

Reproducible Scientific Computing and Data Analysis

Nadia Marounina, Henry Lütcke
Scientific IT Services, ETH Zurich

March 26, 2025

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



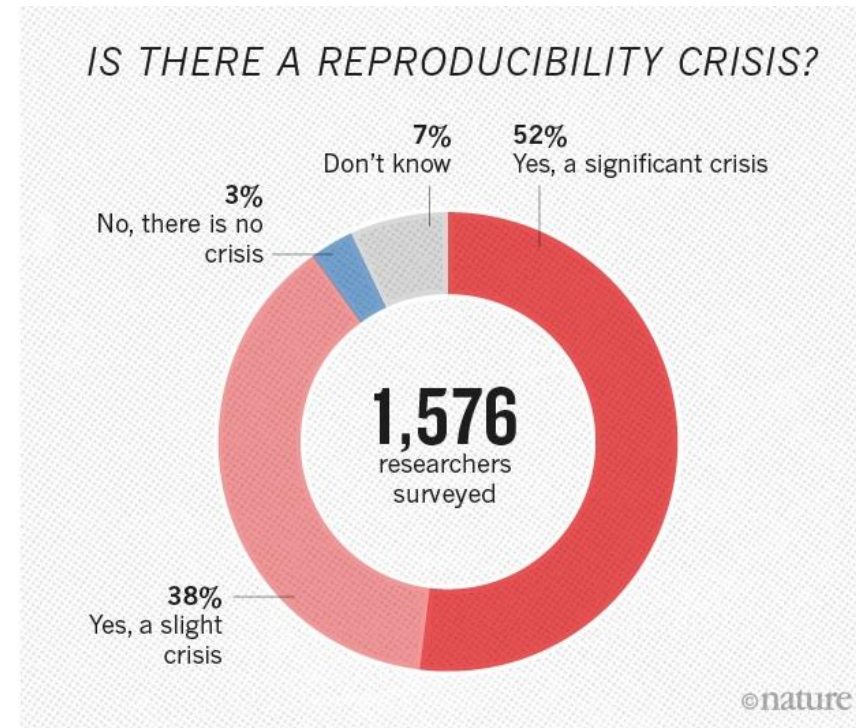
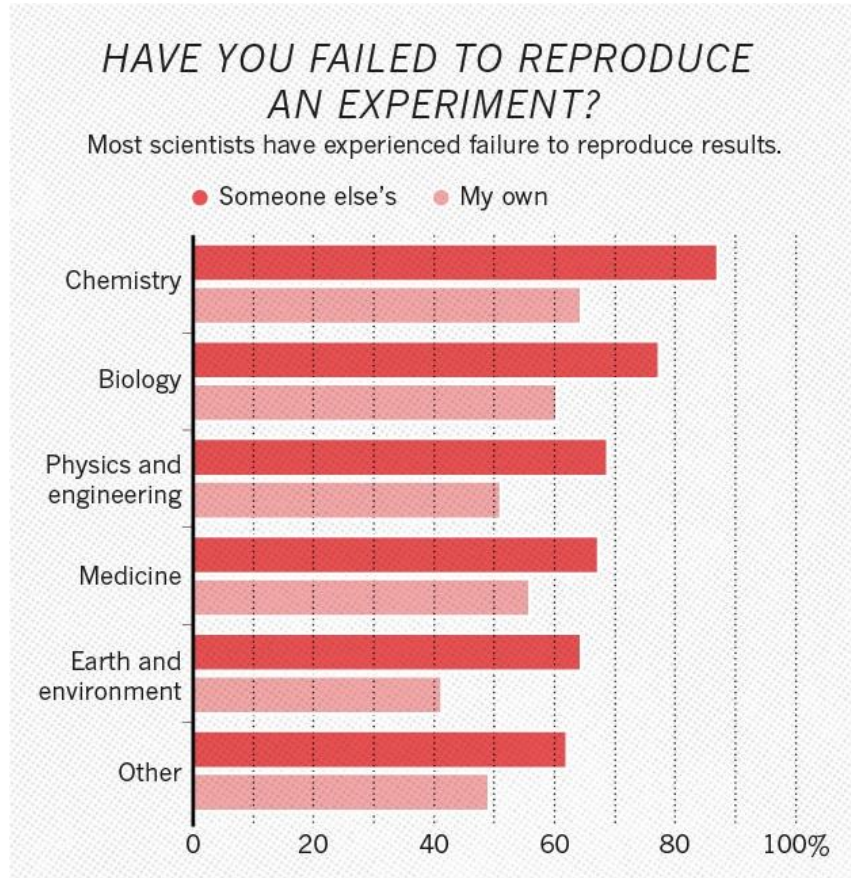


Setting the Scene



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

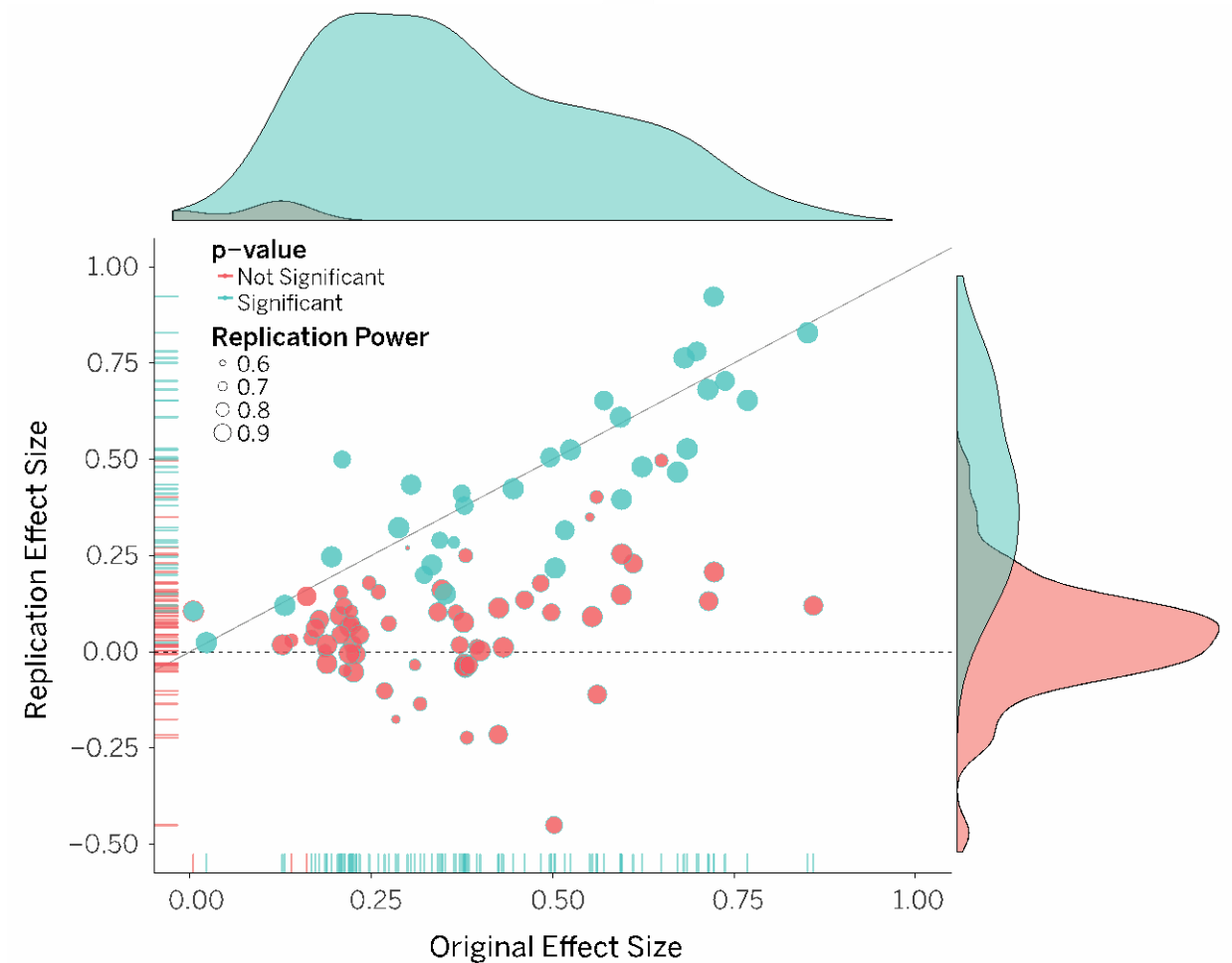
Open Science Collaboration^{*,†}

[†] 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

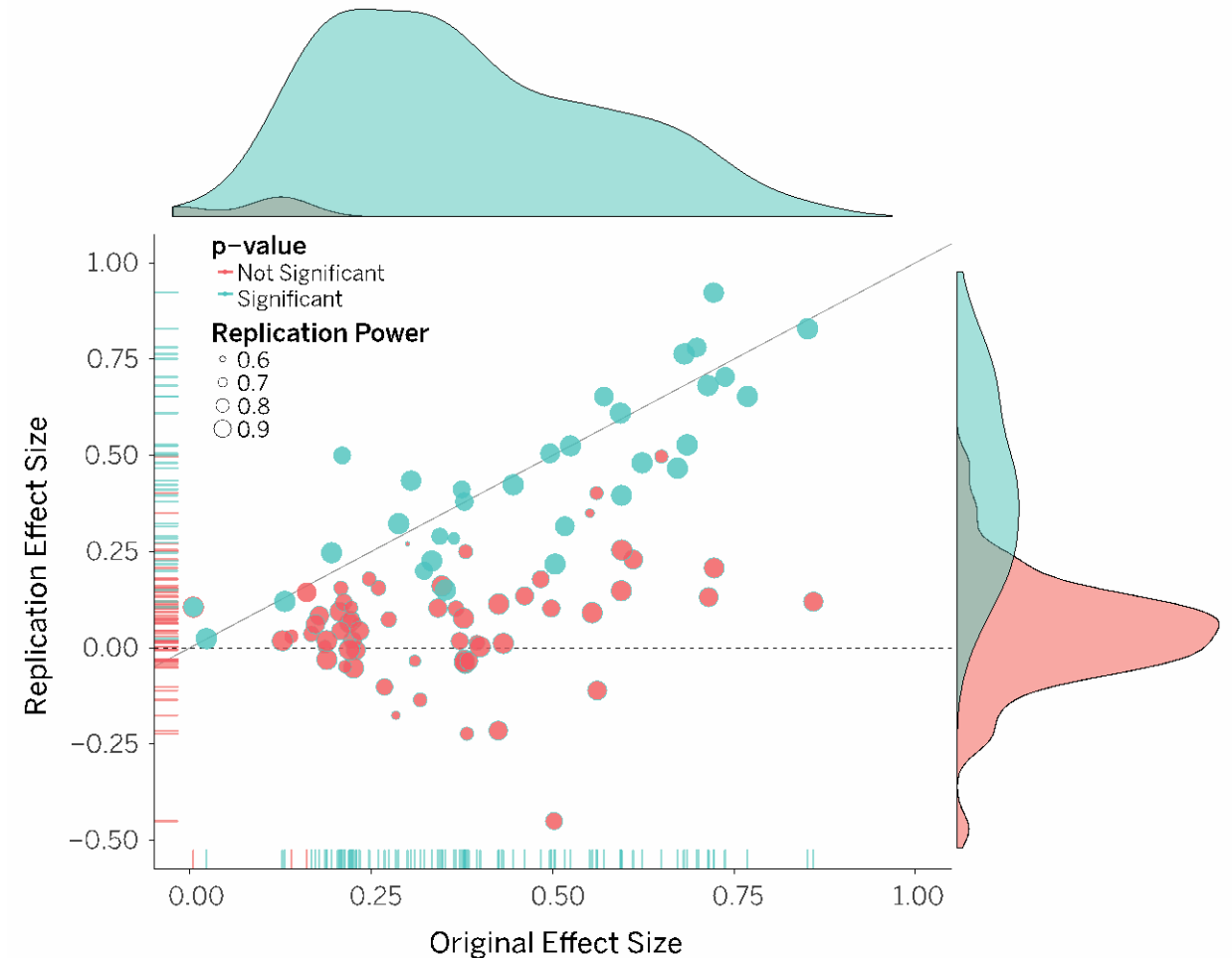
Open Science Collaboration^{*,†}

^{*} 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**



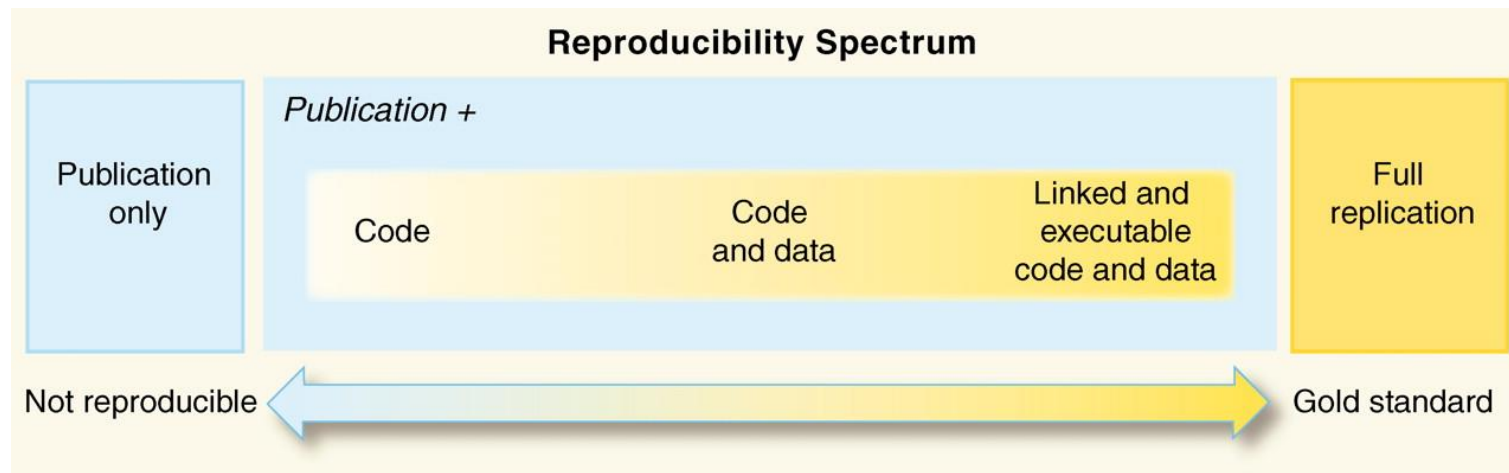
Reproducibility & Replicability in Science

