

Reproducible Scientific Computing and Data Analysis

Nadia Marounina, Henry Lütcke
Scientific IT Services, ETH Zurich

March 22, 2023

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



Overview of today's workshop



Setting the Scene



Managing your Source Code



Managing Dependencies & Computing Environments



Virtualizing Computing Environments



Interactive Computational Notebooks

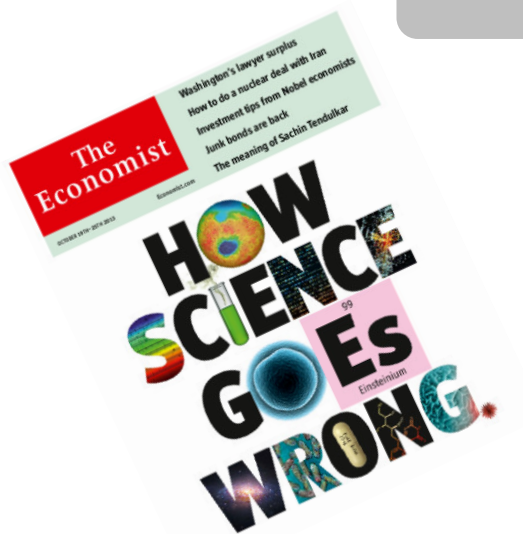


Reproducible Computing Platforms



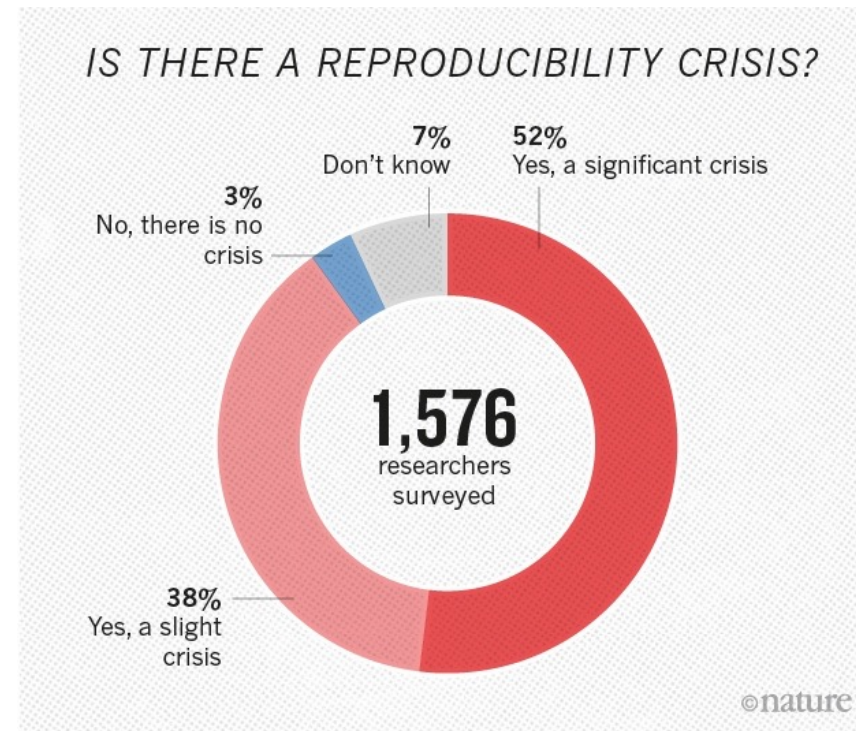
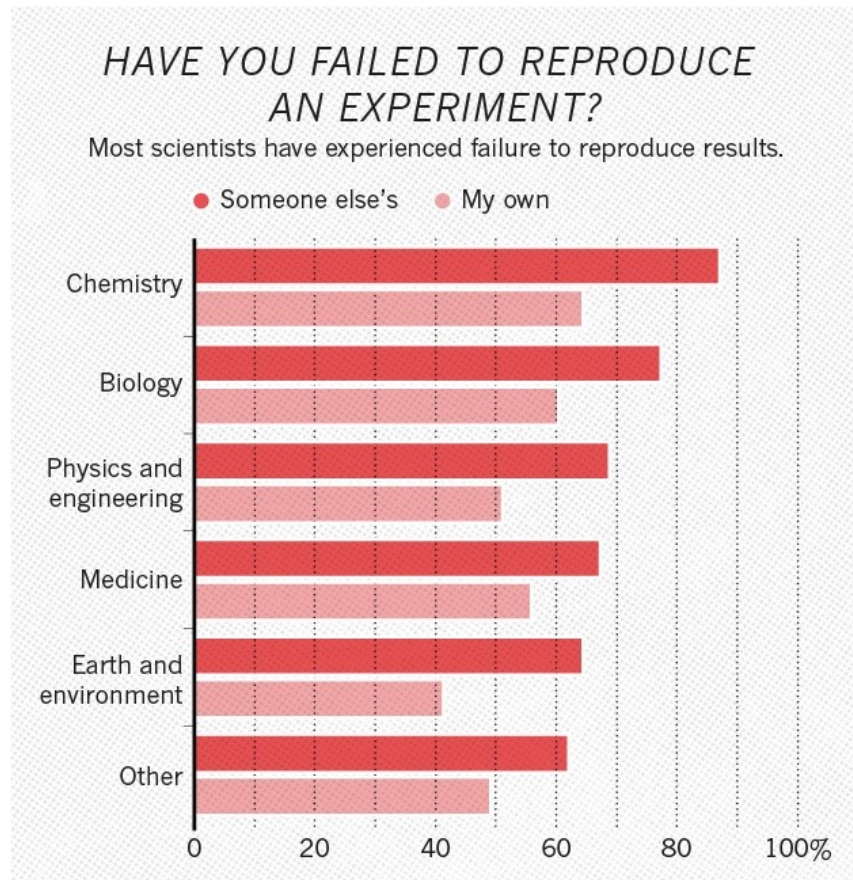


Setting the Scene



Reproducibility & Replicability in Science

Nature survey on reproducibility across all scientific domains



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

Reproducibility & Replicability in Science

RESEARCH ARTICLE

Estimating the reproducibility of psychological science

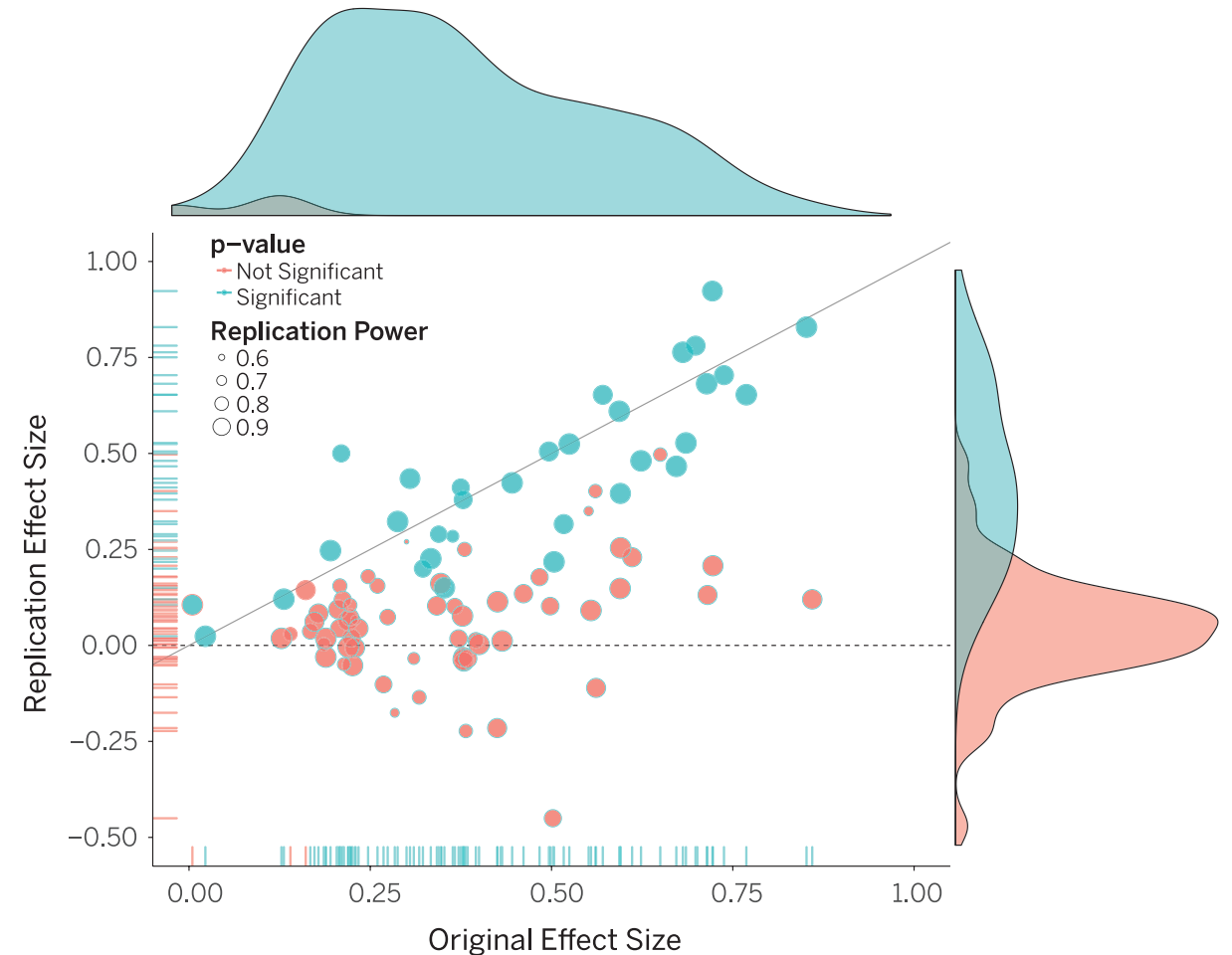
Open Science Collaboration^{*,†}

† 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**



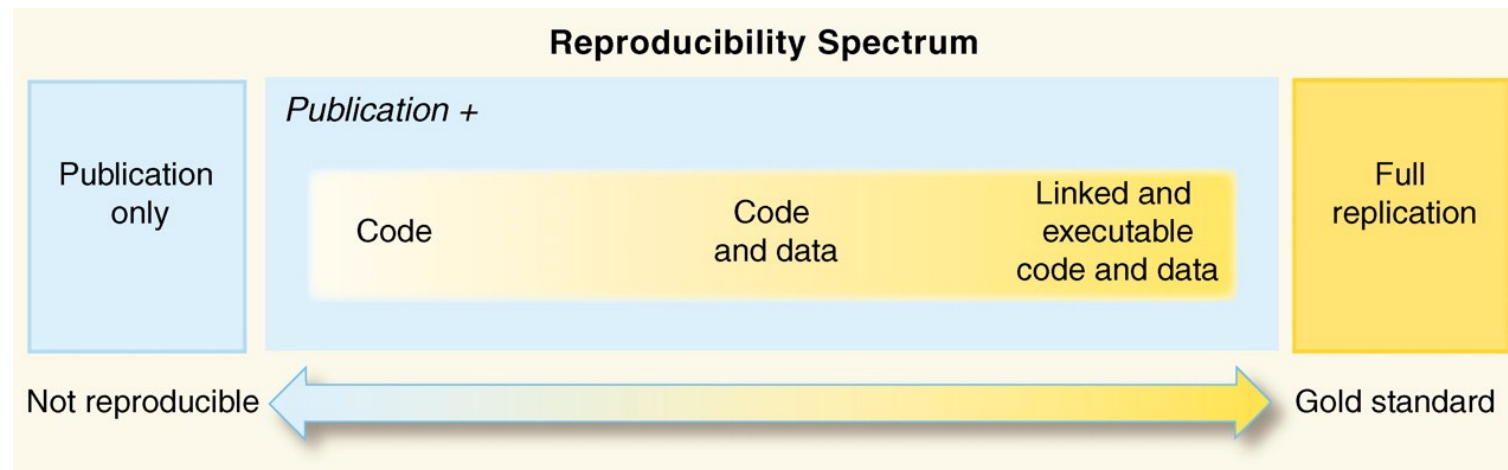
Reproducibility & Replicability in Science

Replication:

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

Reproducible research:

same data + same method = same results



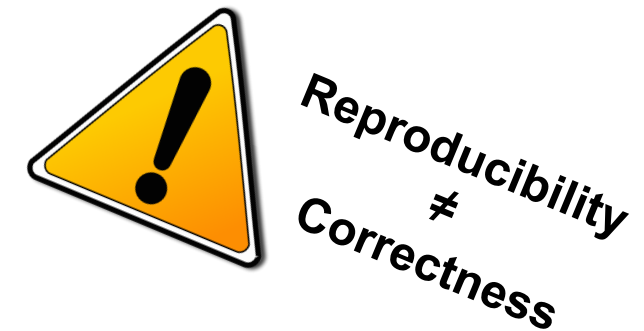
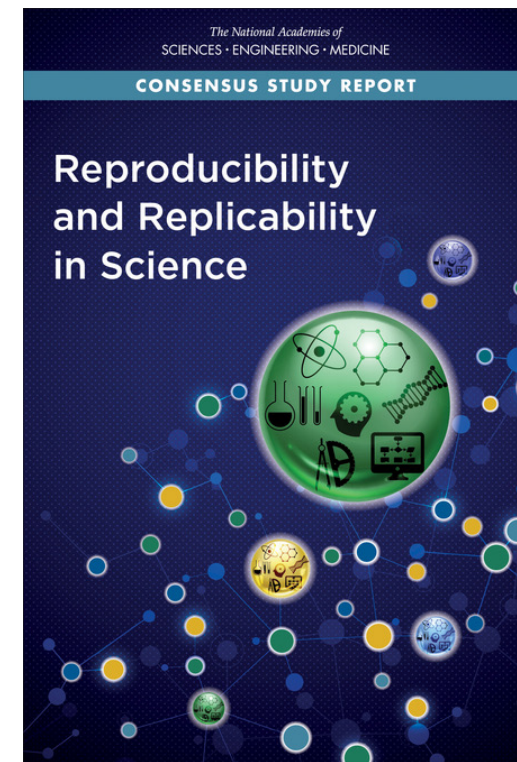
Peng (2011). [doi:10.1126/science.1213847](https://doi.org/10.1126/science.1213847)

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>



Computational Reproducibility: What can go wrong?

