FAIR_bioinfo : Open Science and FAIR principles in a bioinformatics project

How to make a bioinformatics project more reproducible

C. Hernandez[1] T. Denecker[2] J. Seiler[2] G. Le Corguillé[2]
C. Toffano-Nioche[1]

[1]Institute for Integrative Biology of the Cell (I2BC)
UMR 9198, Université Paris-Sud, CNRS, CEA
91190 - Gif-sur-Yvette, France

[2]IFB Core Cluster taskforce

June 2021

General information

Practical information:

- Dates: June 28th - 30th
- Location: Institut des Systèmes Complexes, 113 rue Nationale, 75013-Paris
- Courses: 9:00 to 17:30
- Meal: 12:30-14:00
- Pauses: 10:30-11:00 + 15:30-16:00
- 2 days of courses + 1 day of course building

Round table:

- Teachers
- Learners

Ressources:

-
- GitLab
- LaTeX

Training schedule

Day 1:

- Introduction to reproducibility
- History management (3 Practical Sessions, git, GitHub)
- Control your development environment (1 PS, CONDA)
- Encapsulation (2 PS, docker)

Day 2:

- Workflow (2 PS, SNAKEMAKE)
- Traceability with notebooks (2 PS, jupyter,)
- IFB resources (2 PS, slurm,)
- Sharing and disseminating (GitHub, zenodo)
- Conclusion

Day 3:

- Empowerment and improvement of resources

Table of contents

1. Introduction to reproducibility
2. History management
3. Control your development environment
4. Workflow
5. Tracability with Notebook
6. IFB resources
7. Sharing and dissemination
8. Conclusion
   - Let's take a step back
9. 3rd Day

Conclusion

Current schedule

Day 1:

- Introduction to FAIR_bioinfo
- History management (git, GitHub)
- Environment management (CONDA, docker)

Day 2:

- Workflow (SNAKEMAKE)
- Traceability with notebooks (jupyter, markdown)
- IFB resources (S, slurm)
- Sharing and disseminating (GitHub, zenodo)

Let's take a step back.

FAIR_bioinfo

Findable

Easy to find protocols (GitHub, GitHub Pages) with DOI (zenodo)

Accessible

Open source (GitHub, docker, CONDA, ...)

Interoperable

Think "workflow" (SNAKEMAKE + docker / CONDA) locally or on servers (S, slurm)

Reusable

Replayable protocols (jupyter, markdown) in virtual environments (docker / CONDA)

A virtuous cycle

FAIR raw data

+

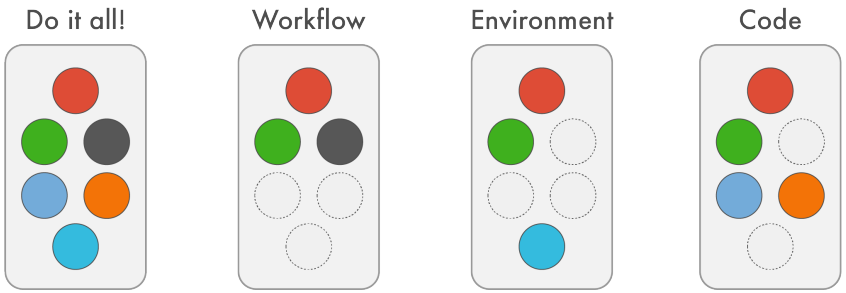
FAIR_bioinfo scripts/protocols

=

FAIR processed data

Swedish similar tutorial

From the NBIS – National Bioinformatics Infrastructure Sweden



nbis-reproducible-research.readthedocs.io/en/latest

Reproducibility checklist[2]

- Code avoid workflows based on point-and-click interfaces (eg. Excel), enshrine computations and data manipulation in code
- Document how code works, define parameters and computational environment required: comments, notebooks and README
- Record key parameters (eg. the 'seed' values of a random-number generator)
- Test functions using positive and negative control data sets, run those tests throughout development
- Guide with master script (eg. 'run.sh') that downloads data sets and executes workflow
- Archive with long-term stability services such as Zenodo, Figshare and Software Heritage (GitHub is impermanent online repository)