Replication:

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

Reproducible research:

same data + same method = same results



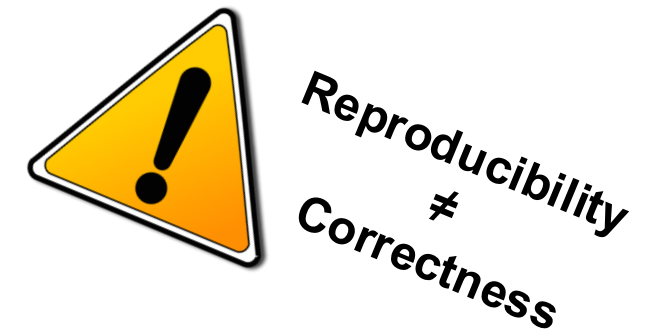
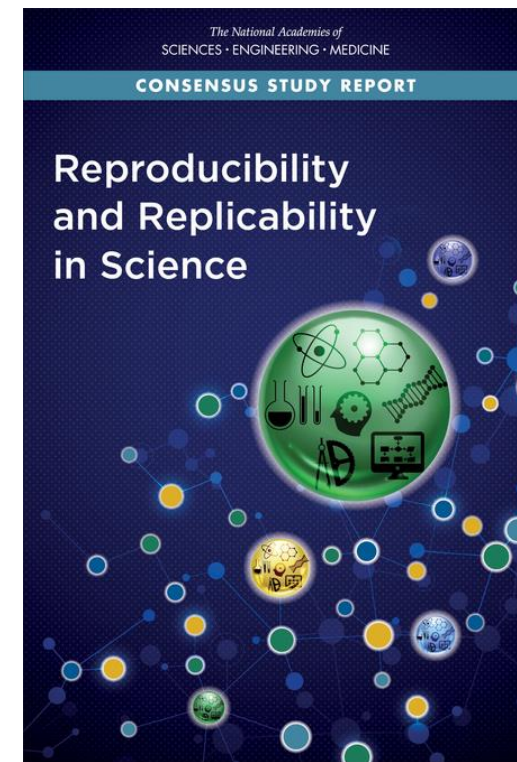
Peng (2011). [doi:10.1126/science.1213847](https://doi.org/10.1126/science.1213847)

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"... »

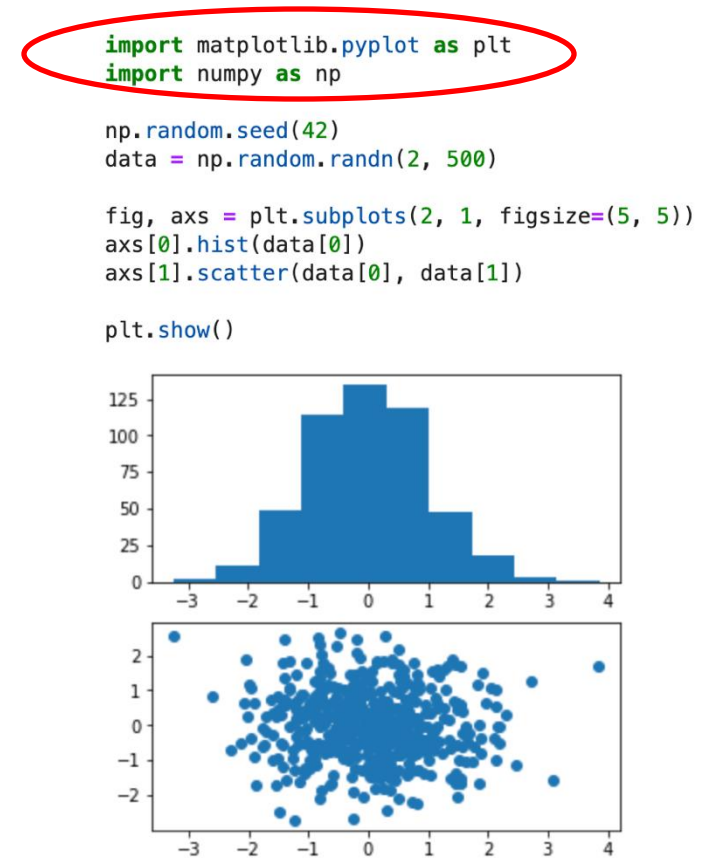
«To help ensure the reproducibility of computational results, researchers should **convey clear, specific, and complete information about any computational methods and data products that support their published results in order to enable other researchers to repeat the analysis**, unless such information 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.)



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__)

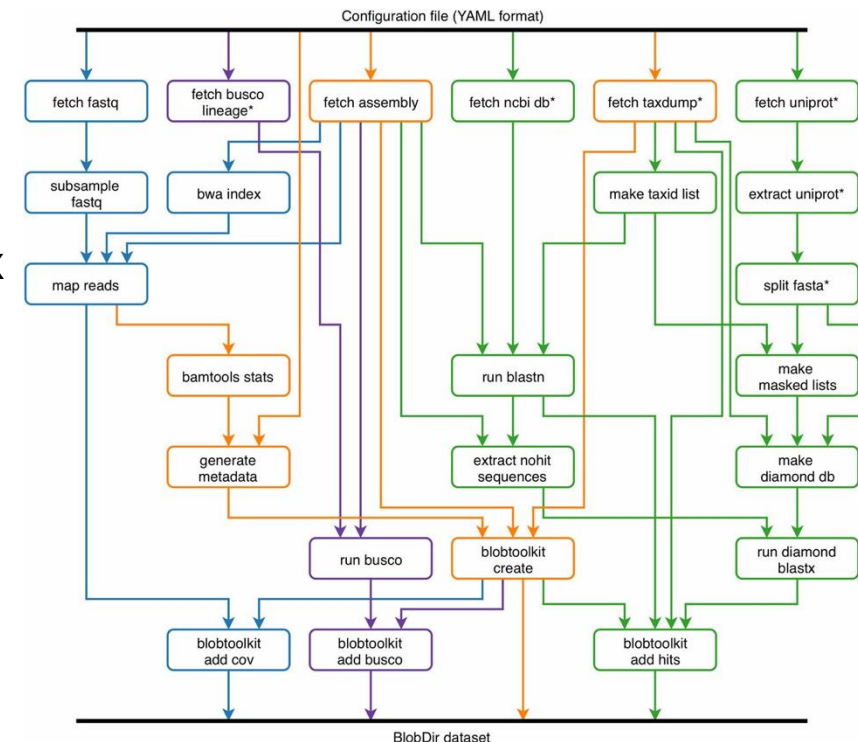
rng = np.random.default_rng(42)
rng.dirichlet((0.04, 0.03), 2)

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>

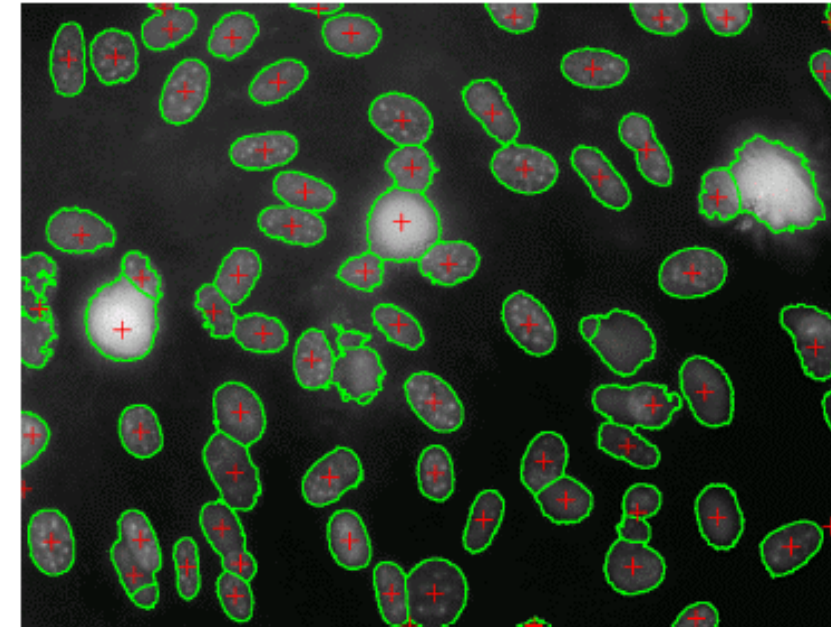
Computational Reproducibility: What can go wrong?

- 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**
- Code may be **incomprehensible** (complex, undocumented workflows)



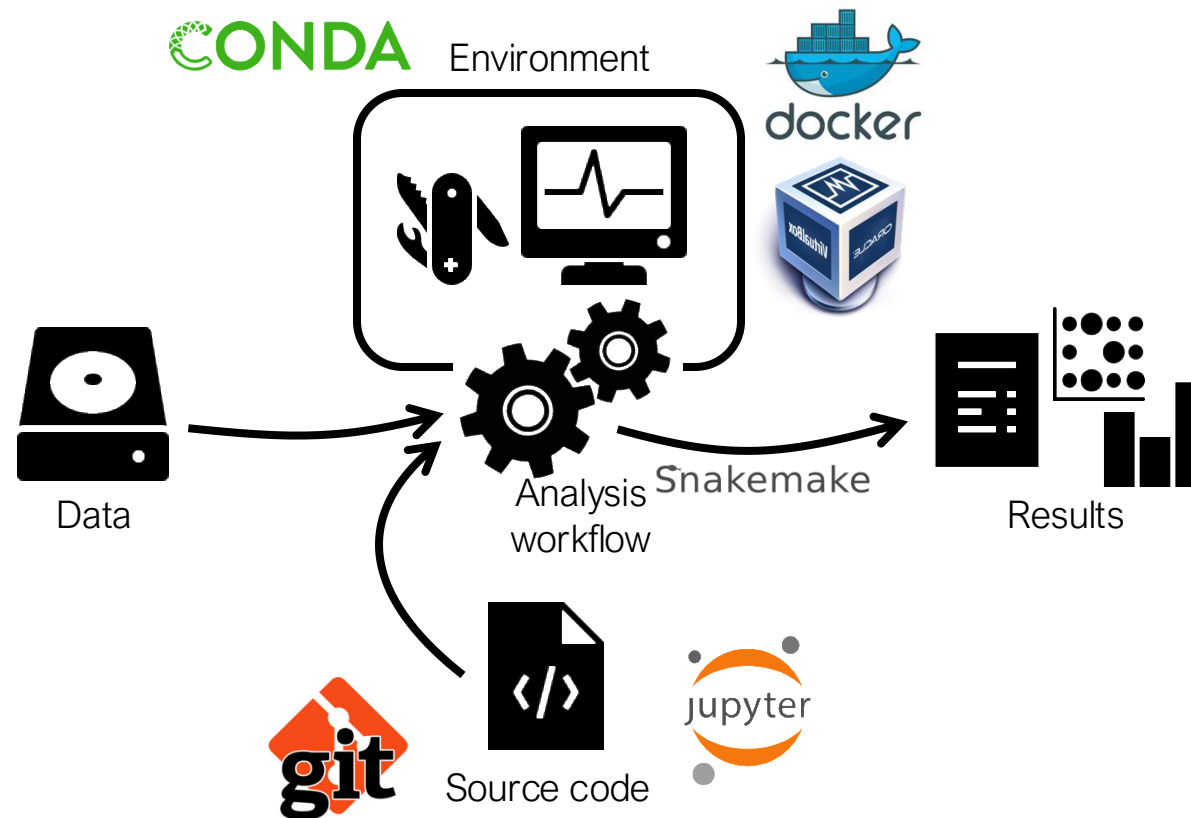
Computational Reproducibility: What can go wrong?