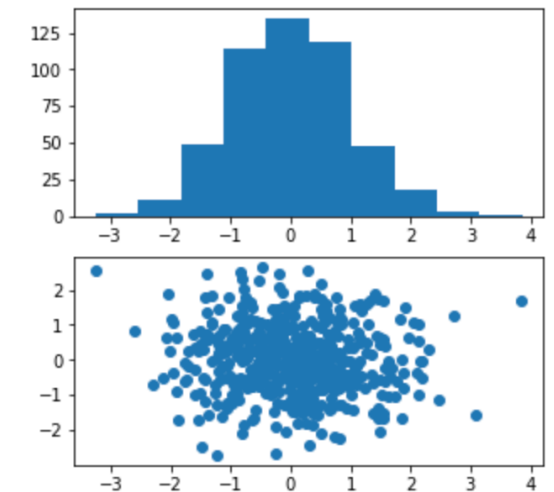
- Code only runs on specific **operating system**
 - Examples: Windows / Linux scripts, special programs (e.g. *SigmaPlot*)
- Code has specific **external dependencies**
 - Example: wget <https://zenodo.org/record/1234567/files/dataset.zip>
- Code has specific **internal dependencies** (libraries, modules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

```
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__)

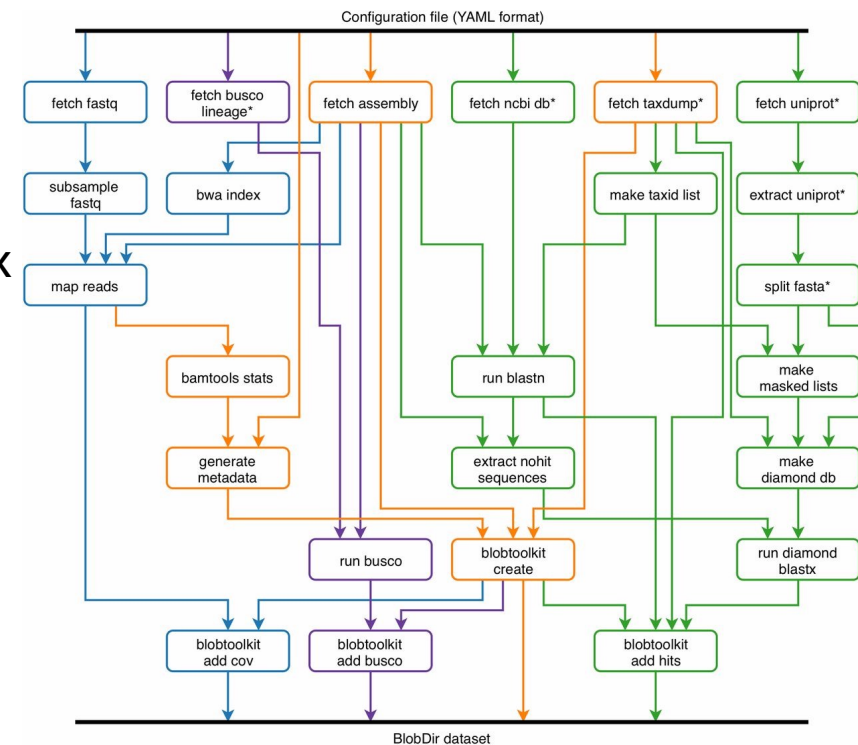
rng = np.random.default_rng(42)
rng.dirichlet((0.04, 0.03), 2)

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>

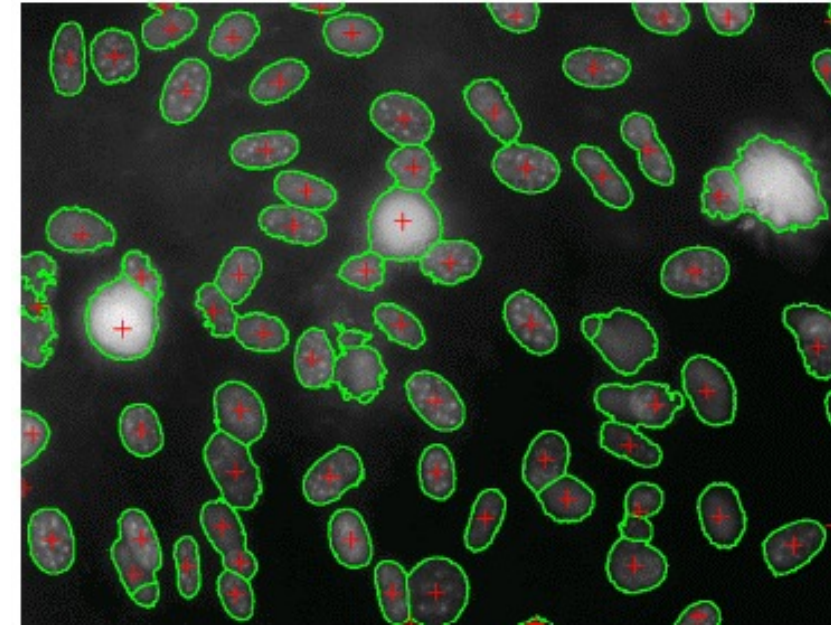
Computational Reproducibility: What can go wrong?

- 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)



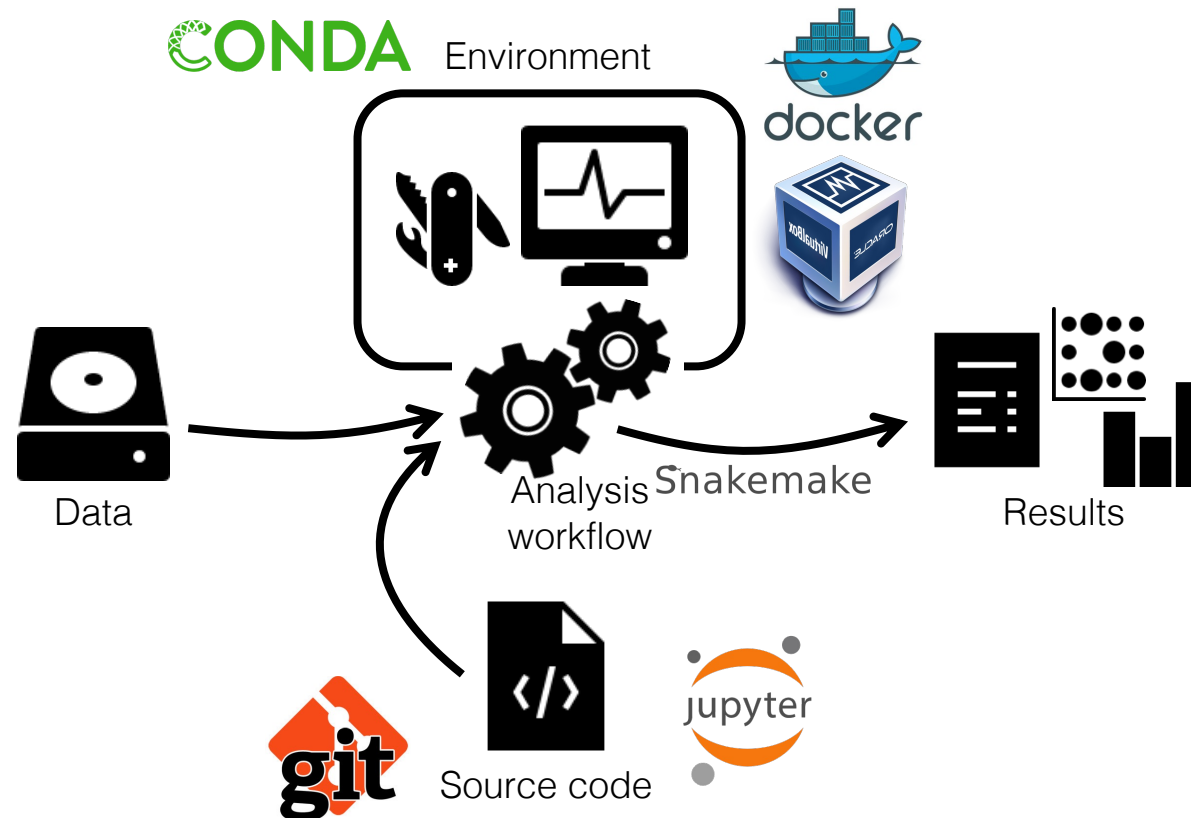
Computational Reproducibility: What can go wrong?

- 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**



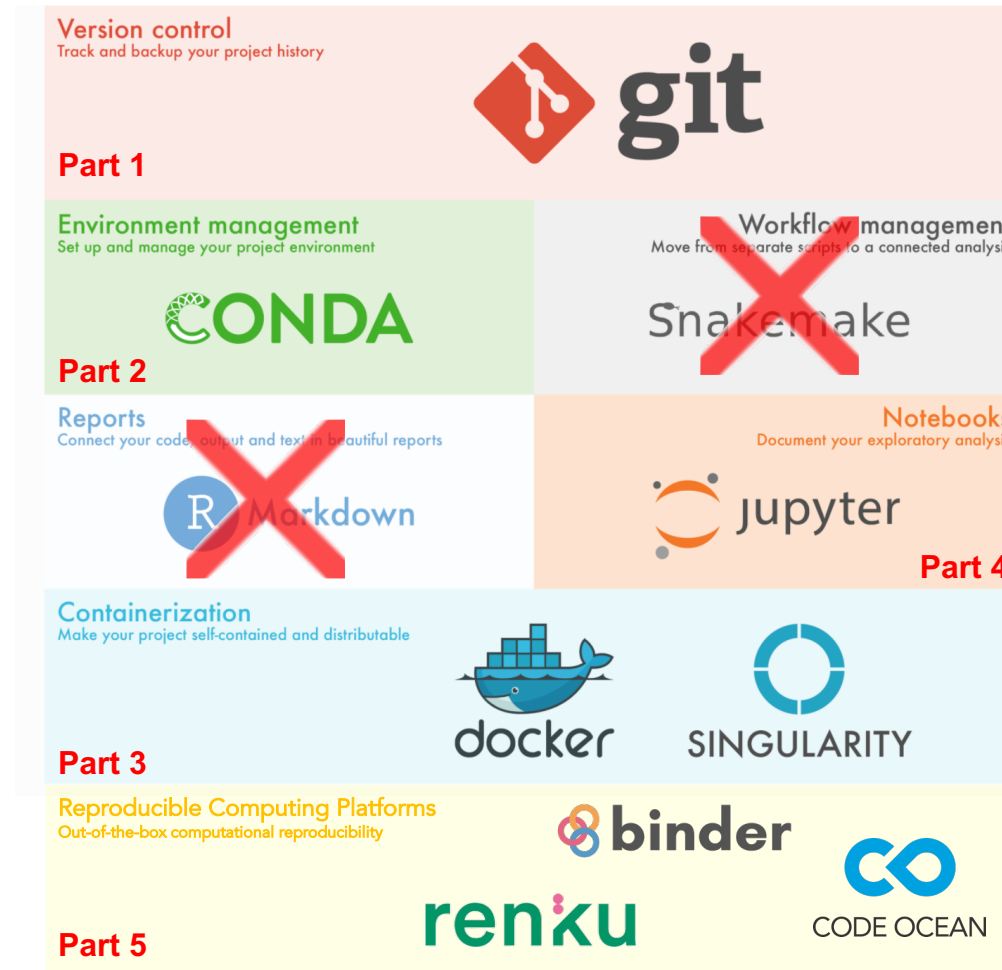
Computational Reproducibility: Pieces of the Puzzle

All parts of a computational analysis have to be reproducible!



Computational Reproducibility: Pieces of the Puzzle

What is covered in today's workshop? And what not?

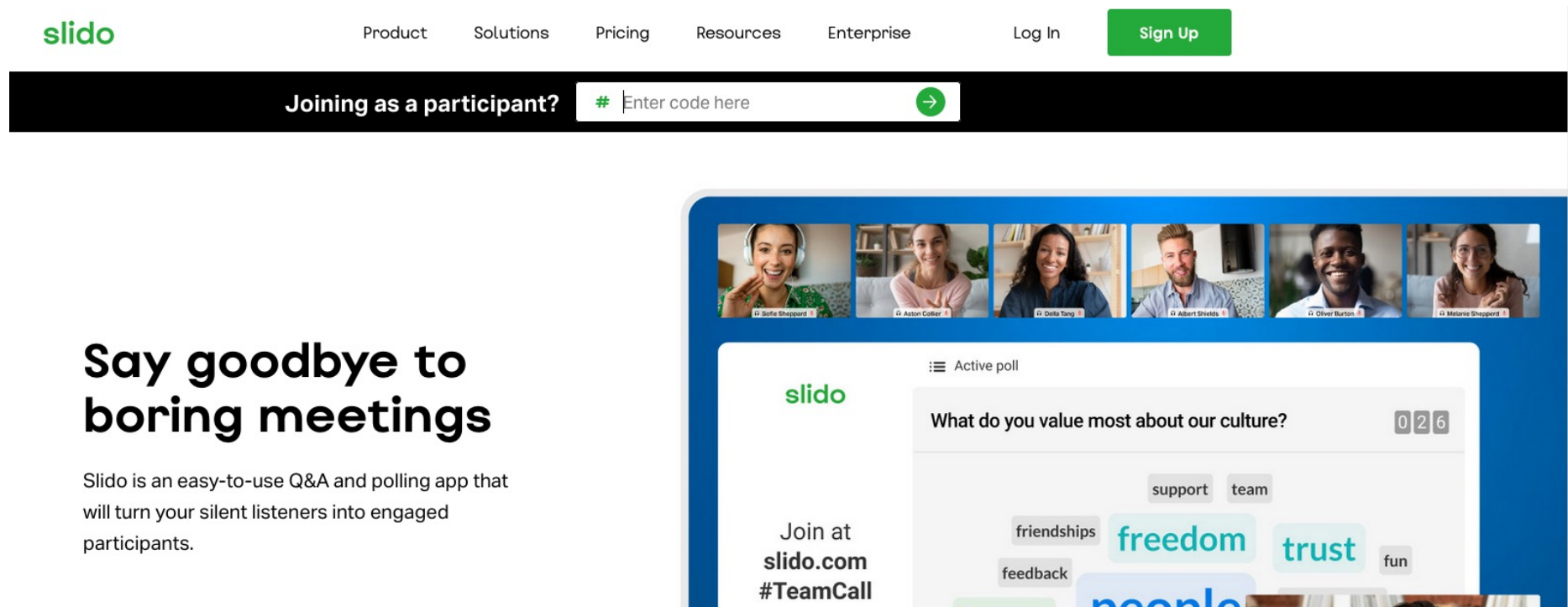


Computational Reproducibility: Questions?



Tell us a bit about yourself

- Go to www.slido.com and enter the event code **#reproducibility**



The image shows a screenshot of the Slido website and a meeting interface. The website header includes the Slido logo, navigation links for Product, Solutions, Pricing, Resources, and Enterprise, and buttons for Log In and Sign Up. A prominent black banner asks "Joining as a participant?" and features a text input field with a placeholder "# Enter code here" and a green arrow button. Below the banner, the meeting interface is displayed, showing a grid of six video thumbnails for participants: Sofie Sheppard, Aston Carter, Diella Tang, Albert Shields, Oliver Burton, and Miriana Sheppard. The main content area shows an active poll titled "What do you value most about our culture?" with 026 votes. The poll results are displayed as word clouds, with "freedom" and "trust" being the most prominent words. Other visible words include "support", "team", "friendships", "feedback", "fun", and "people". The Slido logo is also visible in the top left of the meeting interface. In the bottom left corner of the meeting interface, it says "Join at slido.com #TeamCall".

Say goodbye to boring meetings

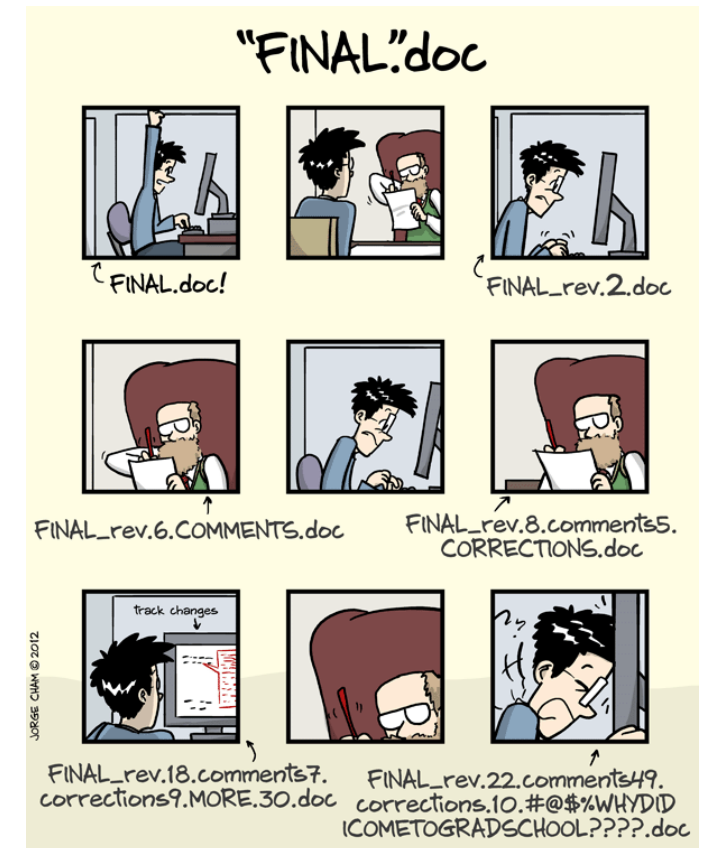
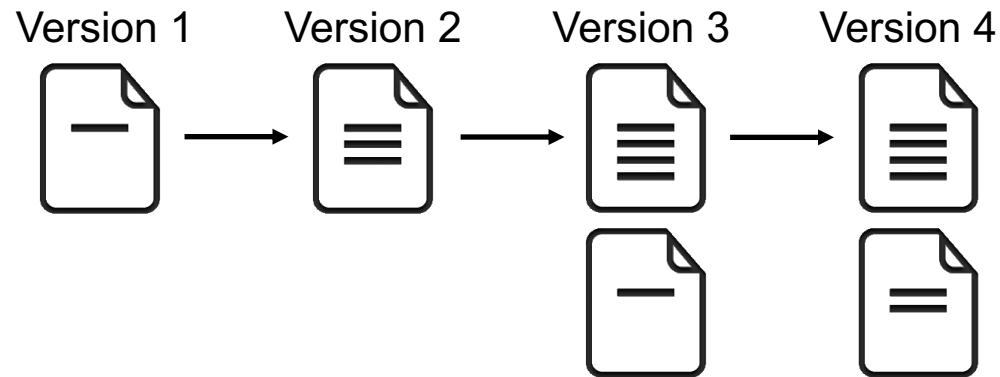
Slido is an easy-to-use Q&A and polling app that will turn your silent listeners into engaged participants.

