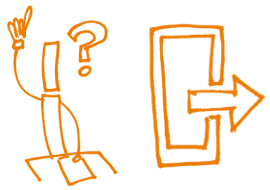


- Live example: https://github.com/hluetck/mybinder-demo-project

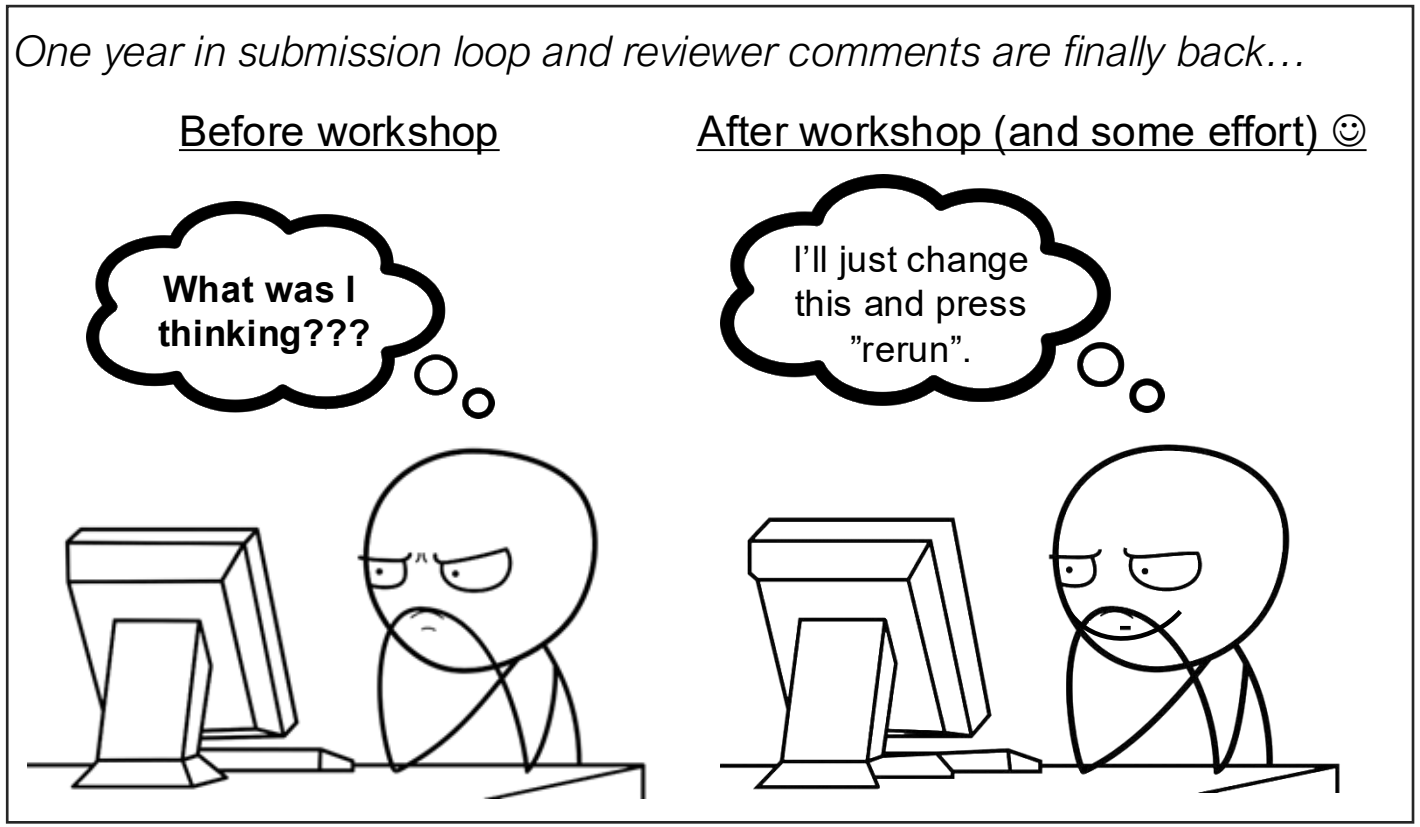
QUIZ TIME ...



Wrap-up & Discussion



What's in it for me?



At the start of the project

- Forced to think about scope and limitations
- Improved structure and organization

During the project

- Easier to rerun experiments and analysis
- Closer interaction between collaborators
- Much of the manuscript "writes itself"

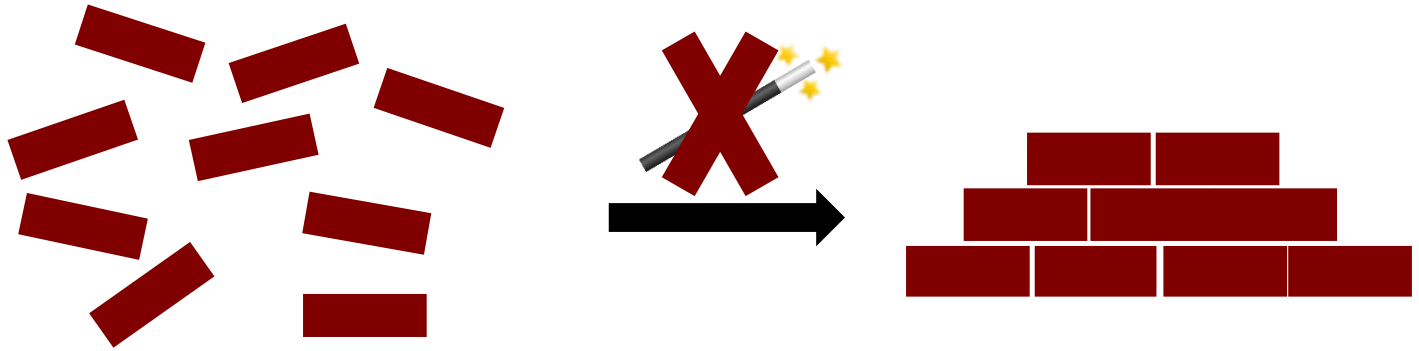
After the end of the project

- Faster resumption of research by others (or your future self), thereby increasing the impact of your work
- Increased visibility in the scientific community