- 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**



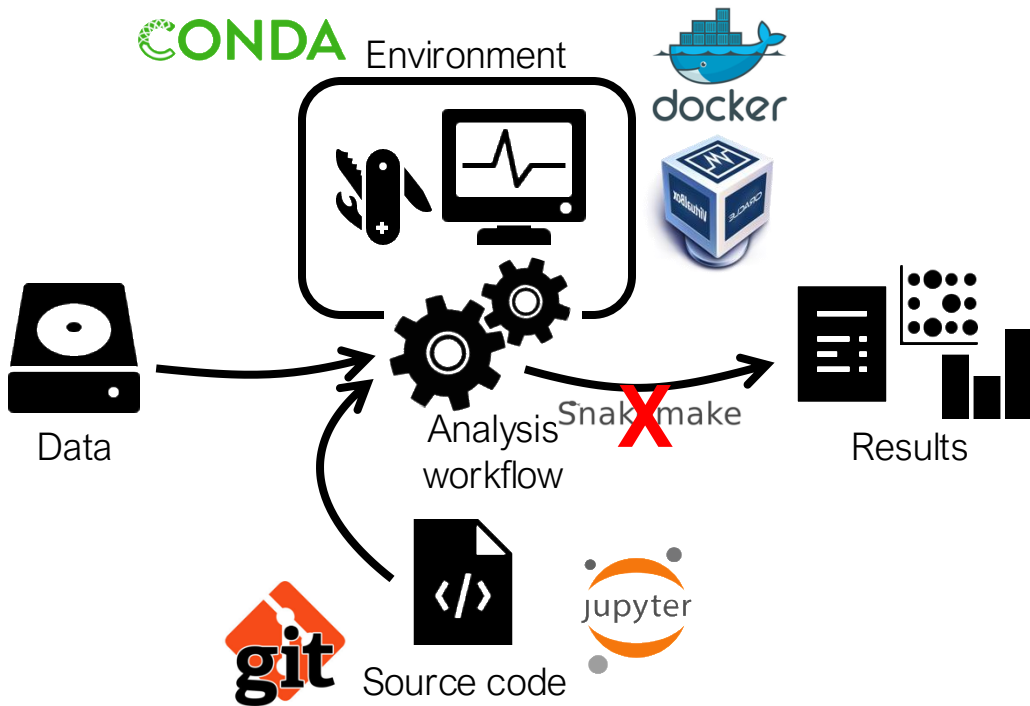
Computational Reproducibility: Pieces of the Puzzle

All parts of a computational analysis have to be reproducible!



Computational Reproducibility: Pieces of the Puzzle

What is covered in today's workshop?



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



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



Part 3 **Notebooks**
Document your exploratory analysis



Part 4 **Virtual machines / containers**
Make your project self-contained and distributable



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



Computational Reproducibility: Questions?



Tell us a bit about yourself



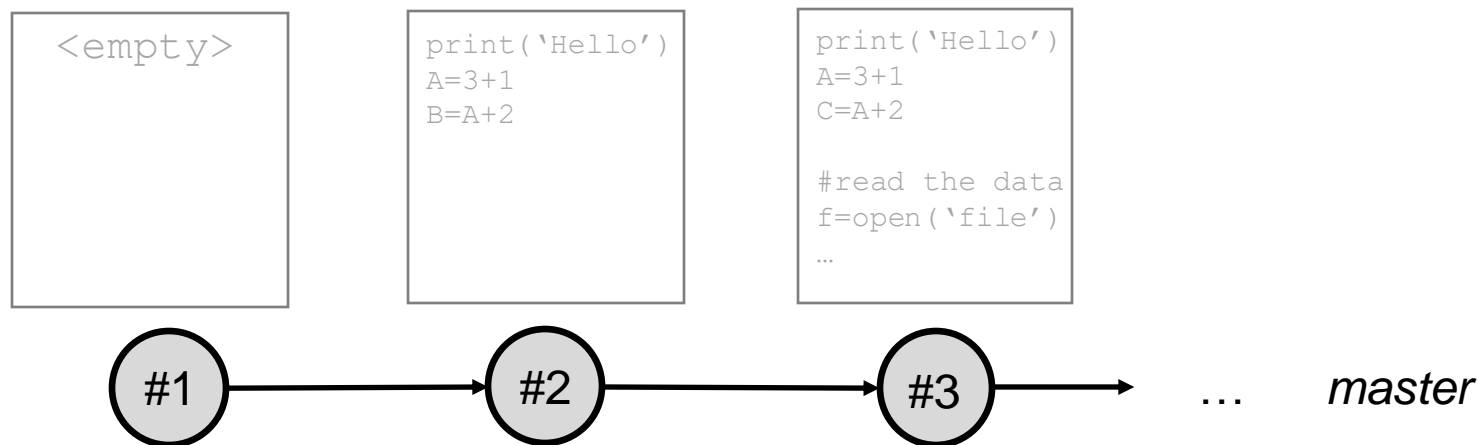
Managing your Source Code



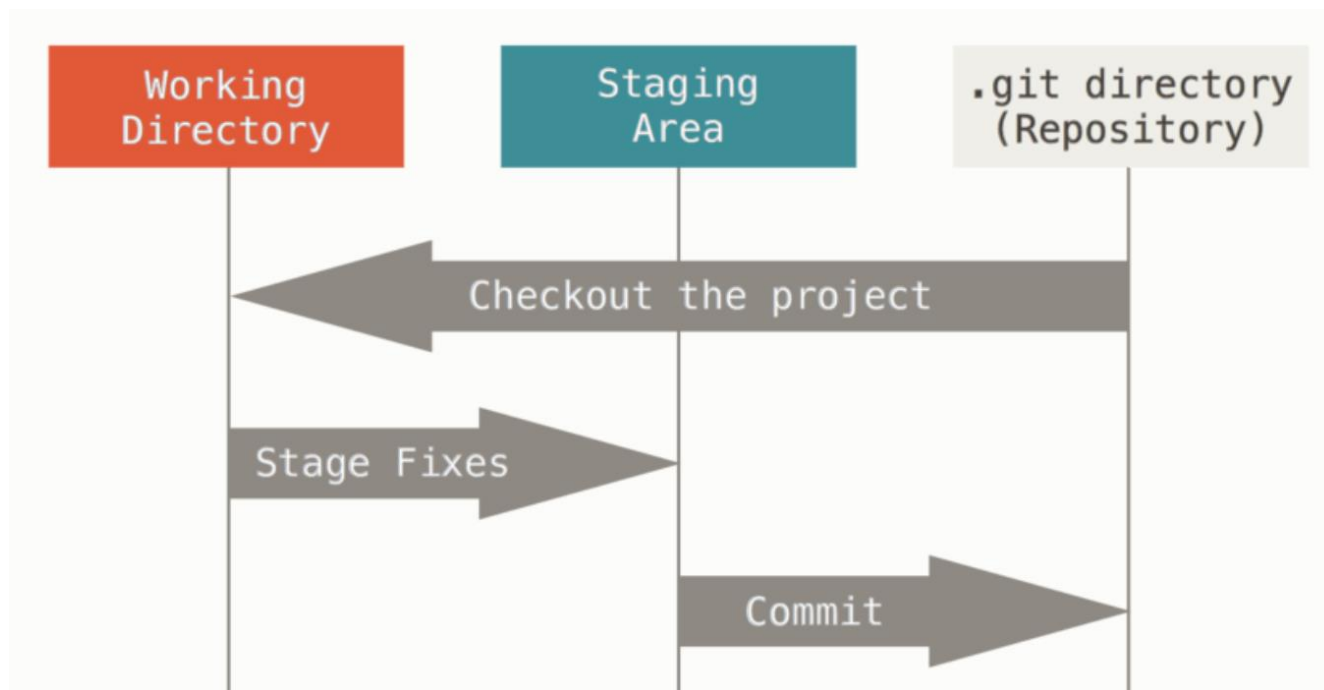
Code Management



- 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)
```

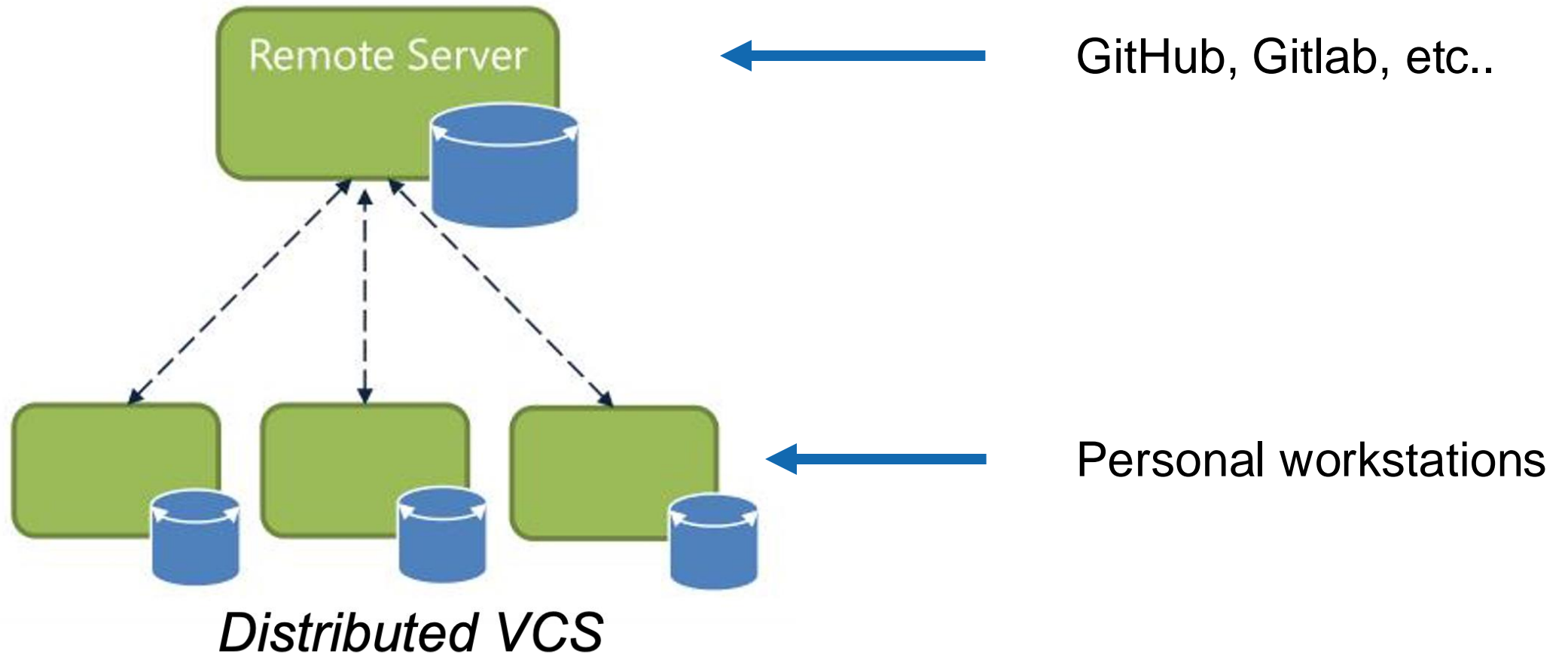
```
Author: Nadia Marounina <nmarounina@ethz.ch>
```

```
Date: Thu Mar 7 14:00:10 2024 +0100
```

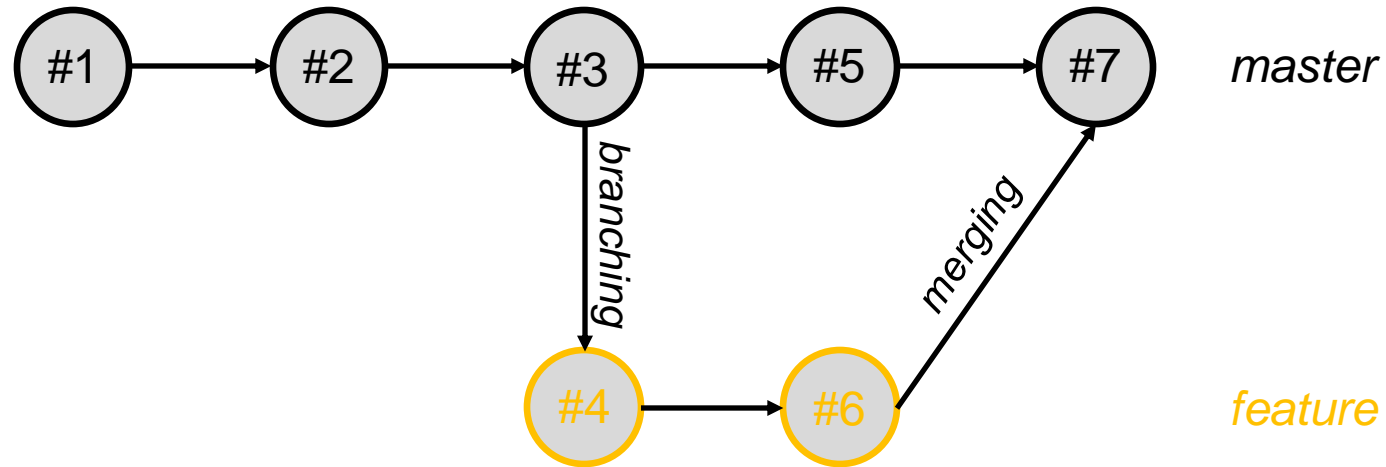
```
Initial commit
```