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 files over time
 - A VCS is useful for tracking changes in any (text-based) content

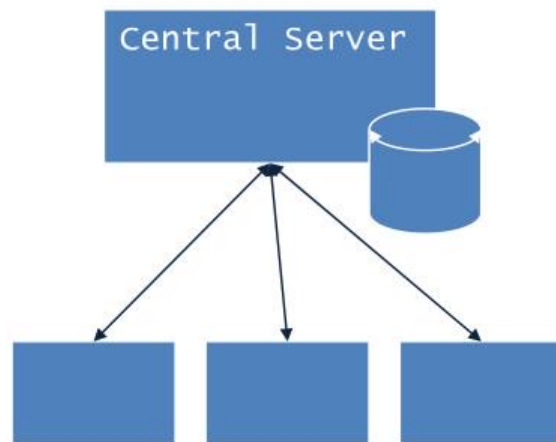


<https://phdcomics.com/comics/archive.php?comicid=1531>

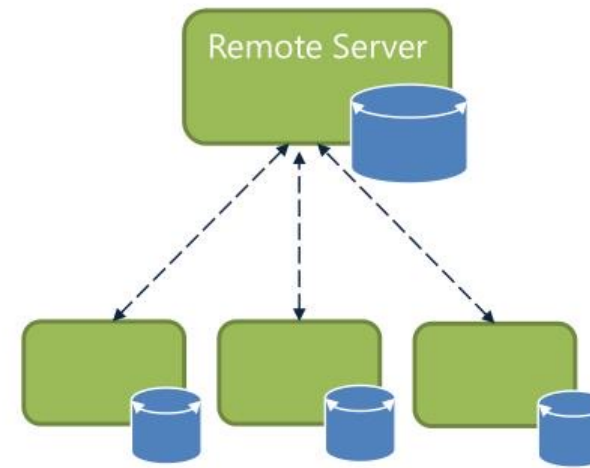
Version Control Systems & Git



- In 2022, Git is (by far) the most popular Version Control System
- Some Git facts
 - Free & open source VCS developed by Linus Torvalds in 2005
 - Actively developed by a large software community
 - Git is a *distributed* VCS (as opposed to *centralized* systems)
 - Git powers the [GitHub](https://github.com) platform with millions of software projects



Centralized VCS



Distributed VCS

Version Control Systems & Git

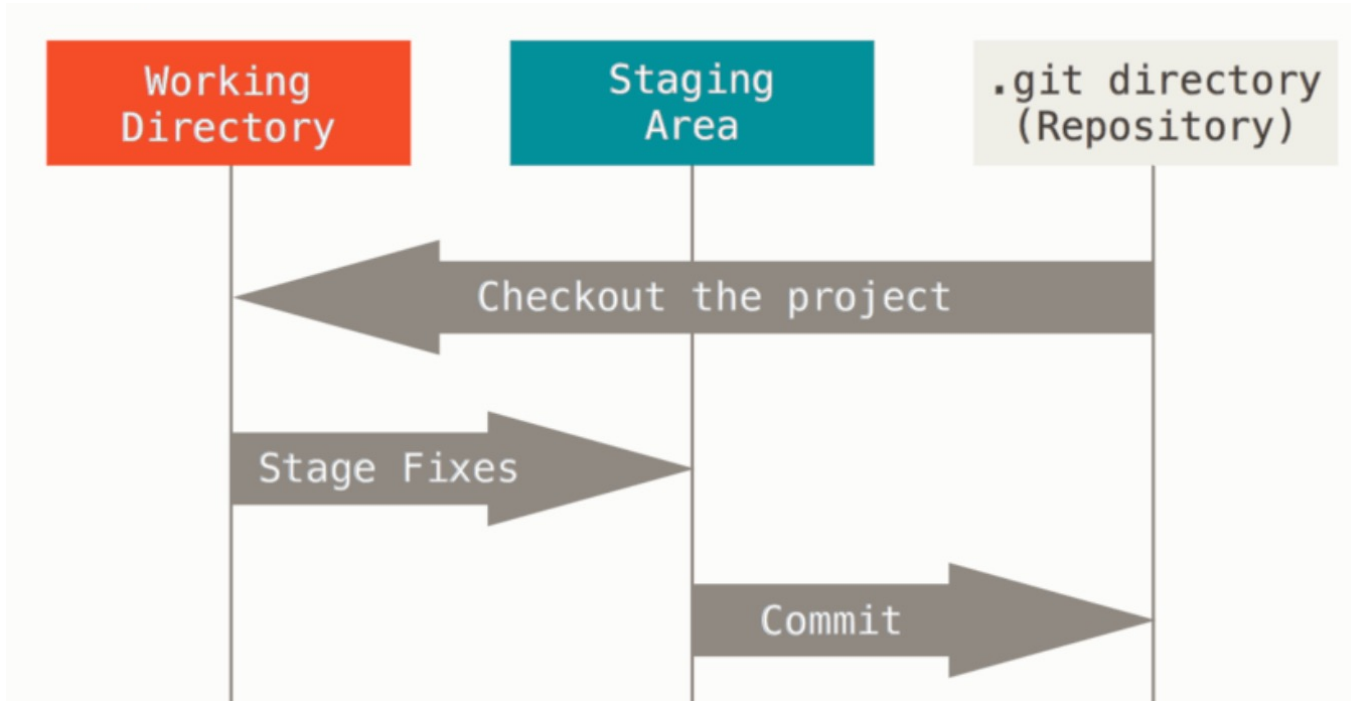


- In 2022, Git is (by far) the most popular Version Control System
- Some Git facts
 - Free & open source VCS developed by Linus Torvalds in 2005
 - Actively developed by a large software community
 - Git is a *distributed* VCS (as opposed to *centralized* systems)
 - Git powers the [GitHub](https://github.com) platform with millions of software projects
- Git allows you to
 - Examine how content evolved over time
 - Split project development into parallel lines which can evolve separately (“branches”)
 - Recombine branches and reconciling differences (“merging”)
 - Work collaboratively on a project and share it with others

Git basics

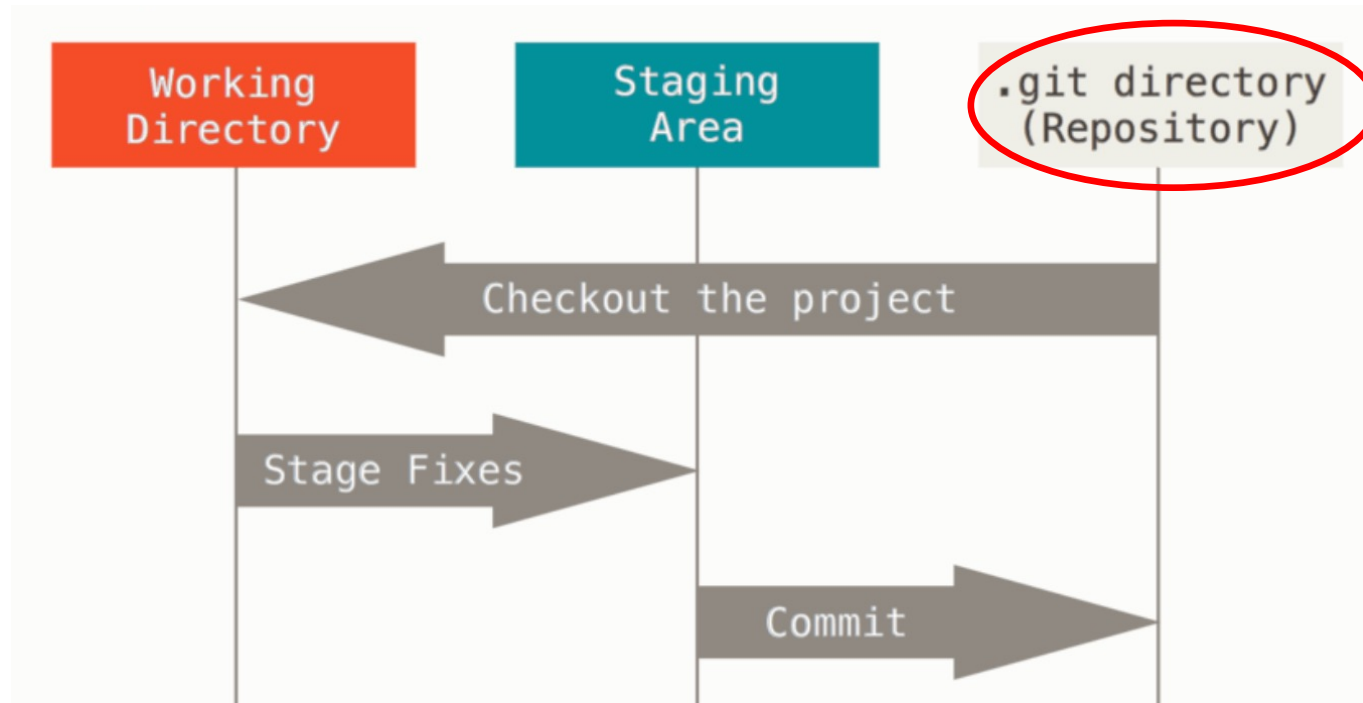


- **Goal of this part:** introduce basic principles & concepts of Git



The basic Git workflow

- Modify files in your working tree
- Selectively stage just those changes you want to be part of your next commit, which adds **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



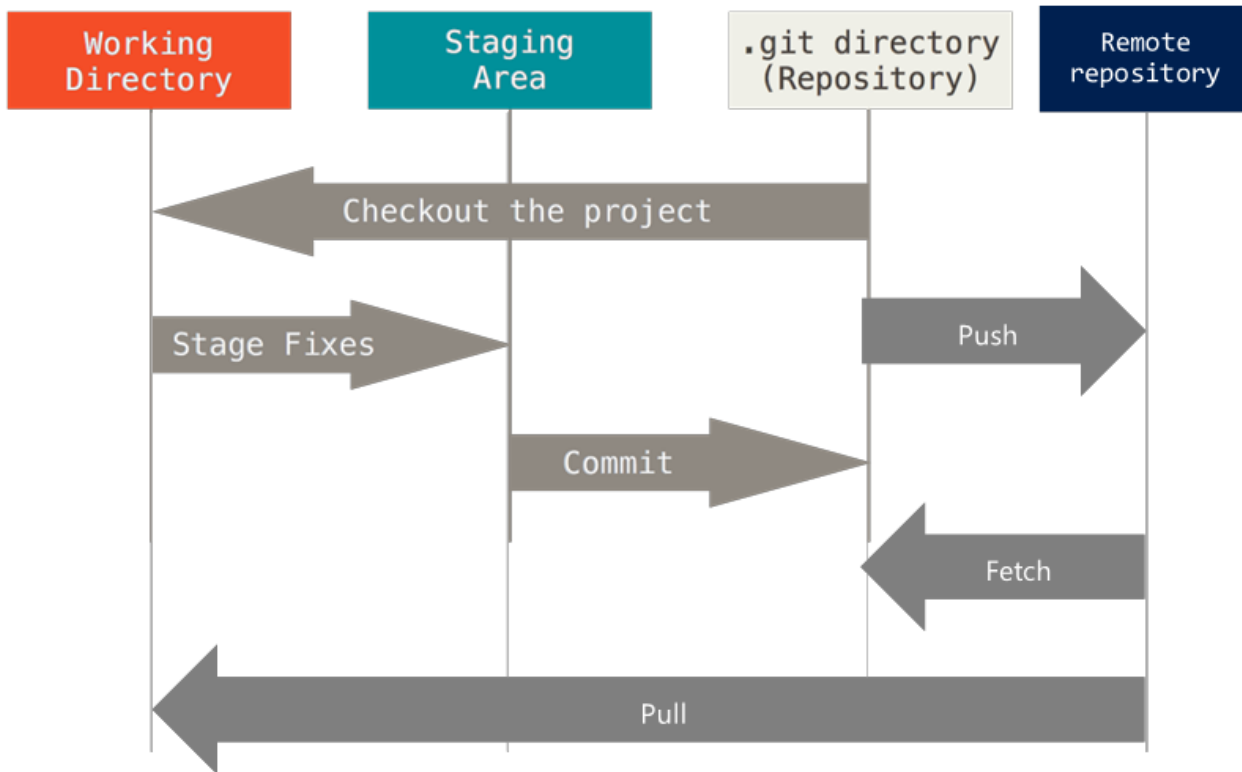
The .git directory (Repository)

- Contains the version database for your project
- Self-contained with all contents and full history
- Created by cloning or initializing a Git project
- Do not try to edit files in the .git directory directly (use git commands instead)

Git Remote Repositories



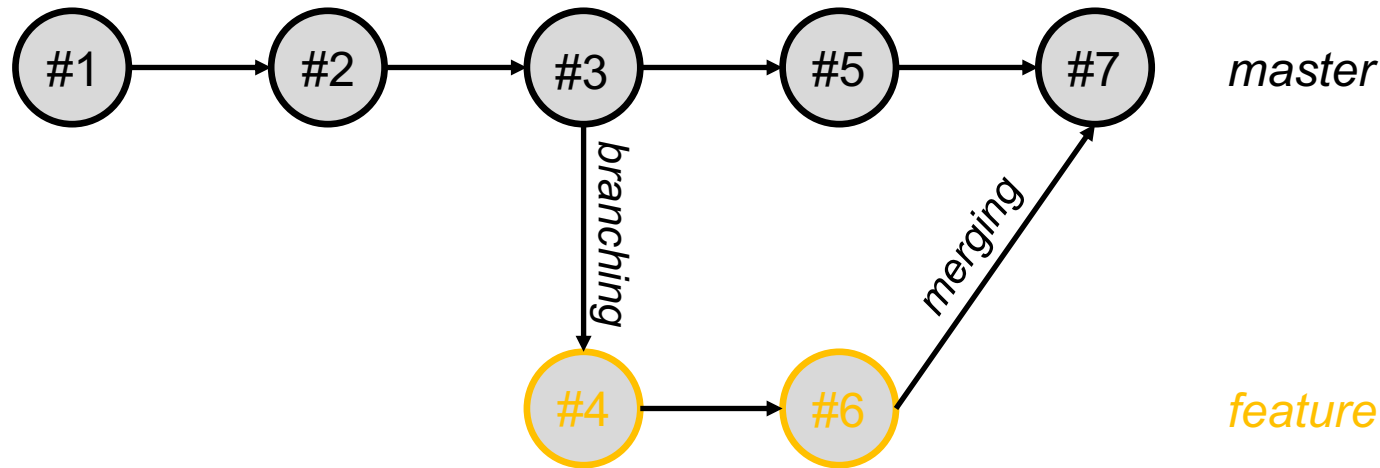
GitLab



Collaboration & sharing with Git

- Using Git for personal code management on your own computer is perfectly fine
- Remote repositories add capabilities for collaborative work, sharing and provide a convenient backup solution
- Typically Git-based platforms are used (GitLab, GitHub etc.)
- Download work from remote: *fetch / pull*
- Upload work to remote: *push*
- *Note:* Git-based platforms typically do not qualify as FAIR data repositories
 - [GitHub integration with Zenodo](#) to archive repositories and receive DOIs

Git branching & merging



Git branches & merges