Slide credit: L. Wigge, R. Ågren & J. Sundh (NBIS & SciLifeLab, Sweden)

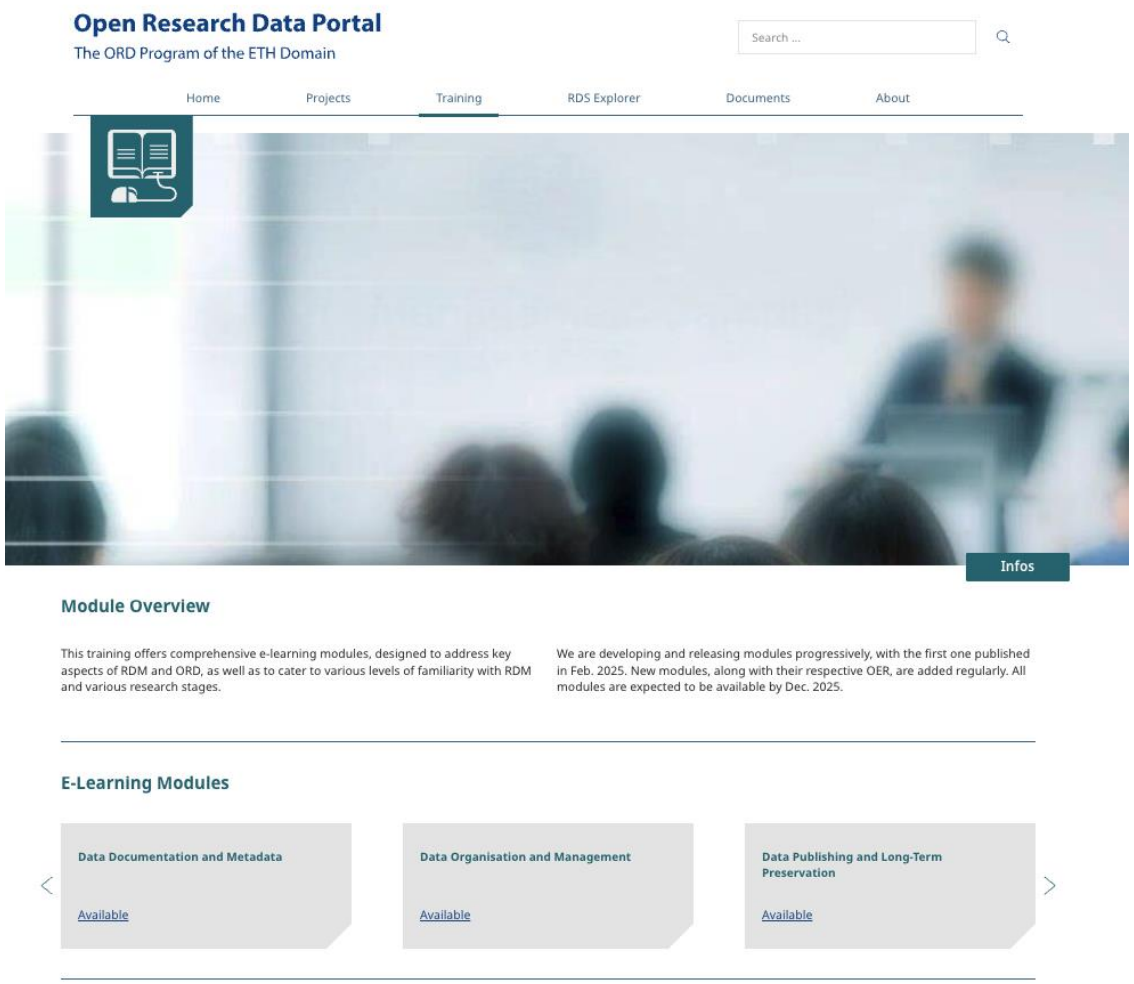
What's in it for me?

- Aim for improvement, not perfection!
- RDM requires **WORK** & **TIME**, but the time spent on this is an **investment** for the future!



Contact us for consultations / trainings on data management, version control, reproducible computational workflows or data science support

sis.helpdesk@ethz.ch



https://open-research-data-portal.ch/training/

Contacts

Nadia Marounina

nadejda.marounina@id.ethz.ch

Henry Lütcke

henry.lutcke@id.ethz.ch

https://siscourses.ethz.ch/reproducible_computing/

sis.helpdesk@ethz.ch

https://sis.id.ethz.ch/

Feedback: https://www.umfrageonline.ch/c/scientificcomputing

Any final questions on what we have discussed this morning?

https://siscourses.ethz.ch/reproducible_computing/

Feedback: https://www.umfrageonline.ch/c/scientificcomputing