[2] Nature

Reproducibility checklist[3]

- Track the project's history with a version-control tools (eg. Git). Note (tag) which version you used to create each result
- Package with ready-to-use computational environments using containerization tools (eg. Docker, Singularity), web services (Code Ocean, Gigantum, Binder) or virtual-environment managers (Conda)
- Simplify and avoid niche or hard-to-install third-party code libraries
- Verify your code's portability by running it in a range of computing environments
- Automate the test of your code with continuous-integration services (eg. Travis CI)

[3] Nature

Adding Tests

Unit test: test a part of the code

```
## module 1
sum <- function(x, y){
    return (x+y)
}

# Unit test
sum(2,2) == 4
```

```
## module 2
power <- function(x, y){
    return (x**y)
}

# Unit test
power(2,2) == 4
```

Functional test: test all the code

```
# Functional test
power(sum(2,2),2) == 16
```

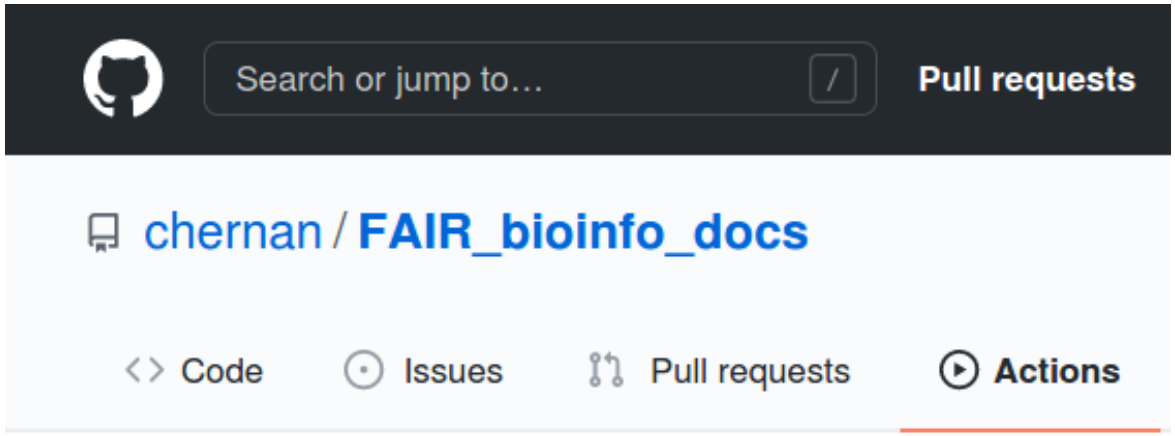
Continuous integration

Automated verification each time the source code is modified that the modifications do not produce:

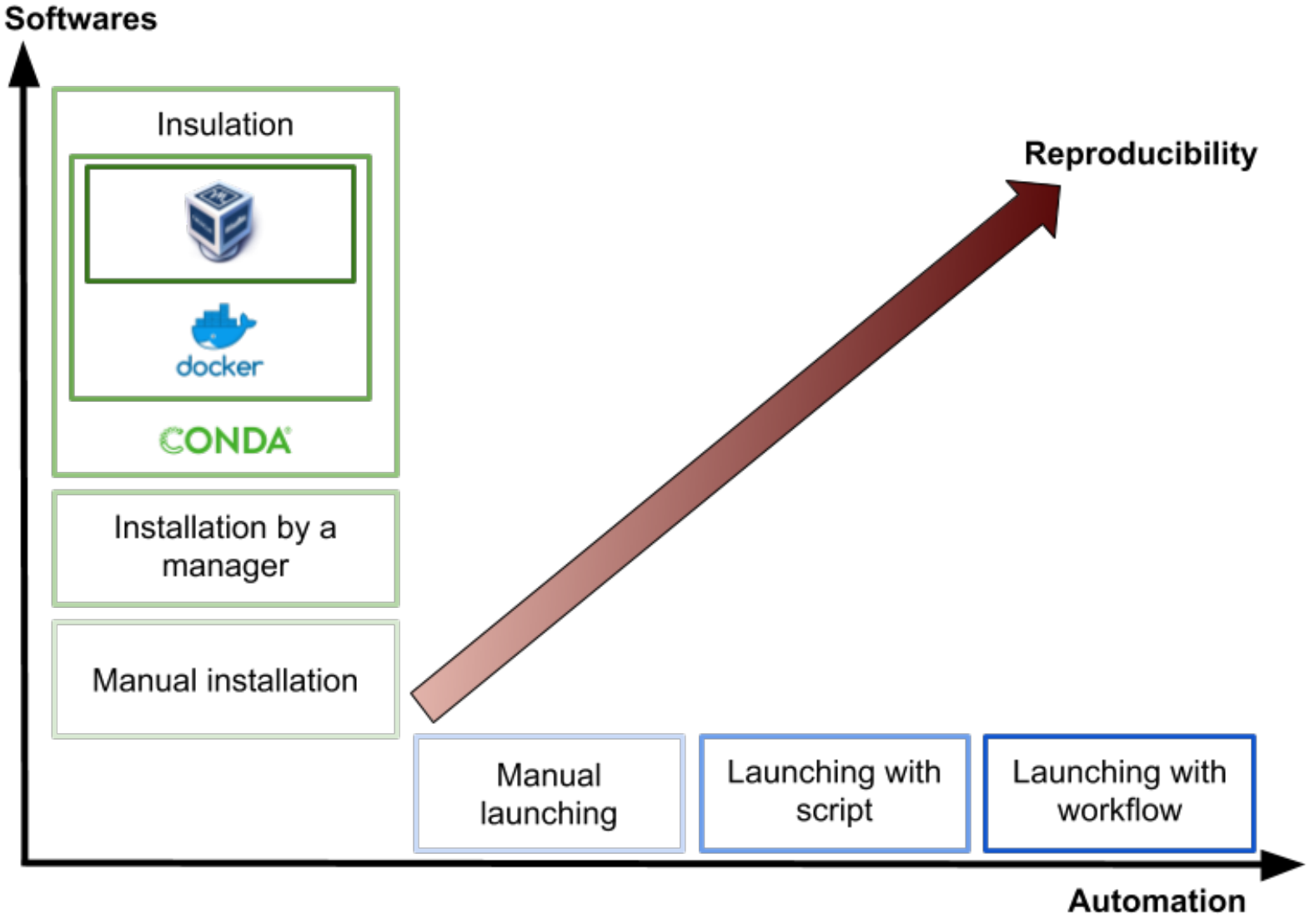
- any regression in the developed application
- any change in the results obtained

Travis CI

circleci



Reproducibility: a multidimensional and multi-level process



FAIR_bioinfo

Automation

Manual command lines

↓

Write a shell script

↓

Use a workflow manager

↓

Tests and continuous integration (*)

User analysis (trial-and-error)

Offer a GUI (eg. with R-Shiny) (*)

↓

Save and re-import choices (*)