```
git_demo 14:00:20 >>
```

Git : How to share my code with others ?



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 committing 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



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 header includes the ETH zürich logo and navigation tabs for Projects, Groups, Activity, Milestones, and Snippets. A search bar is also present. A red banner at the top of the page contains a notification: 'To receive notifications about scheduled maintenance, please subscribe to the mailing-list gitlab-operations@sympa.ethz.ch. You can subscribe to the mailing-list at https://sympa.ethz.ch'. The main content area shows the project details for 'experimental-project-1' (Project ID: 6107). It includes a 'Clone' button, statistics for 7 commits, 1 branch, 0 tags, and 9.5 MB of files. A commit history table is displayed below, showing the most recent commit by Henry Luetcke. Below the commit history, there are several buttons for adding project features like README, CHANGELOG, CONTRIBUTING, Auto DevOps, and a Kubernetes cluster. A table at the bottom lists the files in the repository.

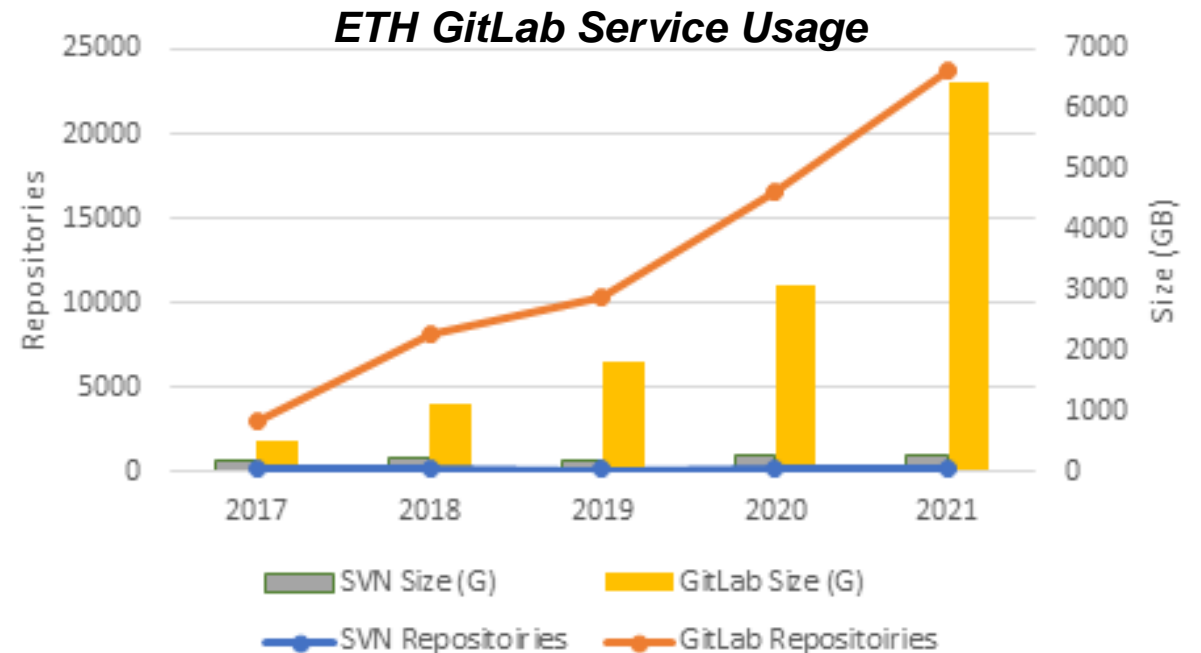
Name	Last commit	Last update
data	change image file size	4 months ago
.gitattributes	my first commit	5 months ago
analysis_code.py	change	4 months ago

<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.)
- 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

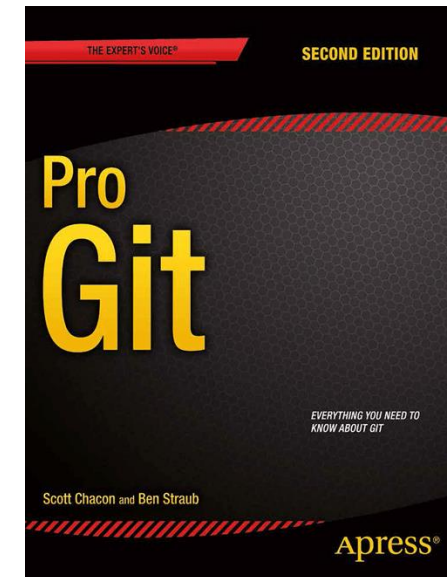


Recommendations for working with Git

- 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?



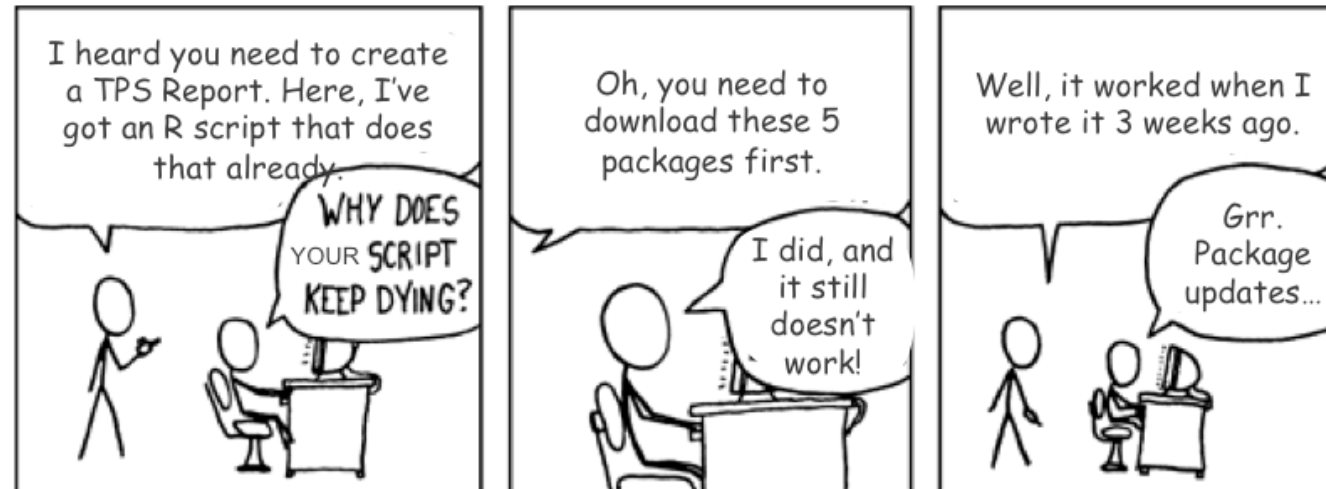
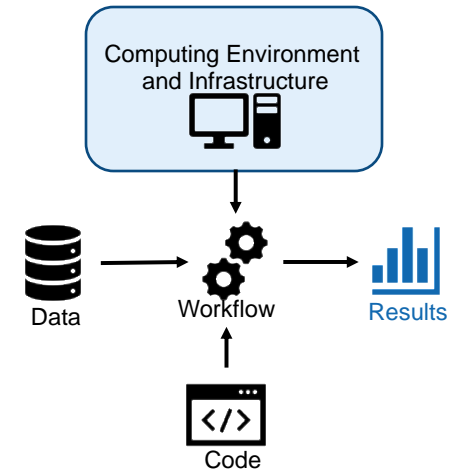
Managing Dependencies & Computing Environments



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

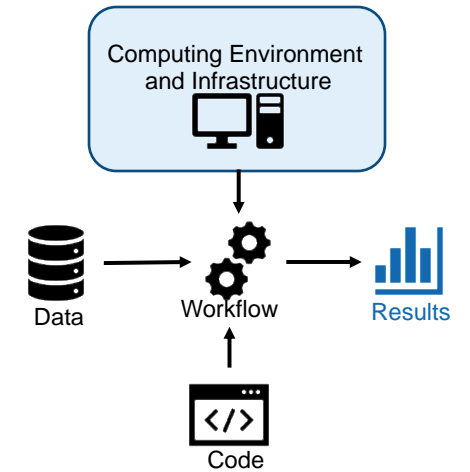
Solution:

Bundle your application and all dependencies

→ Environment Isolation & Dependency management

Tools:

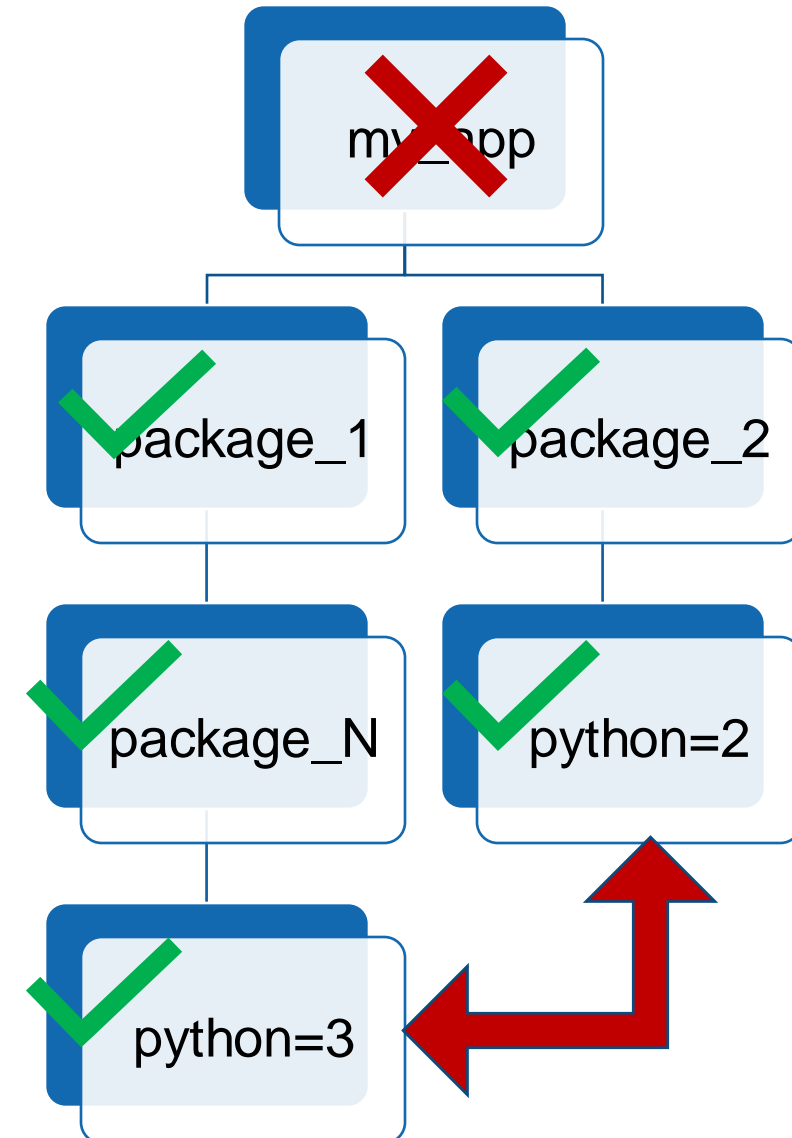
- Application / software level: Conda, pip, virtualenv, renv, Devbox
- Containerization: Docker
- Virtualization (Virtual Machine, VM): VirtualBox, VMware



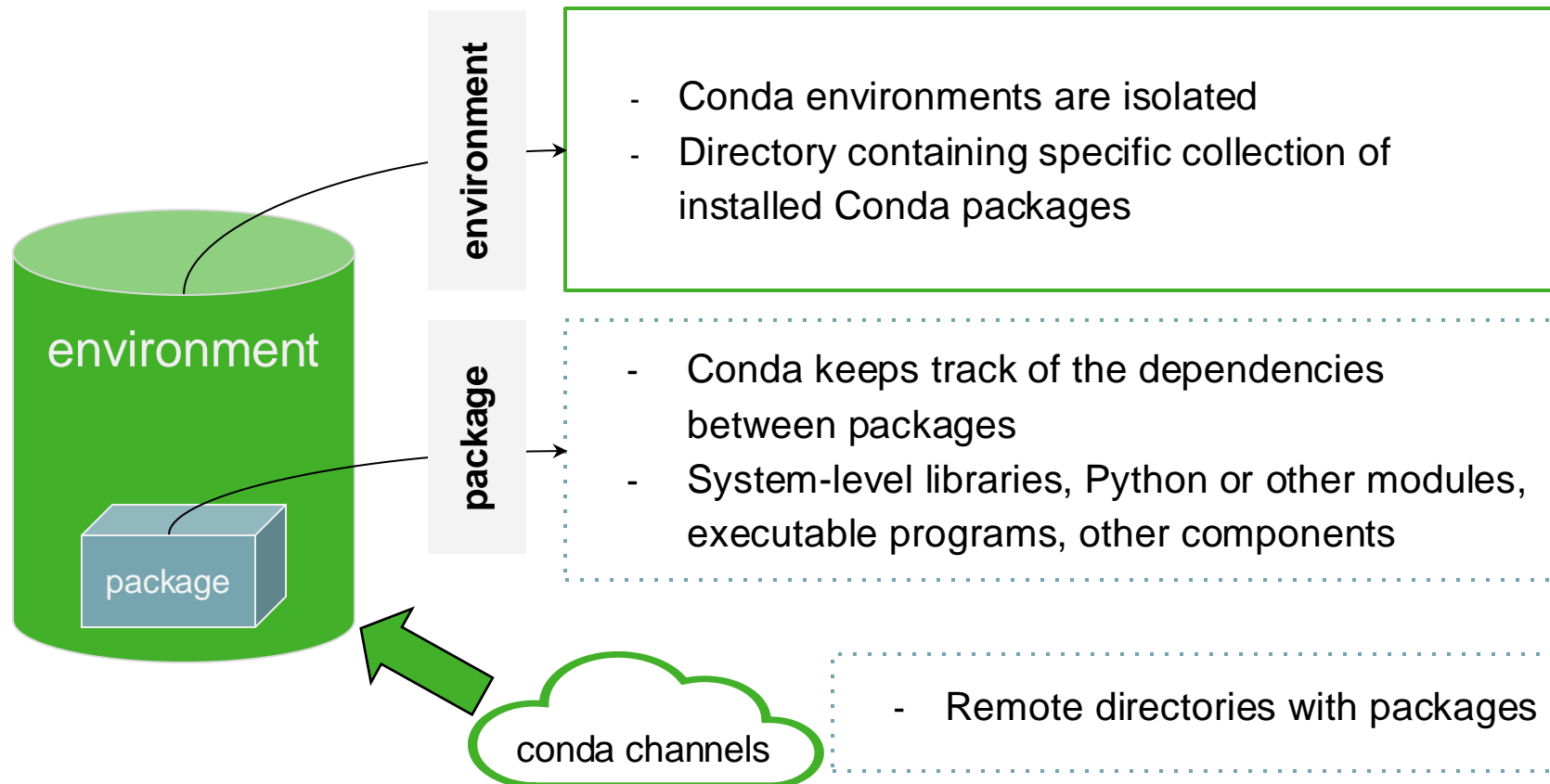
Reproducible Environment for R and Python



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

Conda automatically creates an environment file with packages and dependencies

Environment and Package Management Systems

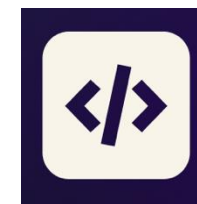
Language	Environment Management	Package Management	Comments
Python 2 (not supported)	virtualenv, conda	pip, conda	
Python 3	venv, virtualenv, pipenv poetry, conda	pip, pipenv, poetry, conda	only conda can install different Python versions (pyenv can be used)
R	renv, conda	renv, conda	only conda can install 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!

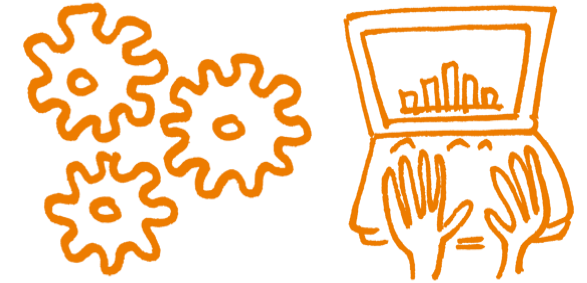


[pixi](#)



[Devbox](#)

Conda Hands-on Session



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



Home

Environments

Projects (beta)

Learning

Community

Documentation

Developer Blog

Feedback

Search Environments

root

snakes

1

2

Created

Channels

Update index...

Search Pac...Q

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
appsript	○		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

Create Clone Import Remove

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



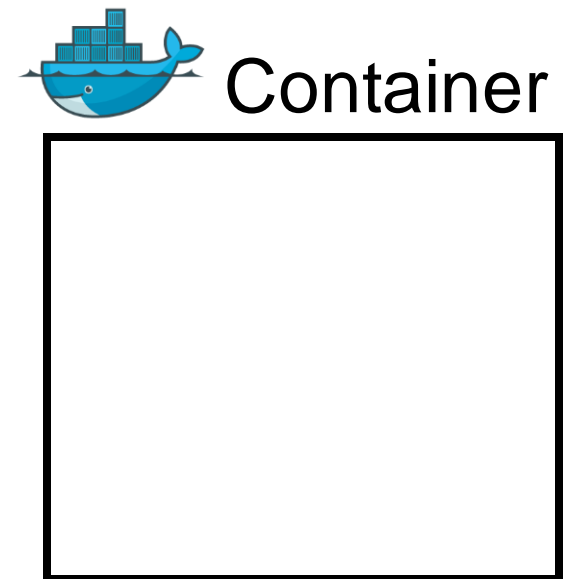
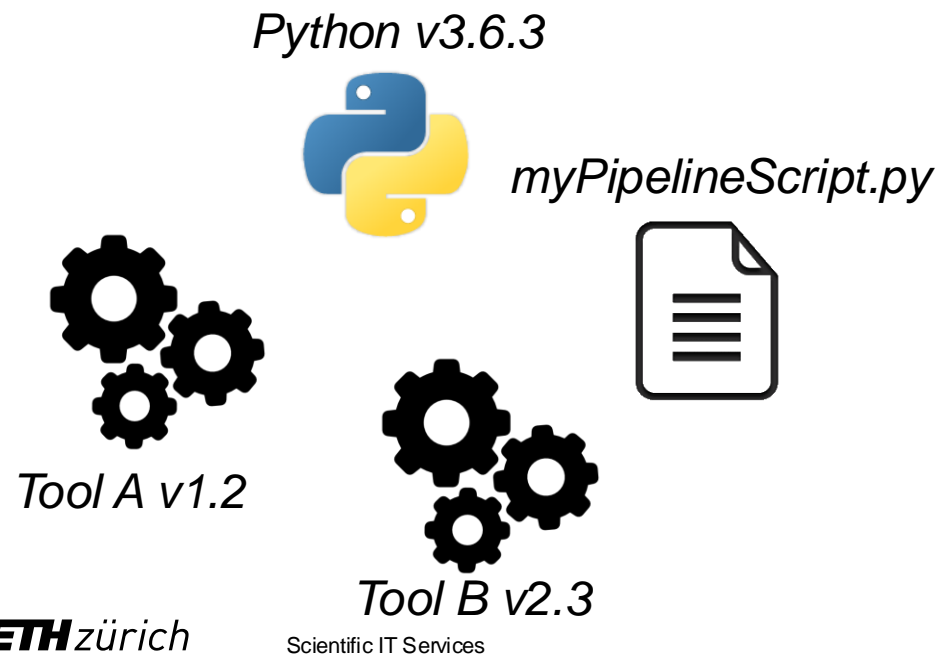
Source: <https://searchservirtualization.techtarget.com/definition/virtual-machine>

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

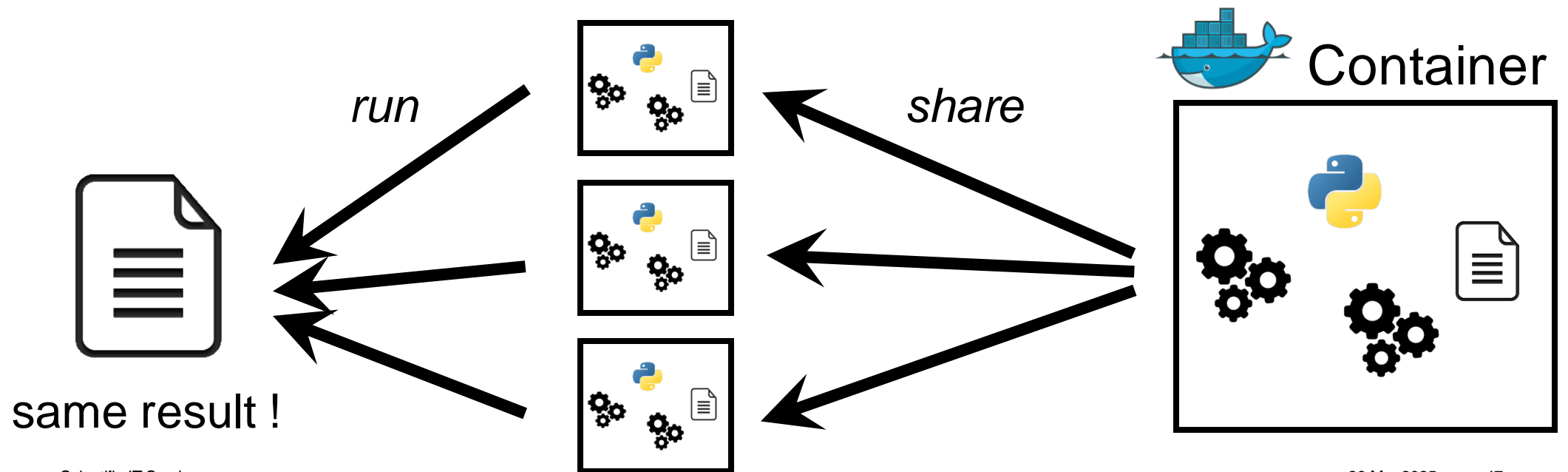
Reproducible Environment – Containerization

- **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**



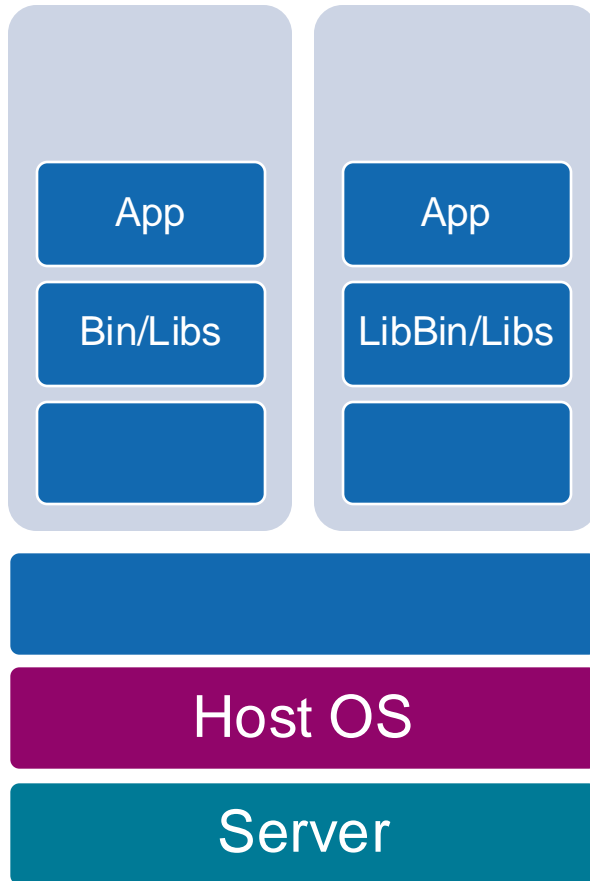
Reproducible Environment – Containerization

- **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**

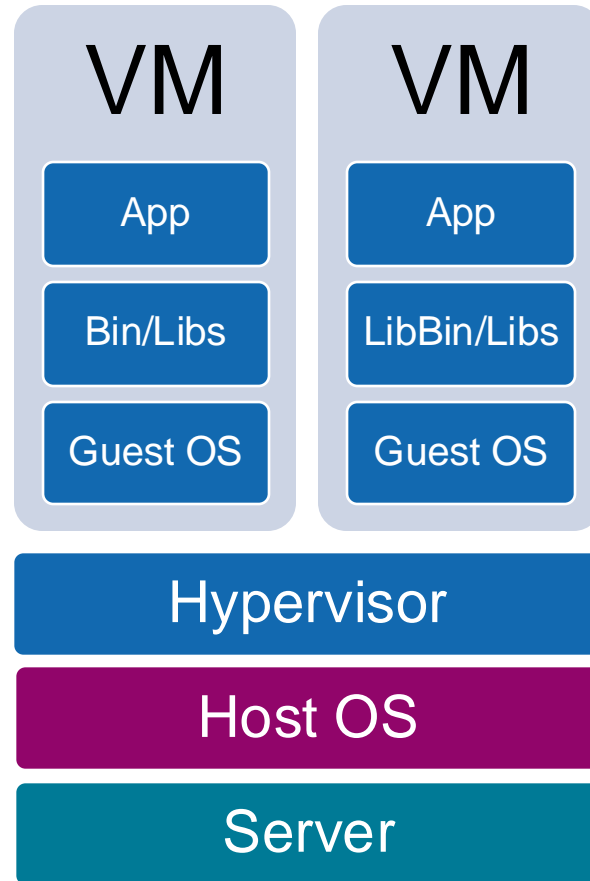


Bare Metal, Virtual Machine (VM) and Container (Docker)

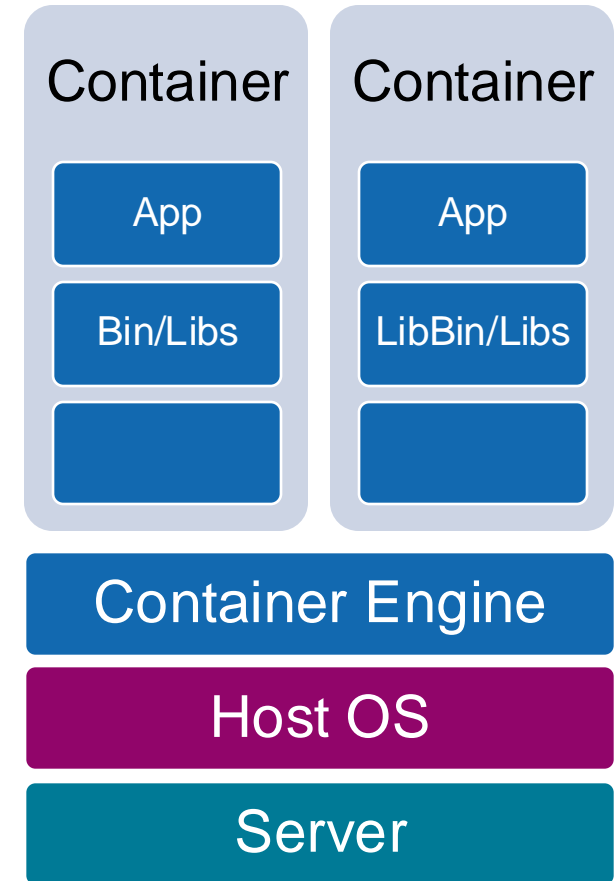
Bare Metal



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?





File Edit View Run Kernel Tabs Settings Help

Lorenz.ipynb x Python 3

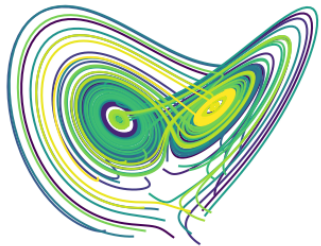
We explore the Lorenz system of differential equations:

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

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

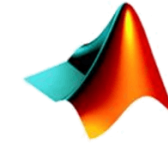
In [2]: `from lorenz import solve_lorenz`
`w=interactive(solve_lorenz,sigma=(0.0,50.0),rho=(0.0,50.0))`
`w`

sigma 10.00
beta 2.67
rho 28.00



lorenz.py x