- The last commit is the branch “tip”
- A branch is defined as collection of commits reachable from the tip
- The initial / default branch is typically called *master* or *main*
- Git manages branches very efficiently
- When merging merging branches, conflicts must be resolved carefully

ETH Zurich GitLab Service



The screenshot shows the GitLab web interface for a project named 'experimental-project-1'. The browser address bar shows the URL <https://gitlab.ethz.ch/sis-rdm-training/experimental-project-1>. The page features a navigation sidebar on the left with options like Project, Details, Activity, Releases, Cycle Analytics, Repository, Issues (1), Merge Requests (0), CI / CD, Operations, Wiki, Snippets, and Settings. The main content area displays the project details, including the project name 'experimental-project-1' with a lock icon, Project ID 6107, and statistics for 7 commits, 1 branch, 0 tags, and 9.5 MB files. A commit history section shows a recent commit titled 'change' by Henry Luetcke, 4 months ago, with a commit hash of 657f9d3a. Below this, there are several buttons for adding project features like README, CHANGELOG, CONTRIBUTING, Auto DevOps, and Kubernetes cluster. At the bottom, a table lists the files in the repository.

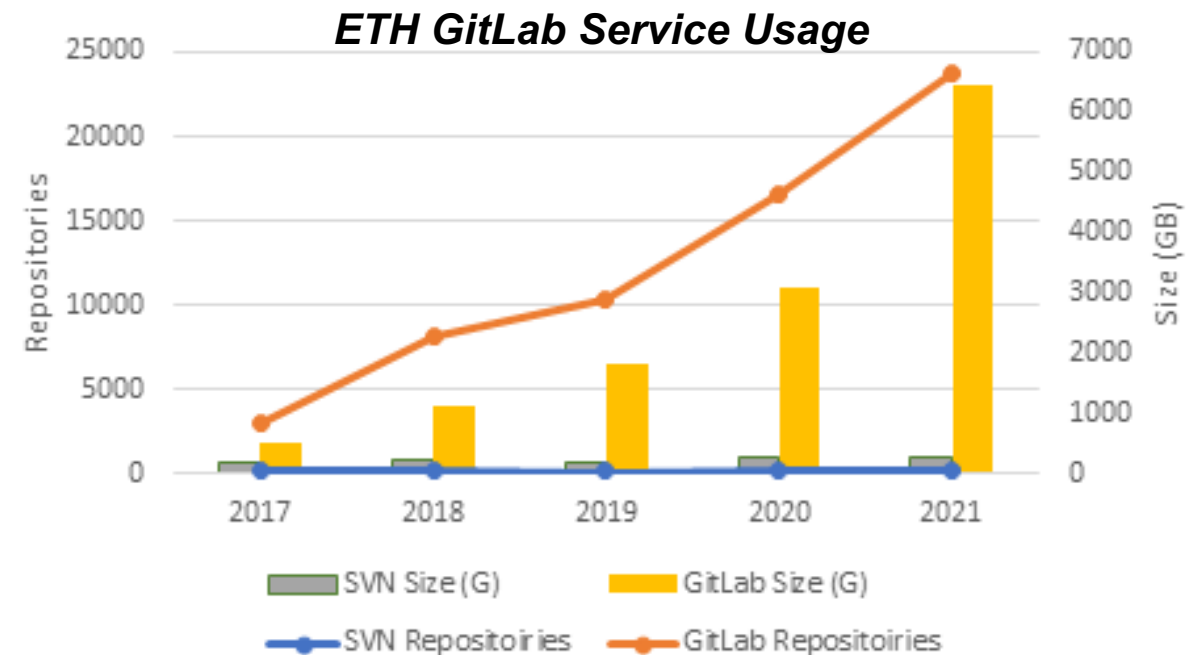
Name	Last commit	Last update
data	change image file size	4 months ago
.gitattributes	my first commit	5 months ago
analysis_code.py	change	4 months ago

<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.)
- Keep track of version history for everything
- Free for small repositories (< 2GB), otherwise yearly price of 250 CHF / TB / year
- Local and remote copies (off-site backup)
- Easily change permissions from private to public (e.g. after publication)
- 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
 - e.g. 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?



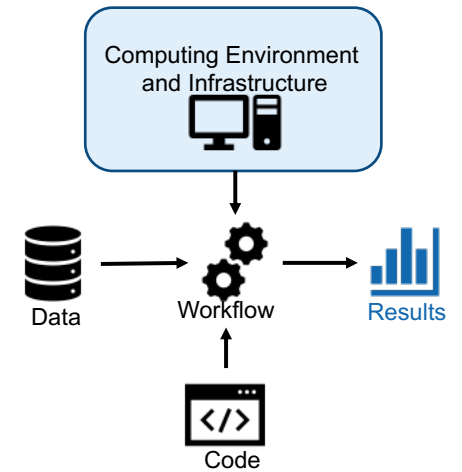
Managing Dependencies & Computing Environments



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

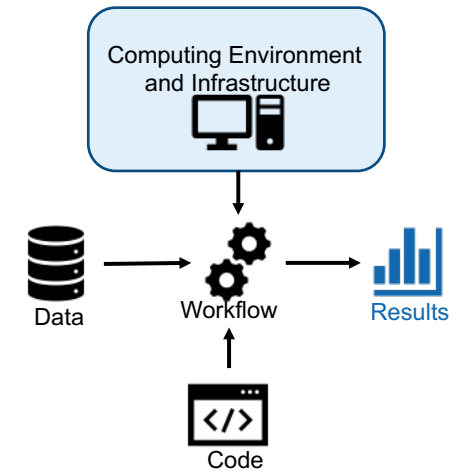
Solution:

Bundle your application and all dependencies

→ Environment Isolation & Dependency management

Tools:

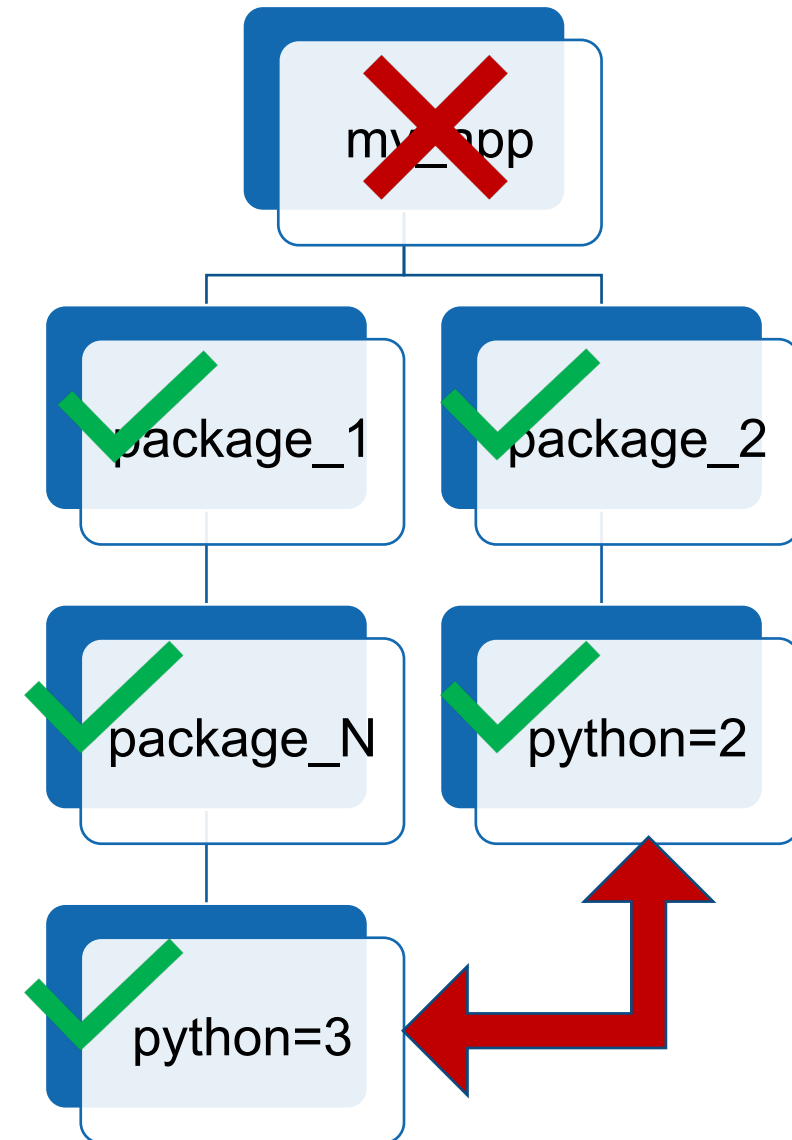
- Application / software level: Conda, pip, virtualenv, renv
- Containerization: Docker
- Virtualization (Virtual Machine, VM): VirtualBox, VMware



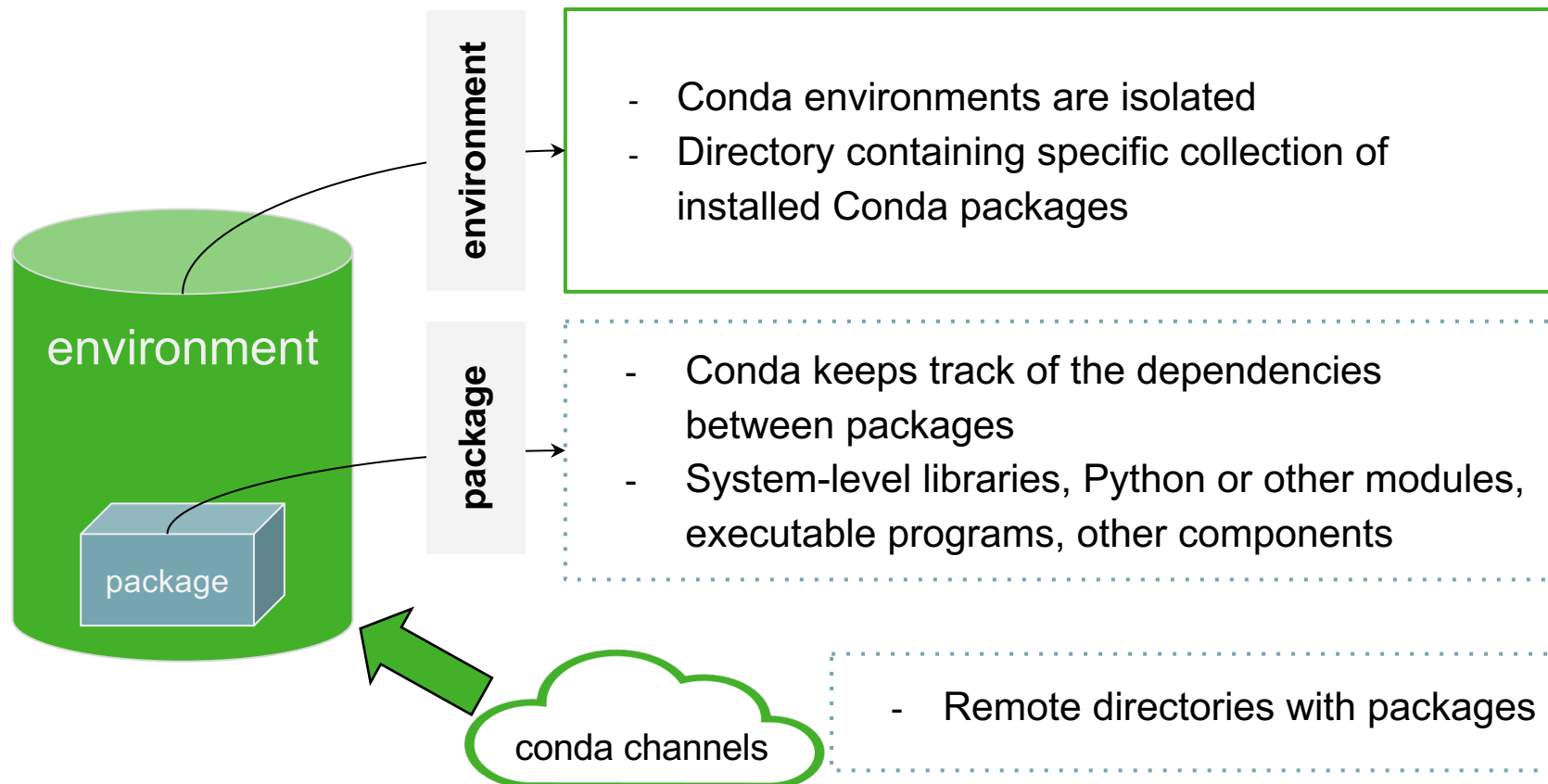
Reproducible Environment for R and Python



- Open source: Anaconda and Miniconda
- Commercial support: Anaconda Enterprise
- 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:  
- defaults  
- conda-forge  
dependencies:  
- python=3.8  
- jupyterlab
```