Softwares

Local installation

↓

Package manager

↓

Conda environment

↓

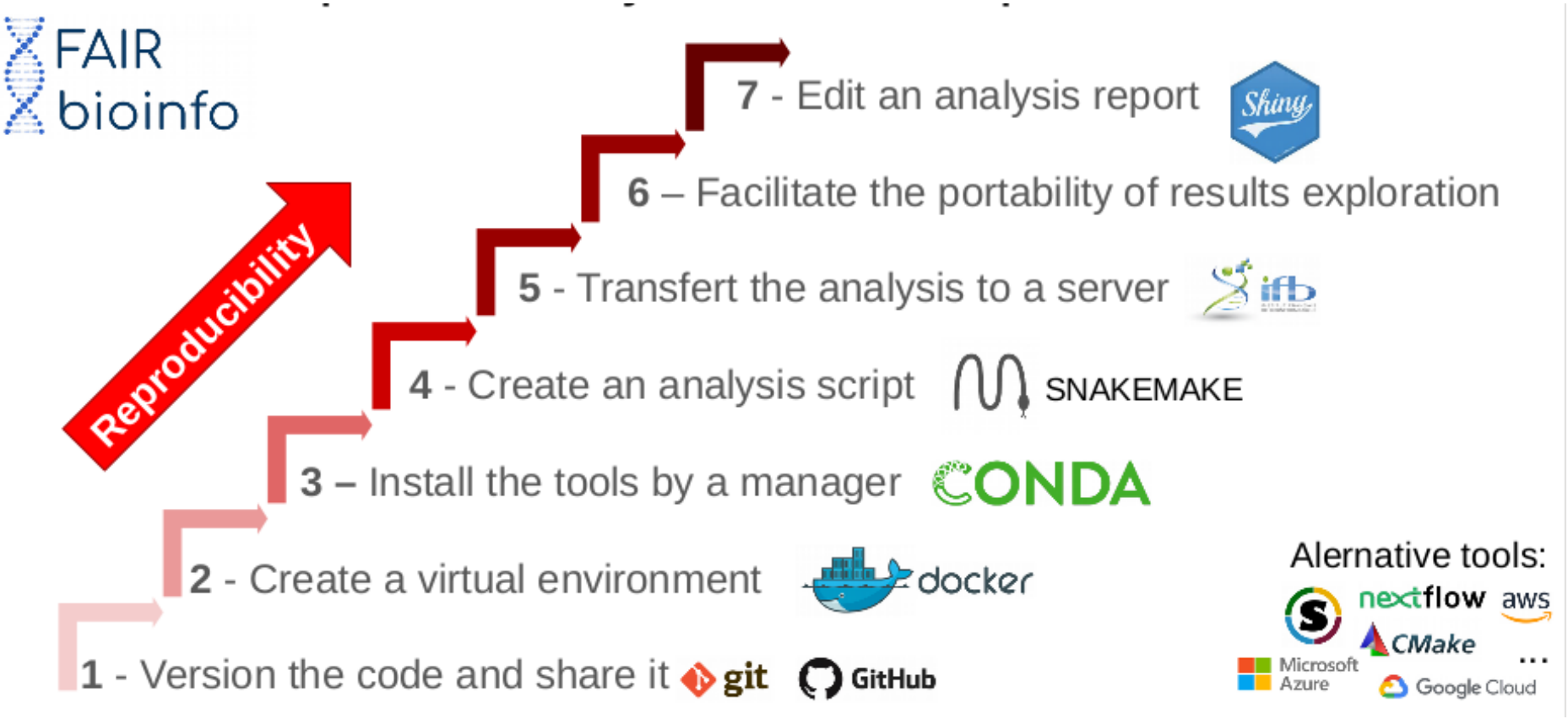
Image / container

↓

Virtual machine (*)

(*) not carried out in the course

FAIR_bioinfo



Reproducibility - how far?

Reproducibility to the exact bit?

✗ container uses some resources of the support machine

✓ version control of the env. (Nix, Guix)

HPC and parallelization?

✗ loss of computanional order, multi-threading, identical hardware?

✓ ...?

Thanks

- Organizational comity (our guardian angels): Yousra, Hélène
- IFB Core Cluster taskforce: Julien, Gildas, and all those who provide in the shadows
- Helpers: Paulette, Emilie, Pauline, Hugo
- Organisations: CNRS, INRAE, IFB, I2BC, Paris Saclay University