```
6 def solve_lorenz(sigma=10.0, beta=8./3, rho=28.0):
7     """Plot a solution to the Lorenz differential equations."""
8
9     max_time = 4.0
10    N = 30
11
12    fig = plt.figure()
13    ax = fig.add_axes([0, 0, 1, 1], projection='3d')
14    ax.axis('off')
15
16    # prepare the axes limits
17    ax.set_xlim((-25, 25))
18    ax.set_ylim((-35, 35))
19    ax.set_zlim((5, 55))
20
21    def lorenz_deriv(x,y,z, t0, sigma=sigma, beta=beta, rho=rho):
22        """Compute the time-derivative of a Lorenz system."""
23        x, y, z = X,Y,Z
24        return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]
25
26    # Choose random starting points, uniformly distributed from -15 to 15
27    np.random.seed(1)
28    x0 = -15 + 30 * np.random.random((N, 3))
29
30    # Solve for the trajectories
31    t = np.linspace(0, max_time, int(250*max_time))
32    x_t = np.asarray([integrate.odeint(lorenz_deriv, x0i, t)
33                    for x0i in x0])
34
35    # choose a different color for each trajectory
36    colors = plt.cm.viridis(np.linspace(0, 1, N))
37
38    for i in range(N):
39        x, y, z = x_t[i, :, :].T
40        lines = ax.plot(x, y, z, '-', c=colors[i])
41        plt.setp(lines, linewidth=2)
42    angle = 104
43    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.)