Conda automatically creates an environment file with packages and dependencies

Environment and Package Management Systems

Programming Language	Environment Management System	Package Management System	Comments
Python 2 (not supported)	virtualenv, conda	pip, conda	
Python 3	venv, virtualenv, pipenv poetry, conda	pip, pipenv, poetry, conda	only conda can install different Python versions (pyenv can be used)
R	packrat (soft-deprecated), renv, conda	packrat (soft-deprecated), 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's search path determines dependencies

Conda Hands-on Session



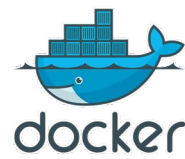
https://siscourses.ethz.ch/reproducible_computing/Conda.slidy.html



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

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



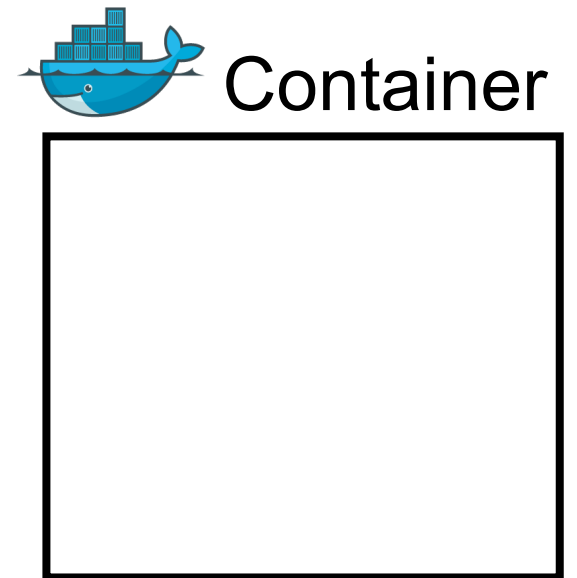
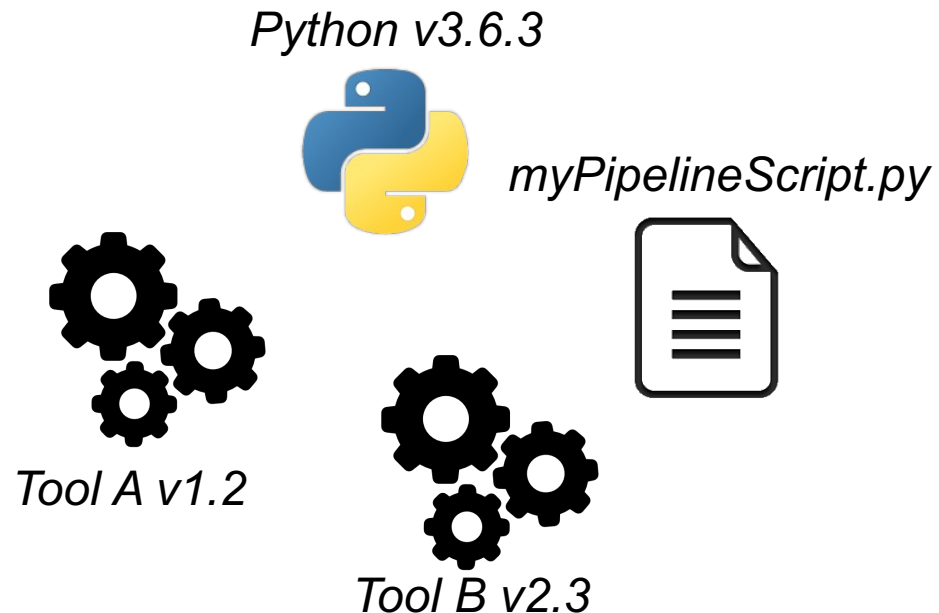
Source: <https://searchservirtualization.techtarget.com/definition/virtual-machine>

Reproducible Environment – Containerization

- **Container:** Operating system level **virtualization method** for running software without launching an entire virtual machine
- In simpler words: containers allow you to **package** your software / pipeline with the **dependencies** inside a **reproducible**, easy to **share**, **runnable** file

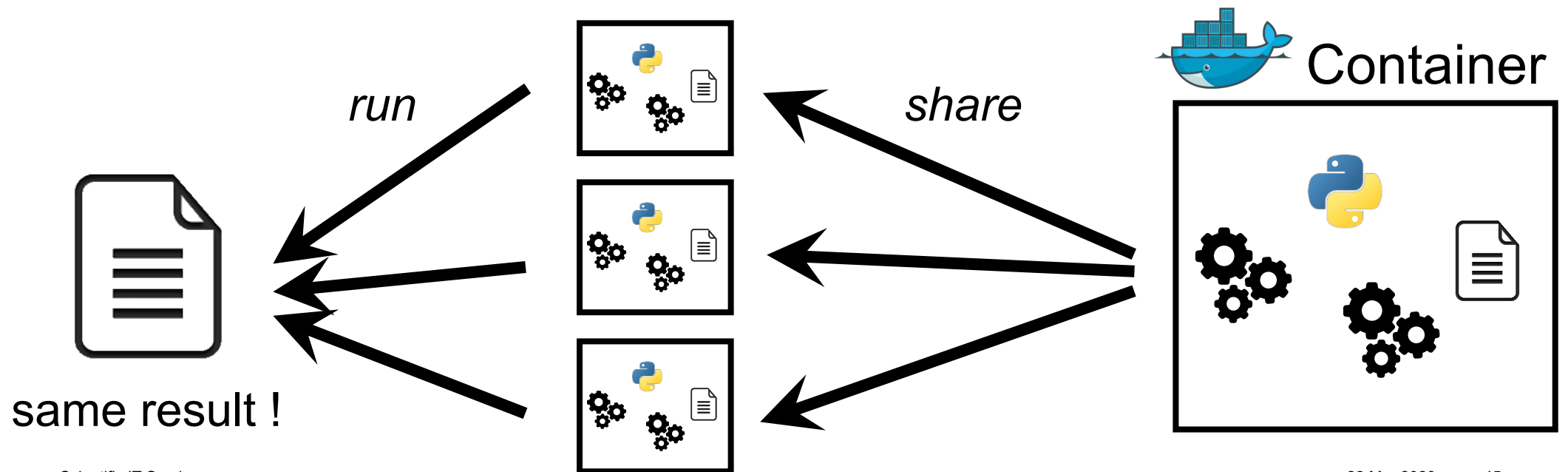
Reproducible Environment – Containerization

- **Container:** Operating system level **virtualization method** for running software without launching an entire virtual machine
- In simpler words: containers allow you to **package** your software / pipeline with the **dependencies** inside a **reproducible**, easy to **share**, **runnable** file
- Example: **Docker containers**



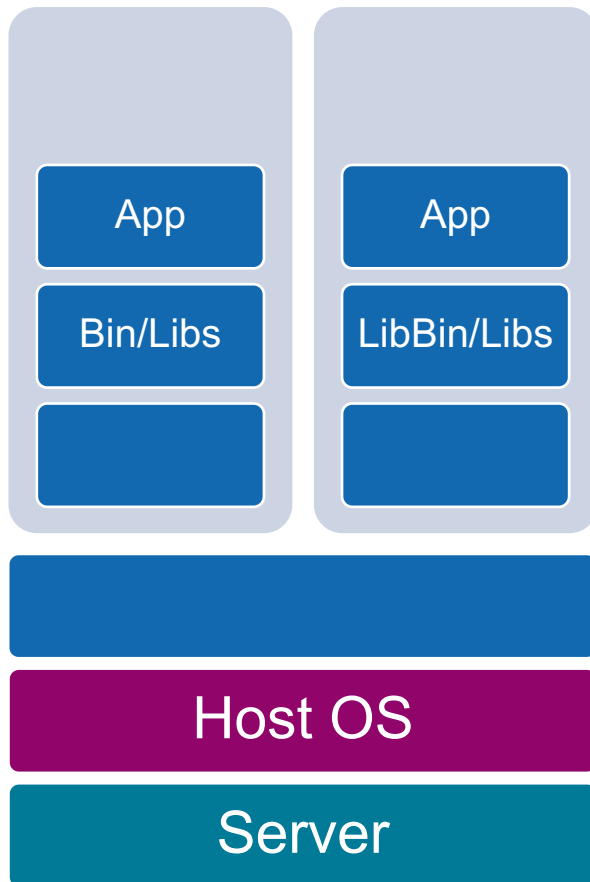
Reproducible Environment – Containerization

- **Container:** Operating system level **virtualization method** for running software without launching an entire virtual machine
- In simpler words: containers allow you to **package** your software / pipeline with the **dependencies** inside a **reproducible**, easy to **share**, **runnable** file
- Example: **Docker containers**

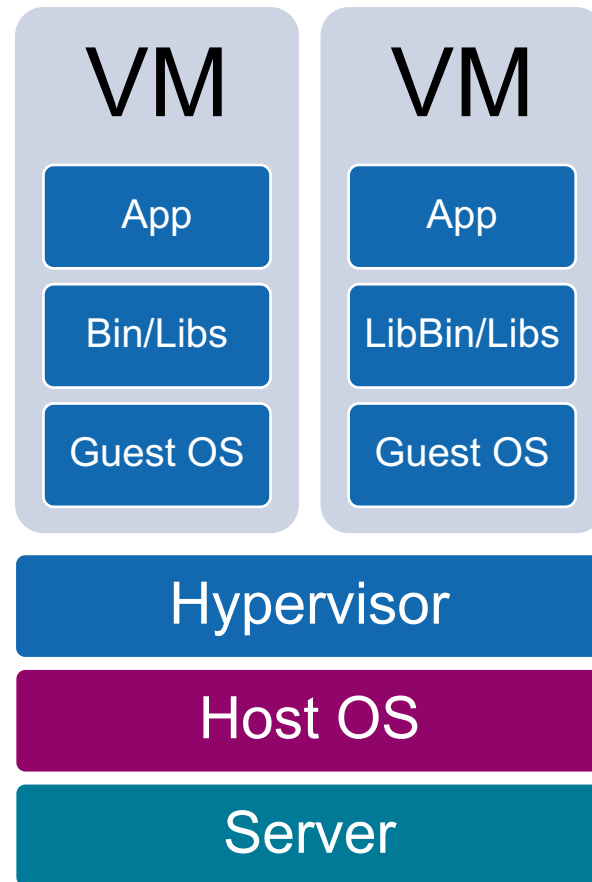


Bare Metal, Virtual Machine (VM) and Container (Docker)

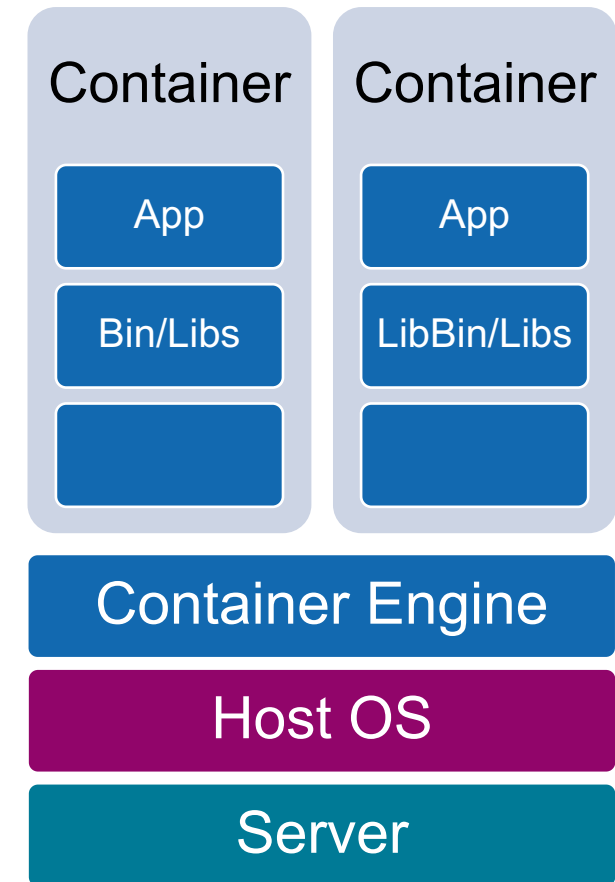
Bare Metal



VM Based



Container Based Shared Host OS kernel



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?





We explore the Lorenz system of differential equations:

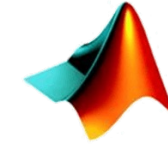
$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

Let's change (σ, β, ρ) with ipywidgets and examine the trajectories.

```
In [2]: from lorenz import solve_lorenz
w=interactive(solve_lorenz,sigma=(0.0,50.0),rho=(0.0,50.0))
w
```

sigma 10.00
beta 2.67
rho 28.00

```
def solve_lorenz(sigma=10.0, beta=8./3, rho=28.0):
    """Plot a solution to the Lorenz differential equations."""
    max_time = 4.0
    N = 30
    fig = plt.figure()
    ax = fig.add_axes([0, 0, 1, 1], projection='3d')
    ax.axis('off')
    # prepare the axes limits
    ax.set_xlim((-25, 25))
    ax.set_ylim((-35, 35))
    ax.set_zlim((5, 55))
    def lorenz_deriv(x,y,z, t0, sigma=sigma, beta=beta, rho=rho):
        """Compute the time-derivative of a Lorenz system."""
        x, y, z = x,y,z
        return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]
    # Choose random starting points, uniformly distributed from -15 to 15
    np.random.seed(1)
    x0 = -15 + 30 * np.random.random((N, 3))
    # Solve for the trajectories
    t = np.linspace(0, max_time, int(250*max_time))
    x_t = np.asarray([integrate.odeint(lorenz_deriv, x0i, t)
                      for x0i in x0])
    # choose a different color for each trajectory
    colors = plt.cm.viridis(np.linspace(0, 1, N))
    for i in range(N):
        x, y, z = x_t[i,:,:].T
        lines = ax.plot(x, y, z, '-', c=colors[i])
        plt.setp(lines, linewidth=2)
    angle = 104
    ax.view_init(30, angle)
```



MATLAB
Live Editor



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



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



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



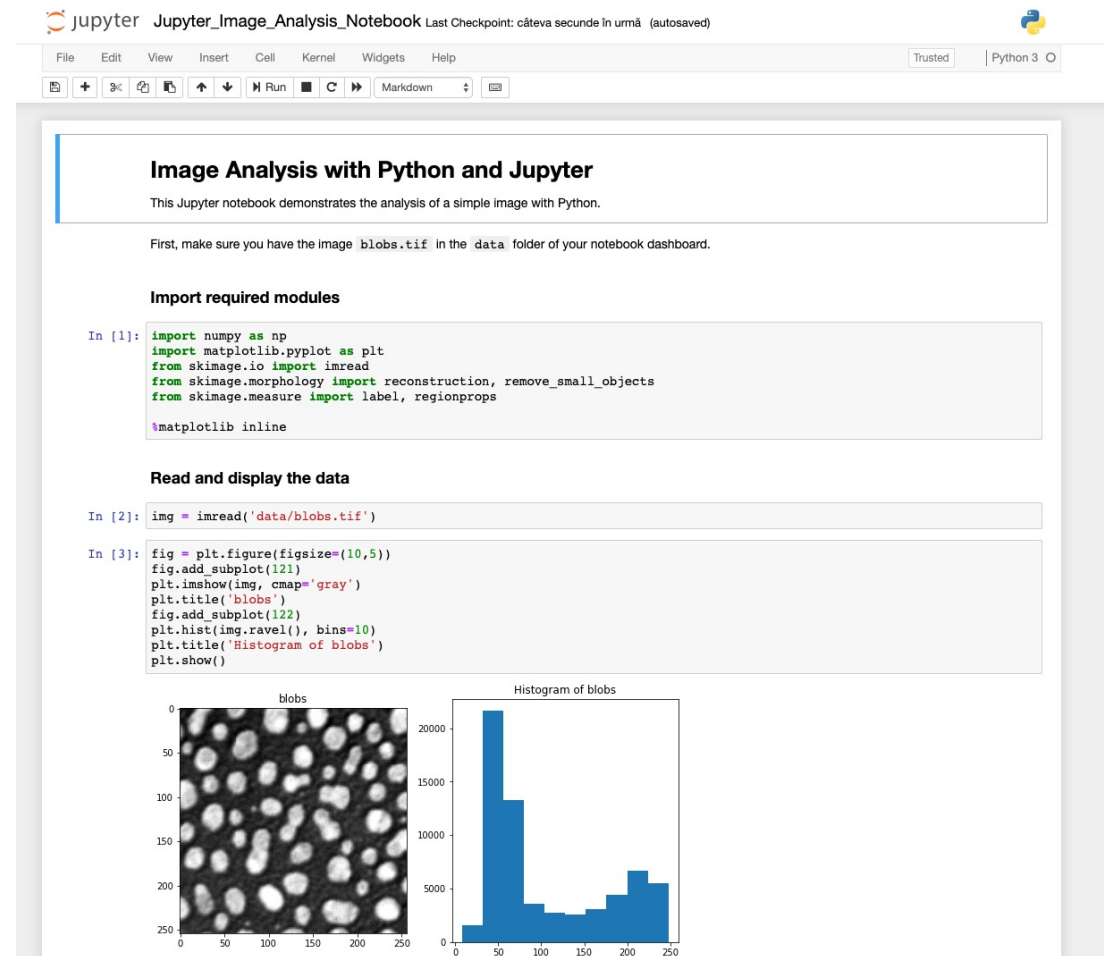
- Commercial
- Used in mathematical fields



- Commercial
- Used in scientific, engineering, mathematical fields

Interactive Notebooks: Jupyter

- **Jupyter notebook:** web-based interactive computational environment



The screenshot displays a Jupyter Notebook titled "Jupyter_Image_Analysis_Notebook" with a "Trusted" security level and "Python 3" kernel. The notebook content includes:

Image Analysis with Python and Jupyter

This Jupyter notebook demonstrates the analysis of a simple image with Python.

First, make sure you have the image `blobs.tif` in the `data` folder of your notebook dashboard.

Import required modules

```
In [1]: import numpy as np
import matplotlib.pyplot as plt
from skimage.io import imread
from skimage.morphology import reconstruction, remove_small_objects
from skimage.measure import label, regionprops

%matplotlib inline
```

Read and display the data

```
In [2]: img = imread('data/blobs.tif')

In [3]: fig = plt.figure(figsize=(10,5))
fig.add_subplot(121)
plt.imshow(img, cmap='gray')
plt.title('blobs')
fig.add_subplot(122)
plt.hist(img.ravel(), bins=10)
plt.title('Histogram of blobs')
plt.show()
```

The output shows two plots: a grayscale image of "blobs" and a histogram titled "Histogram of blobs". The image plot has axes from 0 to 250. The histogram plot has an x-axis from 0 to 250 and a y-axis from 0 to 20000.

Interactive Notebooks: Jupyter

- **Jupyter notebook:** web-based interactive computational environment
- **JupyterLab:** next-generation for Jupyter notebooks (and more)

The screenshot displays a JupyterLab environment. On the left is a file browser showing a directory structure with files like 'data', 'requirements.txt', and 'README.md'. The main area contains a notebook with the following content:

Image Analysis with Python and Jupyter

This Jupyter notebook demonstrates the analysis of a simple image with Python.

First, make sure you have the image `blobs.tif` in the `data` folder of your notebook dashboard.

Import required modules

```
[1]: import numpy as np
import matplotlib.pyplot as plt
from skimage.io import imread
from skimage.morphology import reconstruction, remove_small_objects
from skimage.measure import label, regionprops

%matplotlib inline
```

Read and display the data

```
[2]: img = imread('data/blobs.tif')

[3]: fig = plt.figure(figsize=(10,5))
fig.add_subplot(121)
plt.imshow(img, cmap='gray')
plt.title('blobs')
fig.add_subplot(122)
plt.hist(img.ravel(), bins=10)
plt.title('Histogram of blobs')
plt.show()
```

The notebook output shows two plots: a grayscale image of 'blobs' and a histogram titled 'Histogram of blobs' showing the distribution of pixel values.

On the right sidebar, there are two files:

- requirements.txt**:

```
1 # This file may be used to create an environment using:
2 # $ conda create --name <env> --file <this file>
3 # platform: osx-64
4 matplotlib
5 numpy
6 pandas
7 scikit-image
8 scipy
9
```
- README.md**:

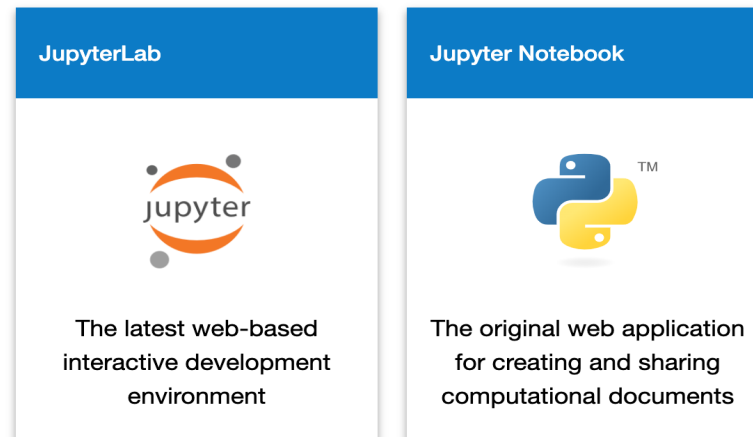
```
1 # Jupyter-Demo-RDM
2
3 Demo of Jupyter notebook for ETH ARDM workshops
4
5 [[Binder]](https://mybinder.org/badge_logo.svg)
6 (https://mybinder.org/v2/gh/hluetck/Jupyter-Demo-
7 RDM/master)
8
```

Interactive Notebooks: Jupyter

- **Jupyter notebook:** web-based interactive computational environment
- **JupyterLab:** next-generation for Jupyter notebooks (and more)
- Dozens of programming languages supported (core: **Julia**, **Python**, **R**)
- Extensions to build simple user interfaces (sliders, buttons etc.)
- Notebook export in various formats (HTML, PDF, Python ...)
- Sharing of interactive notebooks on reproducibility platforms
- Integration with ETH scientific computing infrastructure
(see <https://gitlab.ethz.ch/sfux/Jupyter-on-Euler-or-Leonhard-Open> and <https://jupyter.euler-dev.hpc.ethz.ch/>)
- **JupyterHub:** multi-user version of the notebook for research labs

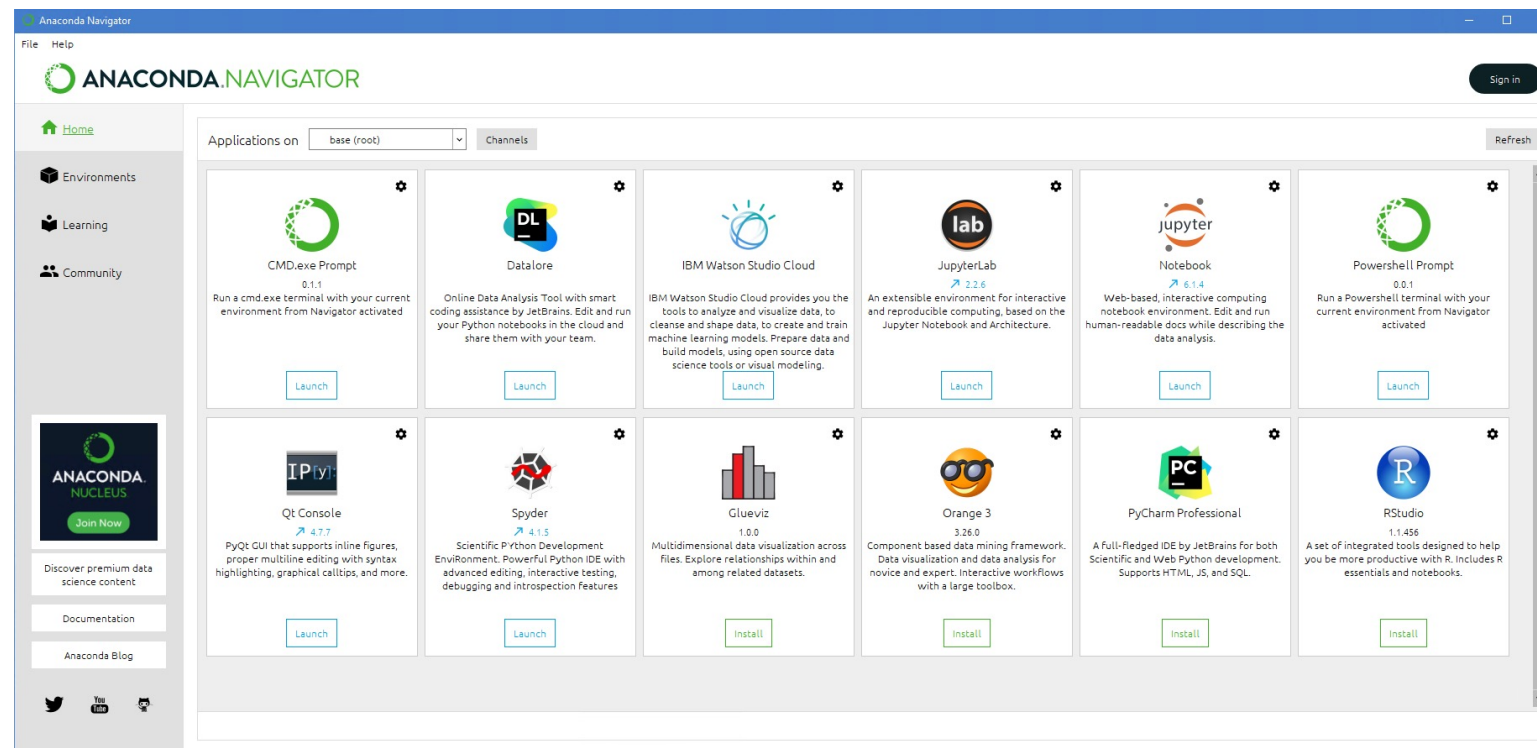
Options for running Jupyter

- Local installation on your computer
- Dedicated JupyterHub server (e.g. running on virtual machine in the cloud)
- Public cloud-based offerings
 - Renku: <https://renkulab.io/>
 - MyBinder: <https://mybinder.org/>
 - Google cloud: <https://colab.research.google.com/notebooks>
- To get started
 - <https://jupyter.org/try>



Local installation of Jupyter

- **Option 1: [Anaconda](#)**
 - Installs Jupyter, Python, R and many other packages
 - Start JupyterLab or Notebook from Anaconda Navigator



Local installation of Jupyter

- **Option 1: [Anaconda](#)**
 - Installs Jupyter, Python, R and many other packages
 - Start JupyterLab or Notebook from Anaconda Navigator
- **Option 2: [Miniconda](#)**
 - `conda install -c conda-forge jupyterlab`
 - Start JupyterLab: `jupyter-lab`
 - Start Notebook: `jupyter-nbclassic`
- **Option 3: [Python](#) only**
 - `pip install --upgrade pip wheel`
 - `pip install --upgrade jupyterlab`
 - Start Lab / Notebook: `jupyter-lab` / `jupyter-nbclassic`

Interactive Notebooks – what can go wrong?