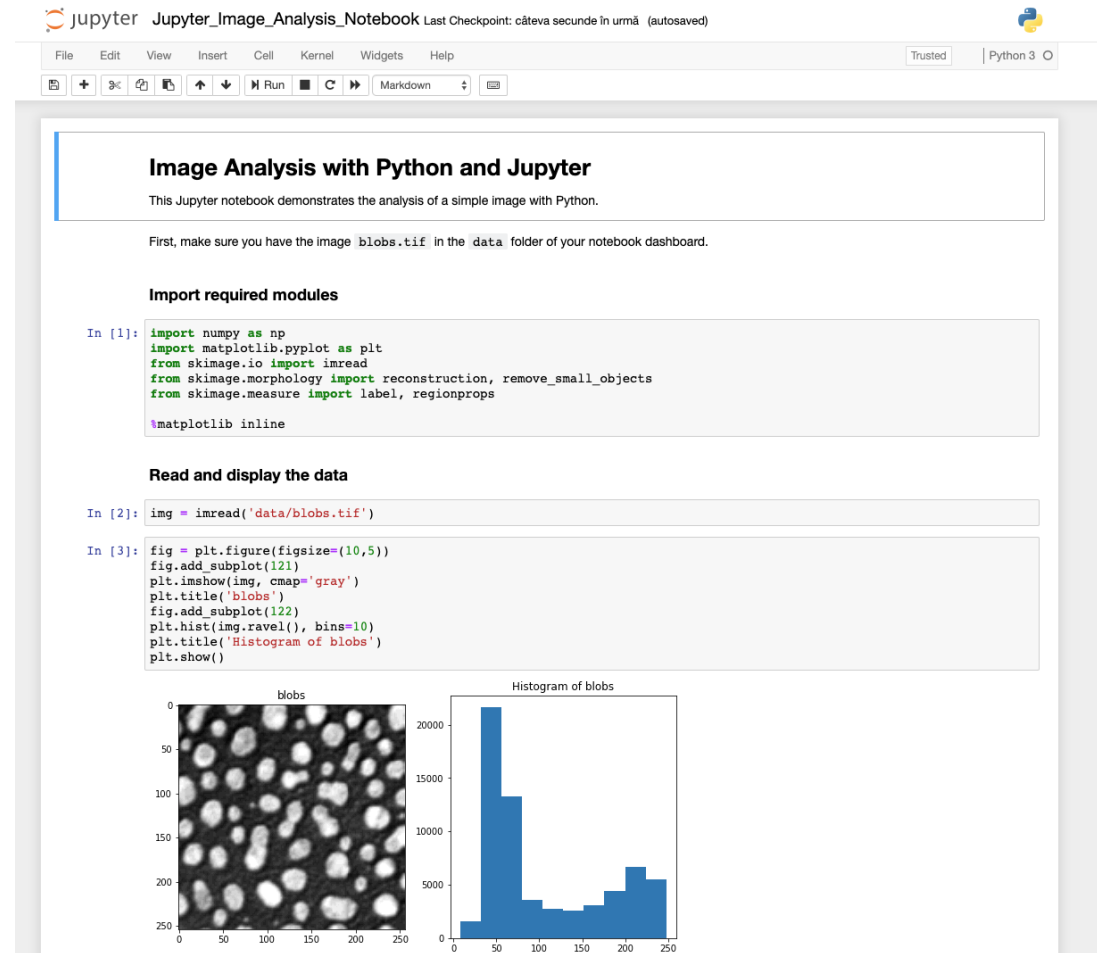
- Commercial
- Used in mathematical fields



- 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". The interface includes a top menu bar with options like File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu is a toolbar with icons for file operations and execution. The notebook content is as follows:

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

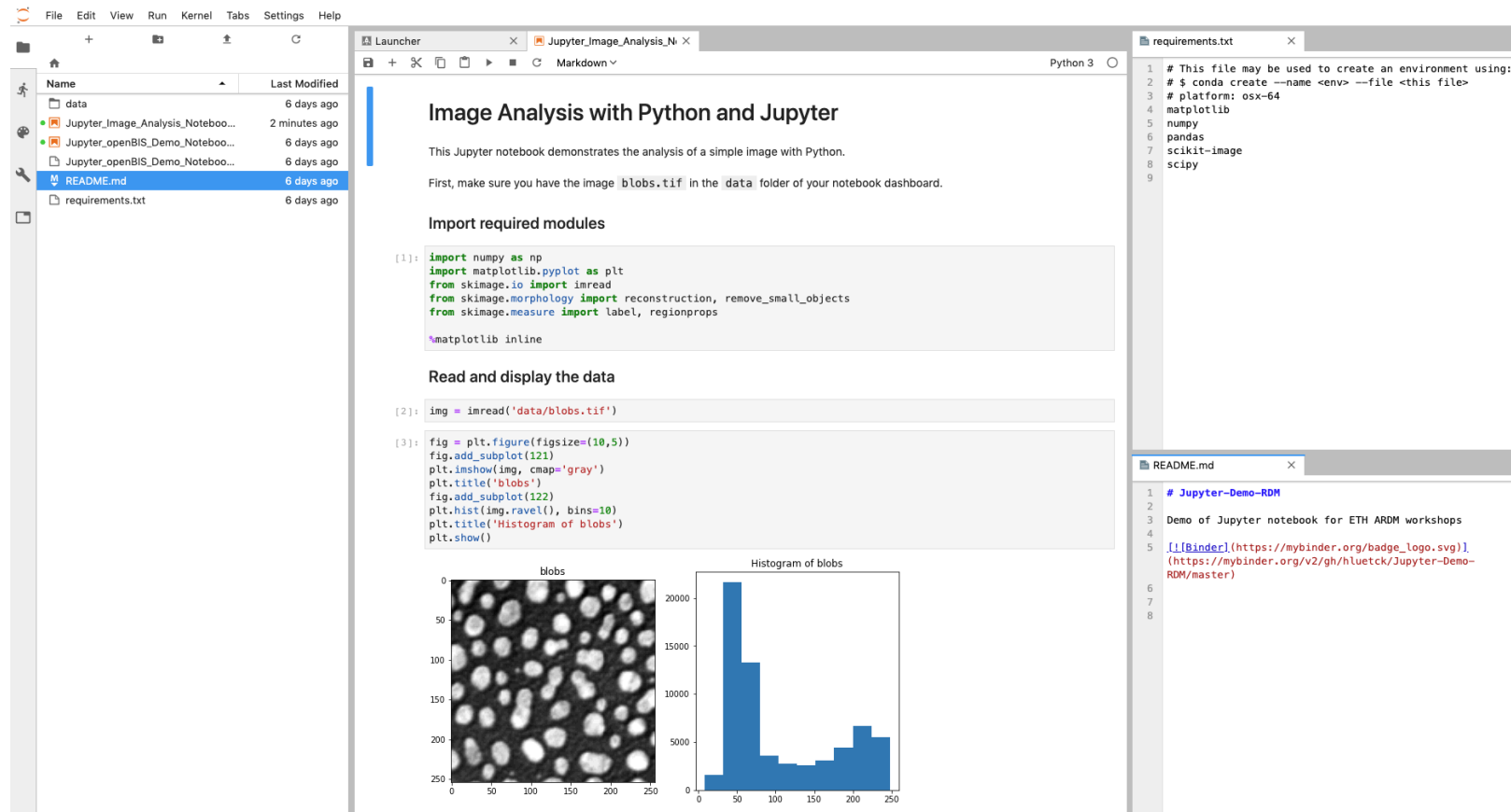
```
In [2]: img = imread('data/blobs.tif')

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 of the notebook shows two plots side-by-side. The left plot, titled "blobs", is a grayscale image of a 250x250 pixel area containing numerous small, bright, irregularly shaped objects (blobs) on a dark background. The right plot, titled "Histogram of blobs", is a histogram showing the distribution of pixel intensities. The x-axis represents pixel intensity from 0 to 250, and the y-axis represents the number of pixels, ranging from 0 to 20,000. The histogram shows a sharp peak at low intensity (around 25) and a smaller peak at high intensity (around 225).

Interactive Notebooks: Jupyter

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



The screenshot displays the JupyterLab interface with a notebook titled "Image Analysis with Python and Jupyter". 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

```
[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 output shows two plots:

- A grayscale image titled "blobs" showing a collection of white circular objects on a black background.
- A histogram titled "Histogram of blobs" showing the distribution of pixel intensities, with a peak around 50.