- **Versioning**

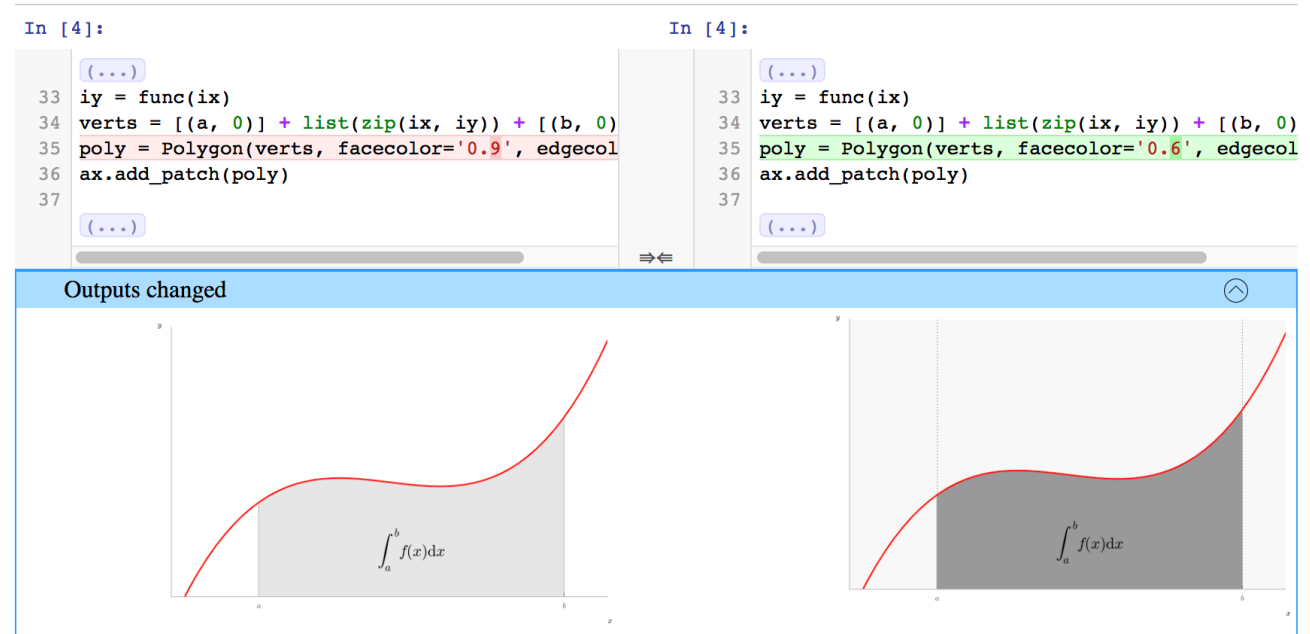
- Version control of even moderately complex NBs is challenging
- Tracking NB history is harder than for traditional source code
- Some tools may help (e.g. [nbdime](#), [JupyterText](#))

```
$ diff a.ipynb b.ipynb
76,77d75
<     "plt.rc('axes', grid=False)\n",
<     "plt.rc('axes', facecolor='white')\n",
90c88
<     "image/png": "iVBORw0KGgoAAAANSUgAABLkAAAMQCAYAAADLj7dLAAAABHNCSVQICAgIfAhki
AAAAA\lwSFlz\nAAAWJQAAFiUBSVIk8AAAIABJREFUeJzsvXeYZFd57b12h0maPNJII2lG0aCAkEBCFgozIxBAp
lY\n1waDyDZg8MX+zMU2F4Mx1x8PwWAwxmBjg4yNi2BfQMa20iiAQFkIjXKWRtJIE3tSz3TXuX+8vV2n\nqyucv
N+9z/o9zzynprvq1D6nqqqr1prbRNFEQghhBCCCCGEEEEII8Zkh1wMghBCCCCGEEEEIIISQv\nFLkIIYQQQgghhB
BCiPdQ5CKEEEEIIYQQQggh3kORixBCCCCGEEEEIIYR4D0UuQgghhBCCCCGEE0I9\nFLkIIYQQQgghhBBCiPdQ5CK
EEEEIIYQQQggh3kORixBCCCCGEEEEIIYR4D0UuQgghhBCCCCGEE0I9\nFLkIIYQQQgghhBBCiPdQ5CKEEEEIIYQQ
Qggh3kORixBCCCCGEEEEIIYR4D0UuQgghhBCCCCGEE0I9\nFLkIIYQQQjzEGH0JMaZljPmo67EkZWq8D7keByGEE
ELChCIXIYQQQirDGPOmKaFj3BhzkMNx/H/G\nnmG3GmP/pagwFEbkeQJUYY75gjNlijHmD67EQQgghRB8UuQgghB
BSJe+DCDMjAH7L4TjeAmA+gLc5\nHEMRGNcDqJi3AVgI4DddD4QQQggh+qDIRQghhJBKMMacCuBMAFsg4sy7jTH
DjobzZwBuBvBxR/dP\nnsVERADcC+LTrgRBCCCFEHxS5CCGEEFIVH4C4uP4SILQcBOD1LgYSRVEzjqIXR1H0frf3
T7IRRdFf\nnRlH0K1EUXe96LIQQQgjRB0UuQgghhJSOMWYpgP8BoAXg7wH8HcTN9Tsux0UIIYQQQsKBihchhBBC\
nguBdAQYAuDyKoscBfByAlgBnGwDe73BkhBCCCCkCChyEUTTTaRUjDEGUjTf0RyciK7eDMP3n65C\nnNyxchhBBC
```

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](#), [JupyterText](#))



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](#))
- **Scalability**
 - Scaling to large datasets is challenging (due to browser limitations)
- **Reproducibility**
 - Interactive working mode can result in hard-to-reproduce notebooks
 - Discipline is needed! Regular pruning & refactoring; “*Restart kernel & Run all*” is your friend
- **Collaboration**
 - Collaborative editing generally not supported
- **Security**
 - Data confidentiality & access controls may be problematic



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

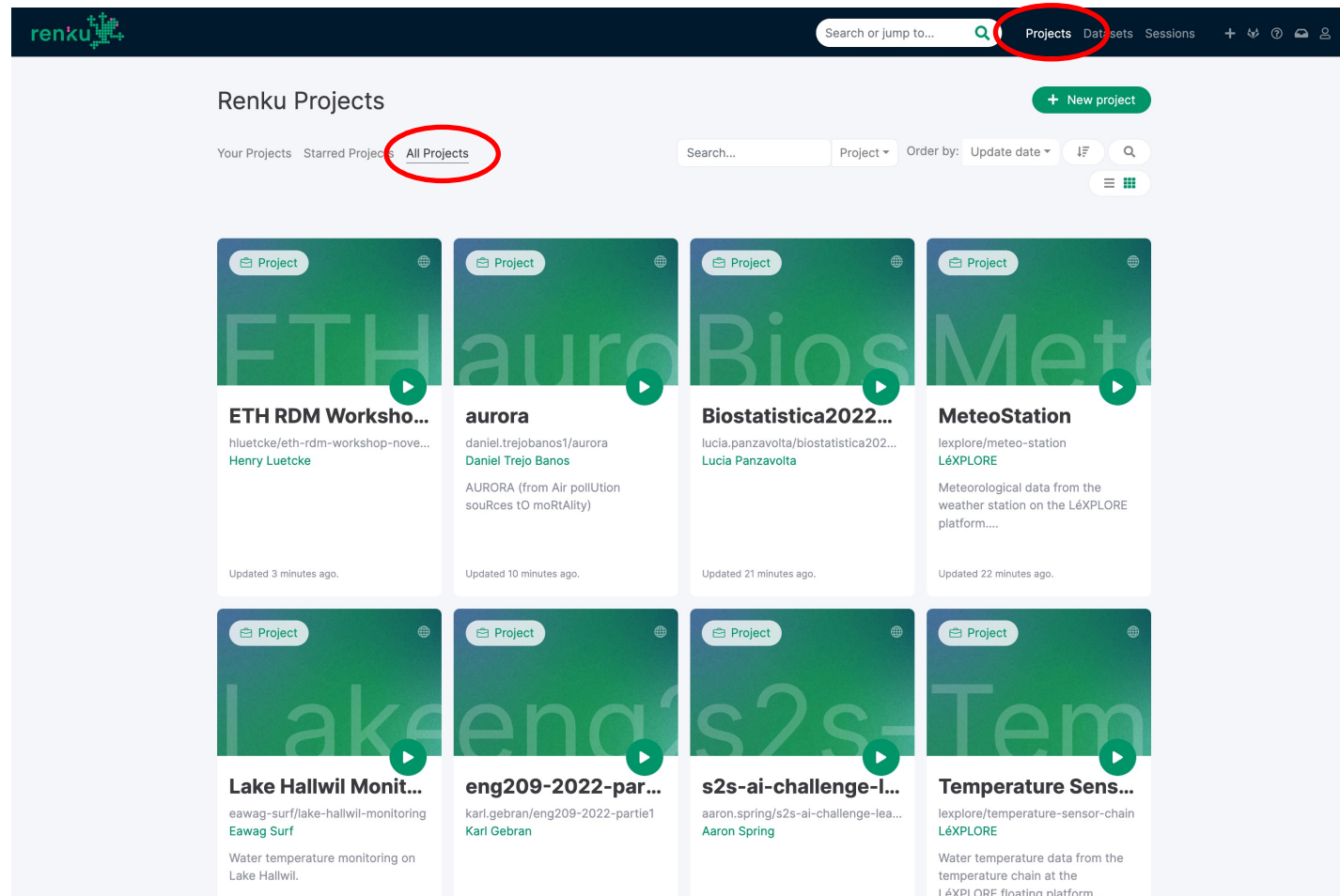
Reproducible Computing Platforms: *renkulab.io*

- [Renkulab](#) is a **platform for reproducible data science** from the [Swiss Data Science Center](#) (SDSC)



Reproducible Computing Platforms: *renkulab.io*

- [Renkulab](#) is a **platform for reproducible data science** from the [Swiss Data Science Center](#) (SDSC)
- First, login to Renkulab (use your Switch Edu-ID or register for a new account)
- After login, go to the Project Dashboard and list All Projects:

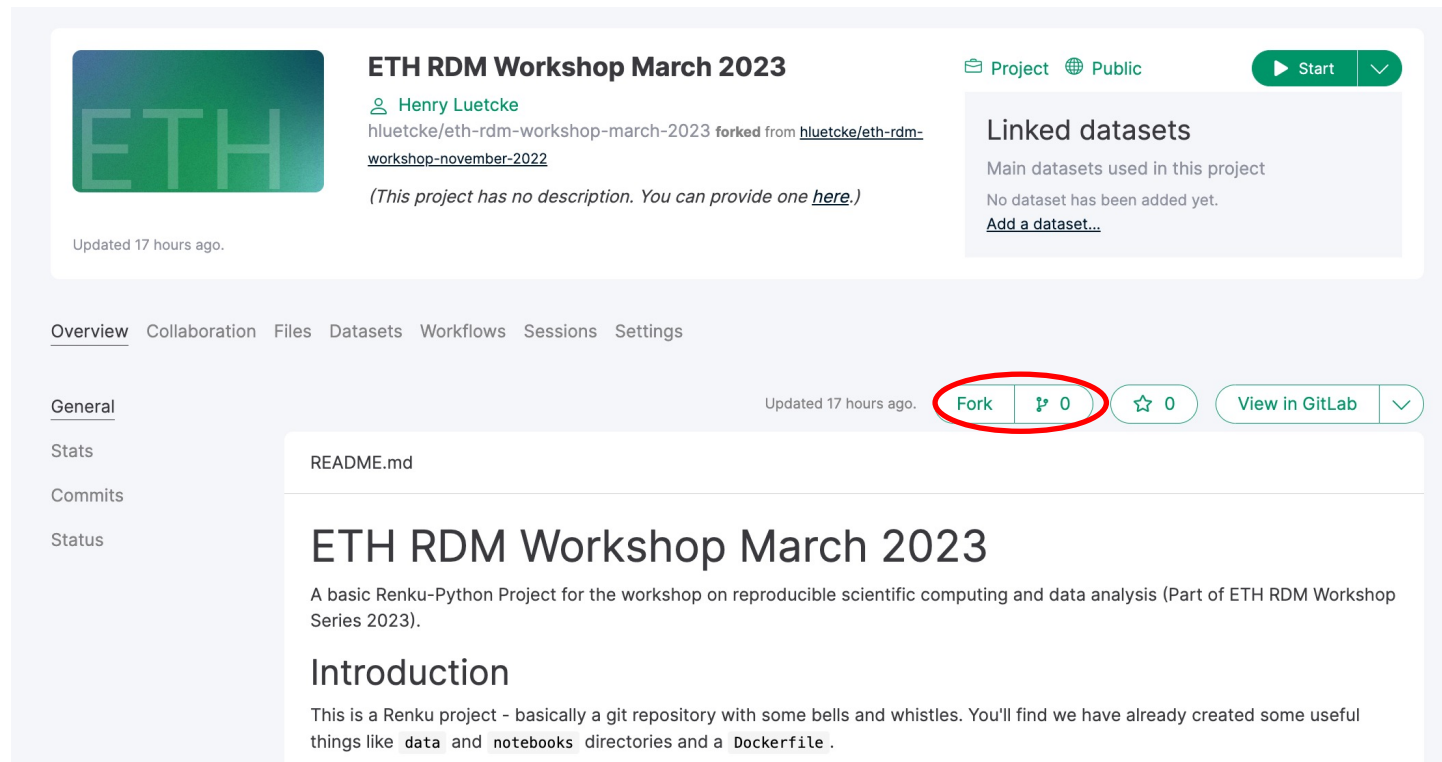


The screenshot displays the Renku Projects dashboard. At the top, the Renku logo is on the left, and a navigation bar contains 'Search or jump to...', 'Projects', 'Data sets', and 'Sessions'. The 'Projects' tab is highlighted with a red circle. Below the navigation bar, the main heading is 'Renku Projects' with a '+ New project' button. Underneath, there are tabs for 'Your Projects', 'Starred Projects', and 'All Projects', with 'All Projects' circled in red. A search bar and filters for 'Project' and 'Order by: Update date' are visible. The dashboard features a grid of project cards, each with a green header, a play button, and project details. The projects shown are:

- ETH RDM Worksho...** by Henry Luetcke (Updated 3 minutes ago)
- aurora** by Daniel Trejo Banos (Updated 10 minutes ago)
- Biostatistica2022...** by Lucia Panzavolta (Updated 21 minutes ago)
- MeteoStation** by LéXPLORE (Updated 22 minutes ago)
- Lake Hallwil Monit...** by Eawag Surf (Water temperature monitoring on Lake Hallwil)
- eng209-2022-par...** by Karl Gebran
- s2s-ai-challenge-l...** by Aaron Spring
- Temperature Sens...** by LéXPLORE (Water temperature data from the temperature chain at the LéXPLORE floating platform...)

Reproducible Computing Platforms: *renkulab.io*

- [Renkulab](#) is a **platform for reproducible data science** from the [Swiss Data Science Center](#) (SDSC)
- First, login to Renkulab (use your Switch Edu-ID or register for a new account)
- After login, go to the Project Dashboard and list All Projects
- Search for the project called *eth-rdm-workshop-march-2023* and fork it to your account



The screenshot shows the Renkulab project page for "ETH RDM Workshop March 2023" by Henry Luetcke. The project is public and was updated 17 hours ago. A red circle highlights the "Fork" button, which shows 0 forks. The project description states it is a basic Renku-Python project for the workshop on reproducible scientific computing and data analysis. The introduction mentions that the project includes data, notebooks, directories, and a Dockerfile.

ETH RDM Workshop March 2023
Henry Luetcke
hluetcke/eth-rdm-workshop-march-2023 forked from hluetcke/eth-rdm-workshop-november-2022
(This project has no description. You can provide one [here](#).)
Updated 17 hours ago.

Project Public Start

Linked datasets
Main datasets used in this project
No dataset has been added yet.
[Add a dataset...](#)

Overview Collaboration Files Datasets Workflows Sessions Settings

General Updated 17 hours ago. Fork 0 0 View in GitLab

Stats
Commits
Status

README.md

ETH RDM Workshop March 2023

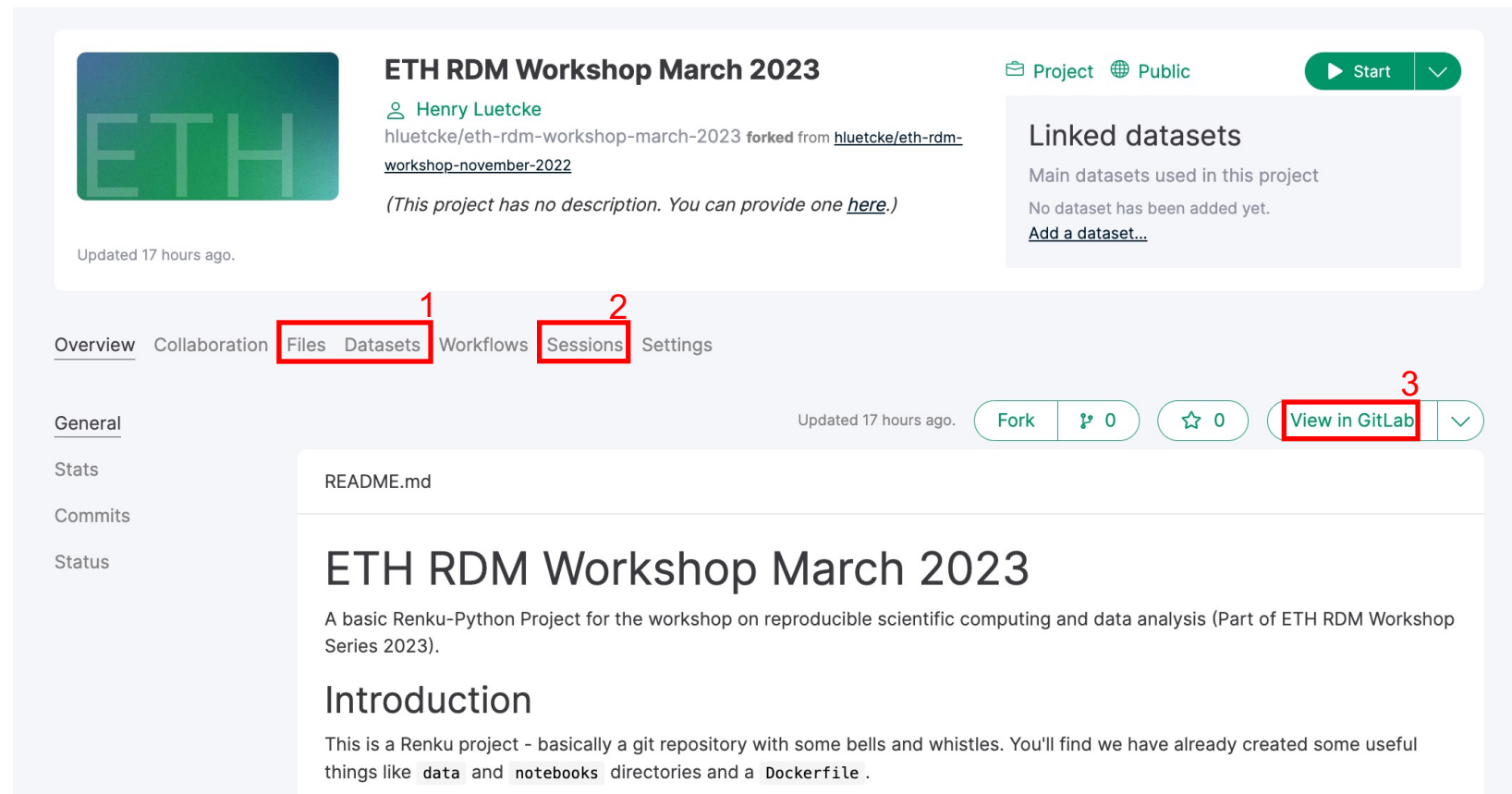
A basic Renku-Python Project for the workshop on reproducible scientific computing and data analysis (Part of ETH RDM Workshop Series 2023).

Introduction

This is a Renku project - basically a git repository with some bells and whistles. You'll find we have already created some useful things like `data` and `notebooks` directories and a `Dockerfile`.

Reproducible Computing Platforms: *renkulab.io*

- In the short demo, we will focus on 3 aspects of the platform related to reproducibility:
 - Files and datasets (1)
 - Compute sessions (2)
 - Integration with Gitlab (3)



ETH RDM Workshop March 2023

Henry Luetcke
hluetcke/eth-rdm-workshop-march-2023 forked from hluetcke/eth-rdm-workshop-november-2022
(This project has no description. You can provide one [here.](#))

Updated 17 hours ago.

Project Public Start

Linked datasets
Main datasets used in this project
No dataset has been added yet.
[Add a dataset...](#)

Overview Collaboration **Files** Datasets Workflows **Sessions** Settings

General Updated 17 hours ago. Fork 0 0 **View in GitLab**

Stats

Commits

Status

README.md

ETH RDM Workshop March 2023

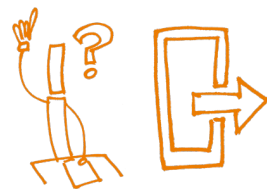
A basic Renku-Python Project for the workshop on reproducible scientific computing and data analysis (Part of ETH RDM Workshop Series 2023).

Introduction

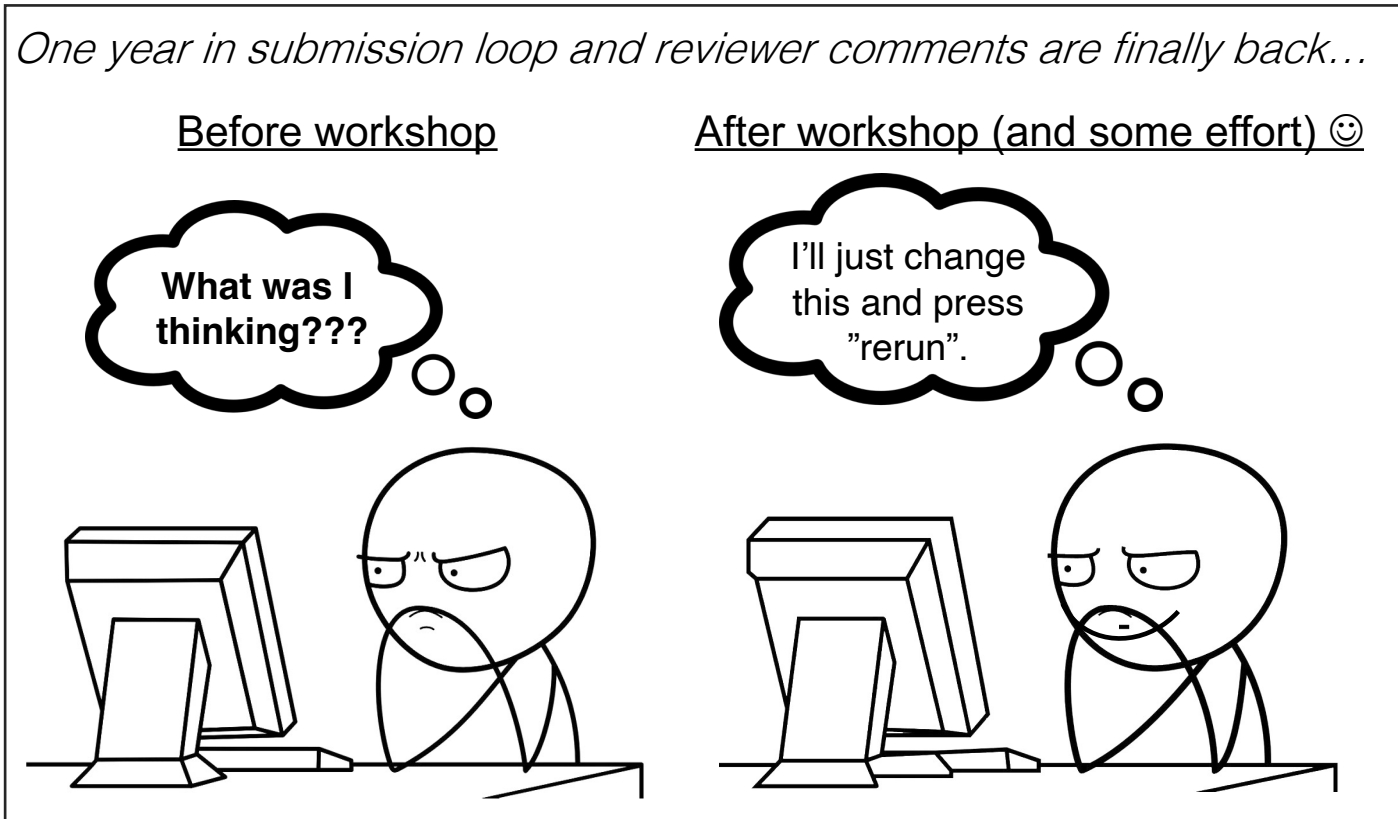
This is a Renku project - basically a git repository with some bells and whistles. You'll find we have already created some useful things like `data` and `notebooks` directories and a `Dockerfile`.

- For a more in-depth introduction, please see SDSC's [First Steps Tutorial](#)

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

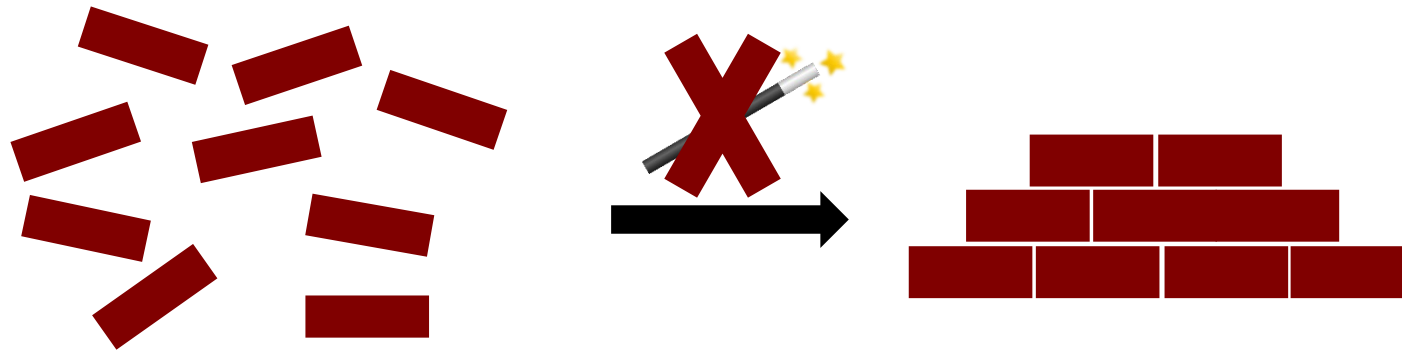
- Easier to rerun experiments and analysis
- Closer interaction between collaborators
- Much of the manuscript "writes itself"

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

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



Contacts

Nadia Marounina

nadejda.marounina@id.ethz.ch

Henry Lütcke

henry.lutcke@id.ethz.ch

sis.helpdesk@ethz.ch

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

Feedback: <https://www.umfrageonline.ch/s/a13b937>



Any final questions on what we have discussed this morning?