On the right side of the interface, there are two open 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/v2/gh/hluetck/Jupyter-Demo-RDM/master)
6
7
8
```

Interactive Notebooks: Jupyter

- **Jupyter notebook:** web-based interactive computational environment
- **JupyterLab:** web-based interactive development environment for notebooks, code, and data
- 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 ...)
- Integration with ETH scientific computing infrastructure
(see <https://jupyter.euler.hpc.ethz.ch/hub/>)
- **JupyterHub:** multi-user version of the notebook for research labs

Interactive Notebooks: Jupyter [demo]

Gravitational wave physics

gwastro / o2-bbh-pe

Watch 8 Star 4 Fork 1

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights

Branch: master o2-bbh-pe / data_release_o2_bbh_pe.ipynb

Find file Copy path

soumide1102 Update contour plots adding boundary bias code, add skymap notebook

f00120b on 26 Apr

1 contributor

11.6 MB

Download History

Posterior samples of the parameters of binary black holes from Advanced LIGO, Virgo's second observing run

Soumi De¹, Christopher M. Bower², Collin D. Capano^{3,4}, Alexander H. Nitz^{3,4}, Duncan A. Brown¹

¹Department of Physics, Syracuse University, Syracuse, NY 13244, USA

²Los Alamos National Laboratory, Los Alamos, NM 87545, USA

³Albert-Einstein-Institut, Max-Planck-Institut for Gravitationsphysik, D-30167 Hannover, Germany

⁴Leibniz Universitat Hannover, D-30167, Hannover, Germany

License



This work is licensed under a <https://creativecommons.org/licenses/by/4.0/deed.ast>.

To plot Fig. 2 of the paper : mass ratio---effective spin ($q - \chi_{\text{eff}}$) posteriors

```
In [36]: fig, ax = pyplot.subplots(figsize=(9.5, 9.5))

handles = []
colors = itertools.cycle(["C{}".format(i) for i in range(10)])

ndim = 2
# read samples
params = [None] * ndim
params[0] = "(primary_mass(mass1, mass2))/(secondary_mass(mass1, mass2))"
params[1] = "chi_eff_from_spherical(mass1, mass2, spin1_a, spin1_polar, spin2_a, spin2_polar)"

for filename, label in zip(files, labels):
    with InferenceFile(filename, "r") as fp:
        # Read samples from the inference output file
        samples = fp.read_samples(params)
        color = colors.next()

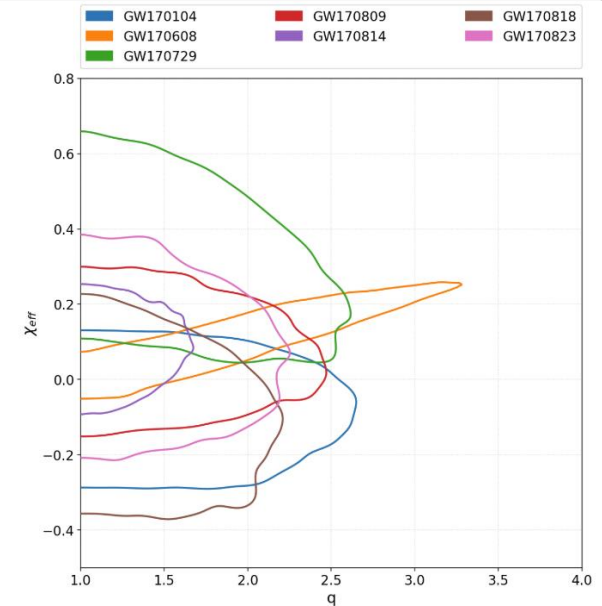
        # Bounds on the domain for evaluating KDE
        xlow_bc, xhigh_bc = 1.0, None
        ylow_bc, yhigh_bc = -1.0, 1.0

        # Make density plot
        create_contour_plot(params[0], params[1], samples, xlow_bc, xhigh_bc,
                            ylow_bc, yhigh_bc, fig=fig, ax=ax, plot_contours=True,
                            xmax=4.0, ymin=-0.5, ymax=0.8, contour_color=color)

        handles.append(patches.Patch(color=color, label=label))

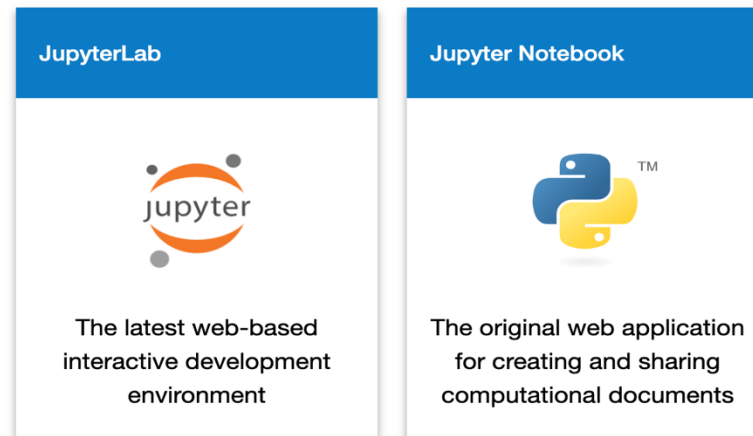
pyplot.xlabel(r"q", fontsize=16)
pyplot.ylabel(r"\chi_{eff}", fontsize=16)
pyplot.xlim(right=4.0)
pyplot.ylim(-0.5, 0.8)
pyplot.tick_params(axis='both', which='major', labelsize=16)
pyplot.legend(bbox_to_anchor=(0, 1, 1, 0), loc='bottom',
              handles=handles, labelmode="expand", borderaxesprops={'color': 'black'})

fig.show()
```



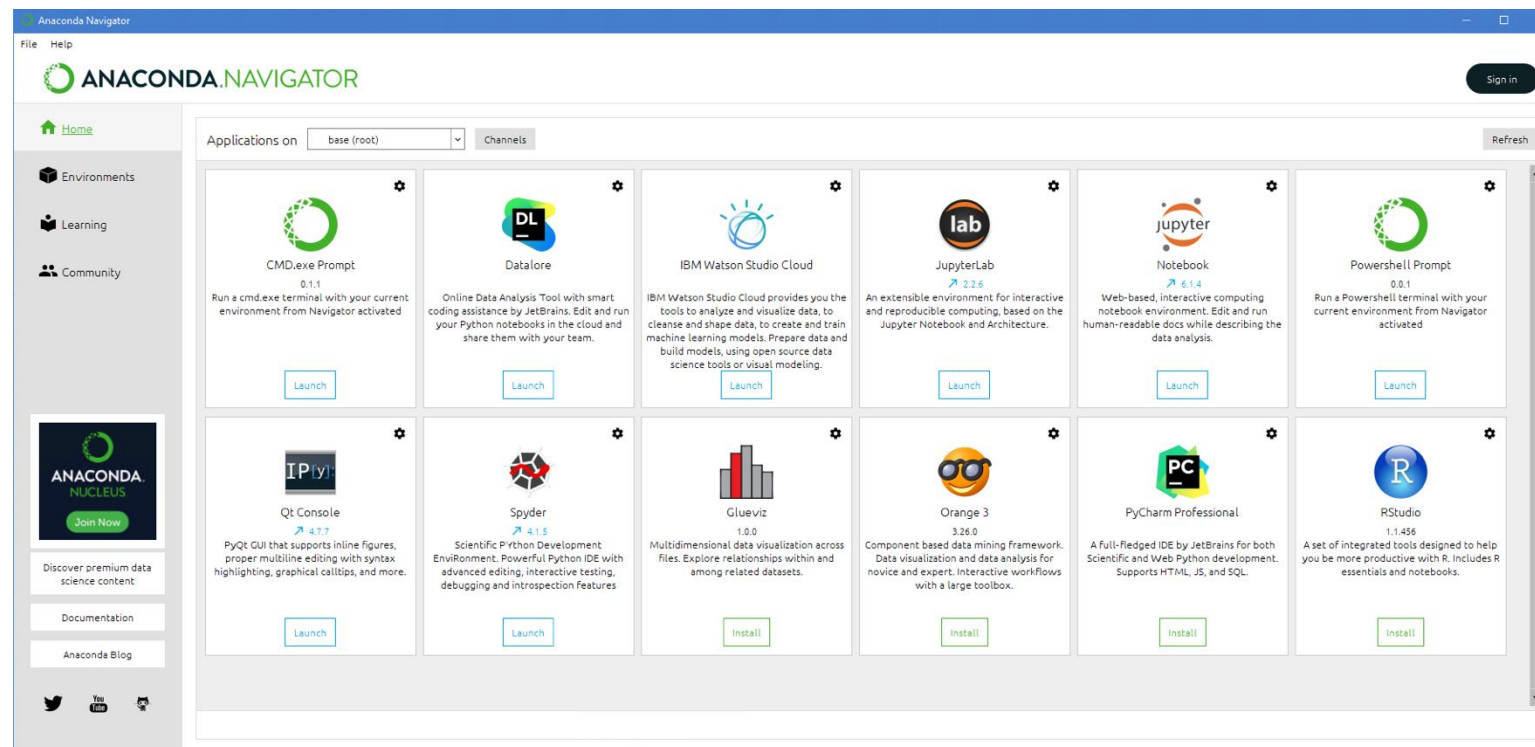
Options for running Jupyter

- Local installation on your computer
- Dedicated JupyterHub server (e.g. running on virtual machine in the cloud or on Euler)
- 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**

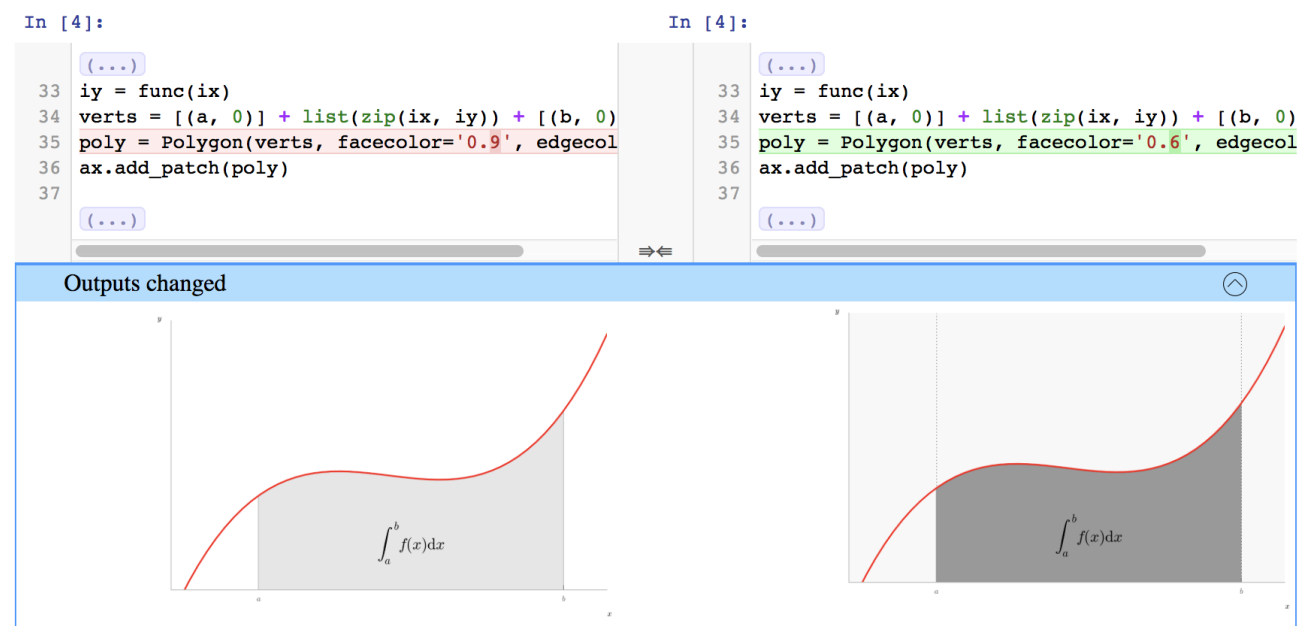
- Version control of even moderately complex NBs is challenging
- Tracking NB history is harder than for traditional source code
- Some tools may help (e.g. [nbdime](#), [JupyterText](#))

```
$ diff a.ipynb b.ipynb
76,77d75
<     "plt.rc('axes', grid=False)\n",
<     "plt.rc('axes', facecolor='white')\n",
90c88
<     "image/png": "iVBORw0KGgoAAAANSUhEUgAABLkAAAMQCAYAAADLj7dLAAAABHNCSVQICAgIFAhki
AAAAAlwSFlz\nAAAWJQAAFiUBSVIk8AAAIABJREFUeJzsvXeYZFd57b12h0maPNJII2LG0aCAkEBCFgozIxBAp
LY\n1waDyDZg8MX+zMU2F4Mx1x8PwWAwxmBjg4yNi2BfQMa20iiAQFkIjXKWRtJIE3tSz3TXuX+8vV2n\nqyucv
N+9z/o9zzynprvq1D6nqqtqr1prbRNFEQghhBCCCCGEEEEII8Zkh1wMghBCCCCGEEEEIIISQv\nFLkIIYQQQgghhB
BCiPdQ5CKEEEEIIYQQQggh3kORixBCCCCGEEEEIIYR4D0UuQgghhBCCCCGEE0I9\nFLkIIYQQQgghhBBCiPdQ5CK
EEEEIIYQQQggh3kORixBCCCCGEEEEIIYR4D0UuQgghhBCCCCGEE0I9\nFLkIIYQQQgghhBBCiPdQ5CKEEEEIIYQQ
Qggh3kORixBCCCCGEEEEIIYR4D0UuQgghhBCCCCGEE0I9\nFLkIIYQQQjzEGHOJMaZljPmo67EkZWq8D7keByGEE
ELChCIXIYQQQirDGPOmKaFj3BhzkMNx/H/G\nnmG3GmP/pagwFEbkeQJUYY75gjNlijHmD67EQQgghRB8UuQgghB
BSJe+DCDMjAH7L4TjeAmA+gLc5\nHEMRGNcDqJi3AVgI4DddD4QQQggh+qDIRQghhJBKMMacCuBMAFsg4sy7jTH
DjobzZwBuBvBxR/dP\nnsVERADcC+LTrgRBCCCFEHxS5CCGEEFIVH4C4uP4SIlQcBOD1LgYSRVEz iqIXR1H0frf3
T7IRRdFf\nnRlH0K1EUXe96LIQQQgjRB0UuQgghhJSOMWYpgP8BoAXg7wH8HcTN9Tsux0UIIYQQQsKBIhchhBBC\
nguBdA0YAuDyKoscBfByAl gBnGwQe73PkhBCCCCkCChyEUTTTaPUjD5GUjTf0Pxcik7eDMP3n65C\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, especially with “classical” git
 - Some jupyter-targeted tools may help (e.g. [nbdime](#))
- **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 : has not been possible [until recently](#). Must be done in JupyterHub or cloud.
- **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)

renku

Sessions Help Login

Connecting the research ecosystem

The research ecosystem is fragmented.
Renku is where it comes together.

Data, Code, and Compute all under one roof.

Try it out Create an account


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 search and search for *eth-rdm-reproducible-analysis-workshop*

Renku Dashboard - Henry Luetcke

Projects [+ Create a new project](#)


Pinned projects

 **ETH RDM Reproducible Analysis Workshop** hluetcke/eth-rdm-re... [Start](#) [▼](#)

[Project](#) [Public](#) [Henry Luetcke](#)

Updated 19 hours ago

Recently visited projects

 **ETH RDM Workshop Spring 2024** hluetcke/eth-rdm-workshop-spri... [Start](#) [▼](#)

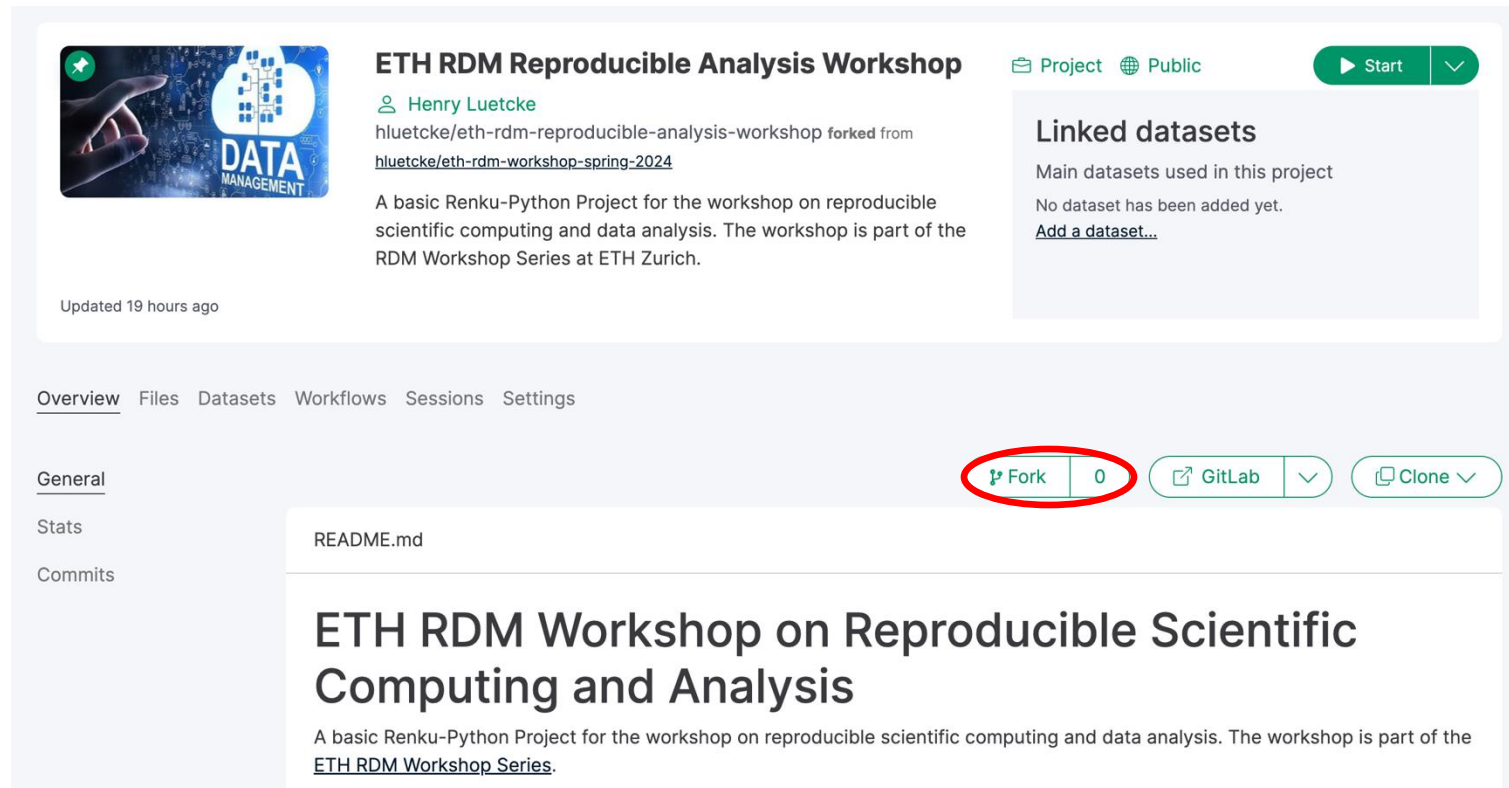
[Project](#) [Public](#) [Henry Luetcke](#)

Updated 19 hours ago

[View all my Projects](#)

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 search and search for *eth-rdm-reproducible-analysis-workshop*
- Select the project called *eth-rdm-reproducible-analysis-workshop* and fork it to your account



The screenshot shows the Renkulab interface for a project titled "ETH RDM Reproducible Analysis Workshop". The project is owned by Henry Luetcke and is public. It is a basic Renku-Python project for a workshop on reproducible scientific computing and data analysis. The project page includes a "Fork" button, which is circled in red, and a "Clone" button. The "Fork" button shows 0 forks. The project description mentions it is part of the RDM Workshop Series at ETH Zurich. The page also features a "Linked datasets" section and a navigation menu with options like Overview, Files, Datasets, Workflows, Sessions, and Settings.

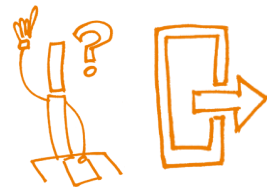
Reproducible Computing Platforms: *renkulab.io*

- 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)

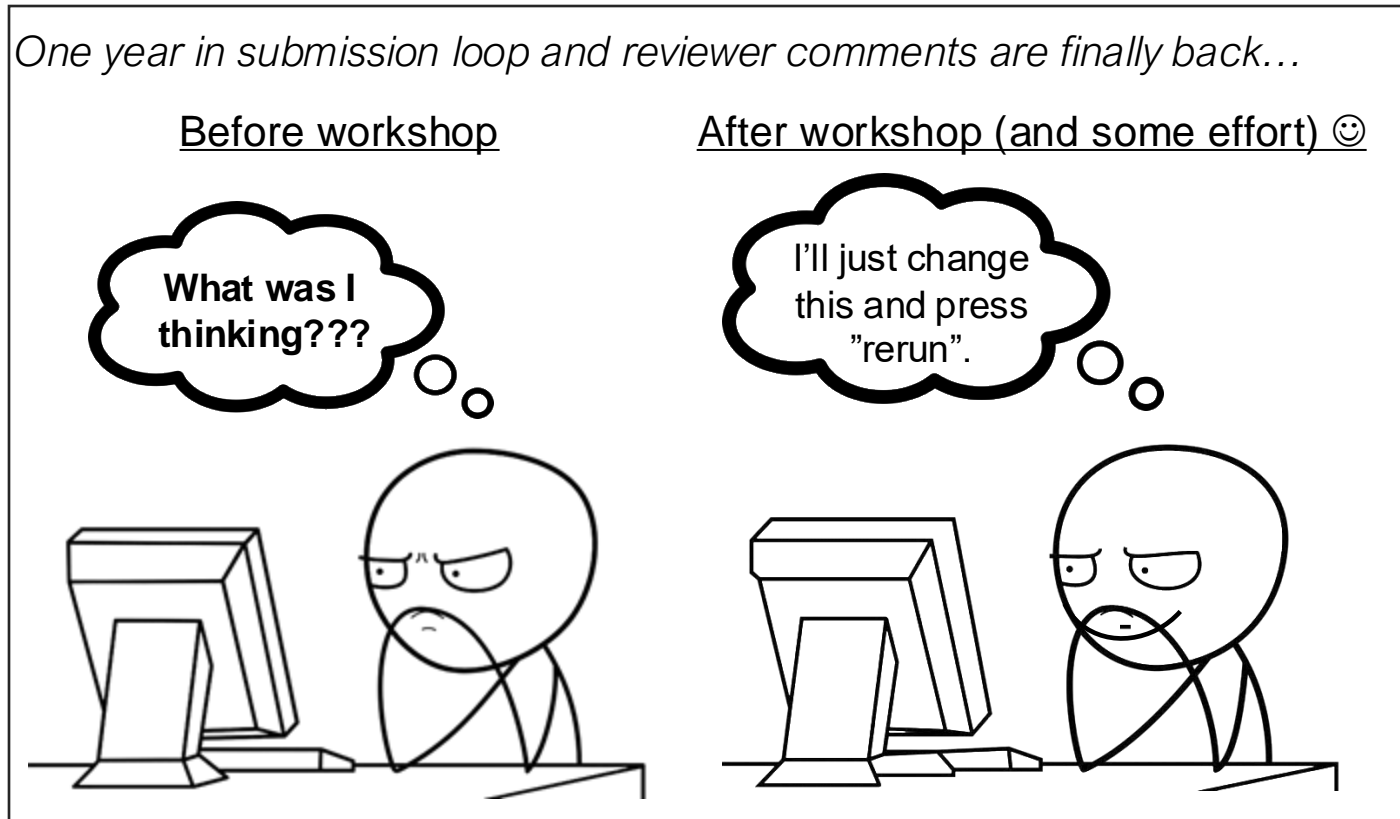
The screenshot displays a project page on the Renku platform. At the top, there is a project card for 'ETH RDM Reproducible Analysis Workshop' by Henry Luetcke, which is a forked project. The card includes a 'Start' button and a 'Linked datasets' section. Below the card, a navigation bar contains tabs for 'Overview', 'Files', 'Datasets' (marked with a red box and '1'), 'Workflows', 'Sessions' (marked with a red box and '2'), and 'Settings'. On the right side, there are buttons for 'Fork', '0', 'GitLab' (marked with a red box and '3'), and 'Clone'. The main content area shows a 'README.md' file with the title 'ETH RDM Workshop on Reproducible Scientific Computing and Analysis' and a description of the project.

- **Note:** Renku is currently undergoing a major version transition from 1.0 to 2.0 (beta)
 - See the [Renku Community Portal](#) for details

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

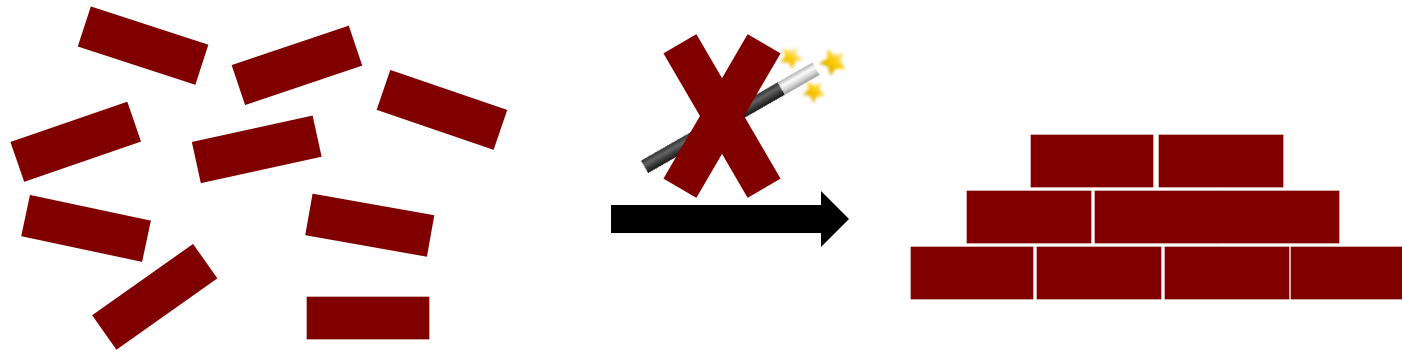
- Easier to rerun experiments and analysis
- Closer interaction between collaborators
- Much of the manuscript "writes itself"

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/c/scientificcomputing>



Any final questions on what we have discussed this morning?



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

