Reproducible science

Tools and good practices

Viola Fanfani

January 2022

Disclaimers

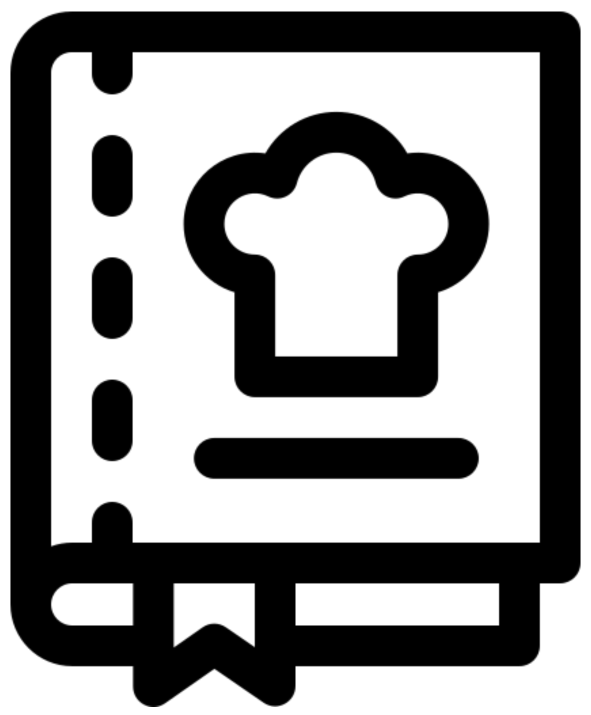
1. If you already know this stuff, I am sorry you have to hear about it again and please correct me. I am open to incorporate more/other things

2. These are things I have used and they have been helpful. We can mix and match, adopt things collectively or singularly, reshape structure according to the needs of the lab

3. Takes a moment to have everything up and running, but it saves time and headaches

Digital project management

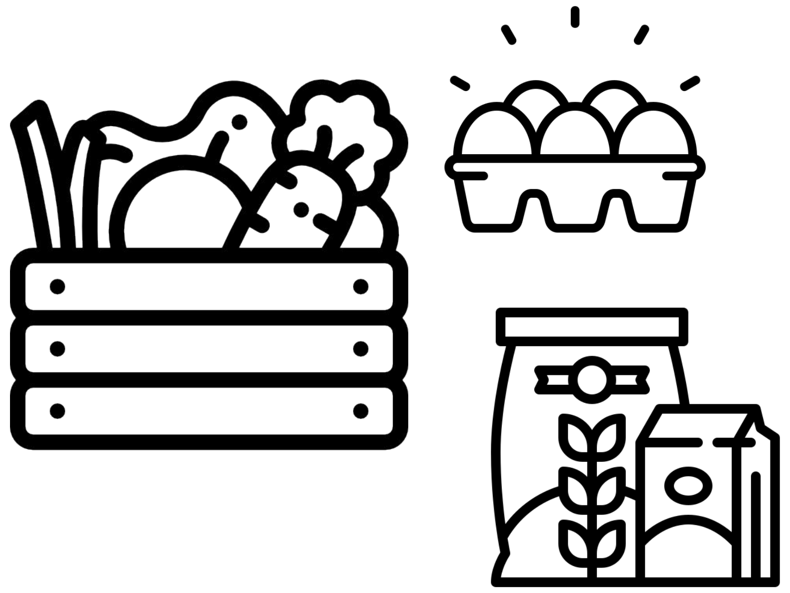
What reproducible science looks like 90% of the time:

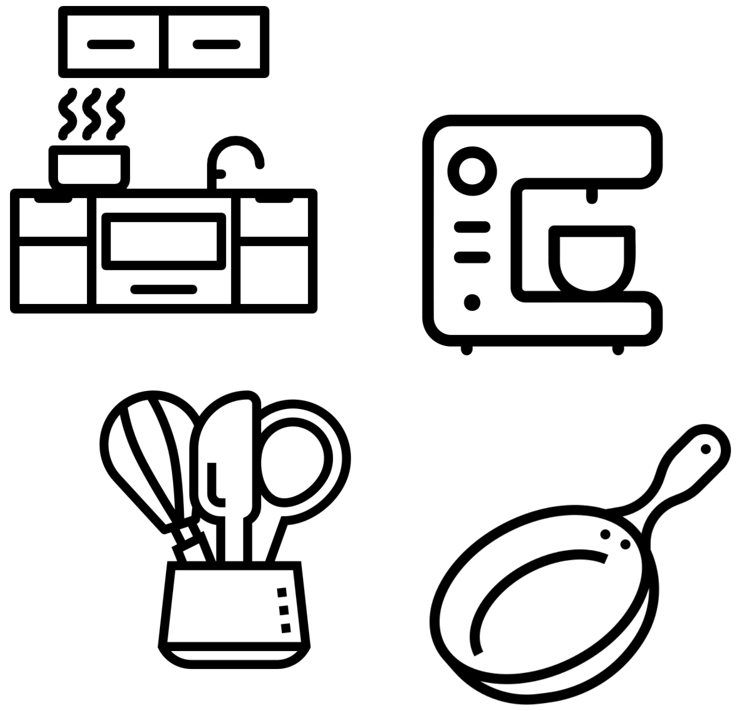


Kitchen icons created by kmg design, Food and restaurant icons created by Atif Arshad, Utensil icons created by Eucalyp, Kitchen icons created by xnimrodx , Kitchen icons created by ultimatearm , Harvest icons created by Freepik , Grocery icons created by iconixar , Ingredient icons created by Aficons studio , Recipe icons created by Freepik

Digital project management

What reproducible science should look like:





Kitchen icons created by kmg design, Food and restaurant icons created by Atif Arshad, Utensil icons created by Eucalyp, Kitchen icons created by xnimrodx , Kitchen icons created by ultimatearm , Harvest icons created by Freepik , Grocery icons created by iconixar , Ingredient icons created by Aficons studio , Recipe icons created by Freepik

Digital project management

It needs to be easy to prepare, to use, to distribute, and to reuse!

Think about a colleague (or yourself) who will work on your stuff 5 years from now

Data:

- safe
- portable
- organized

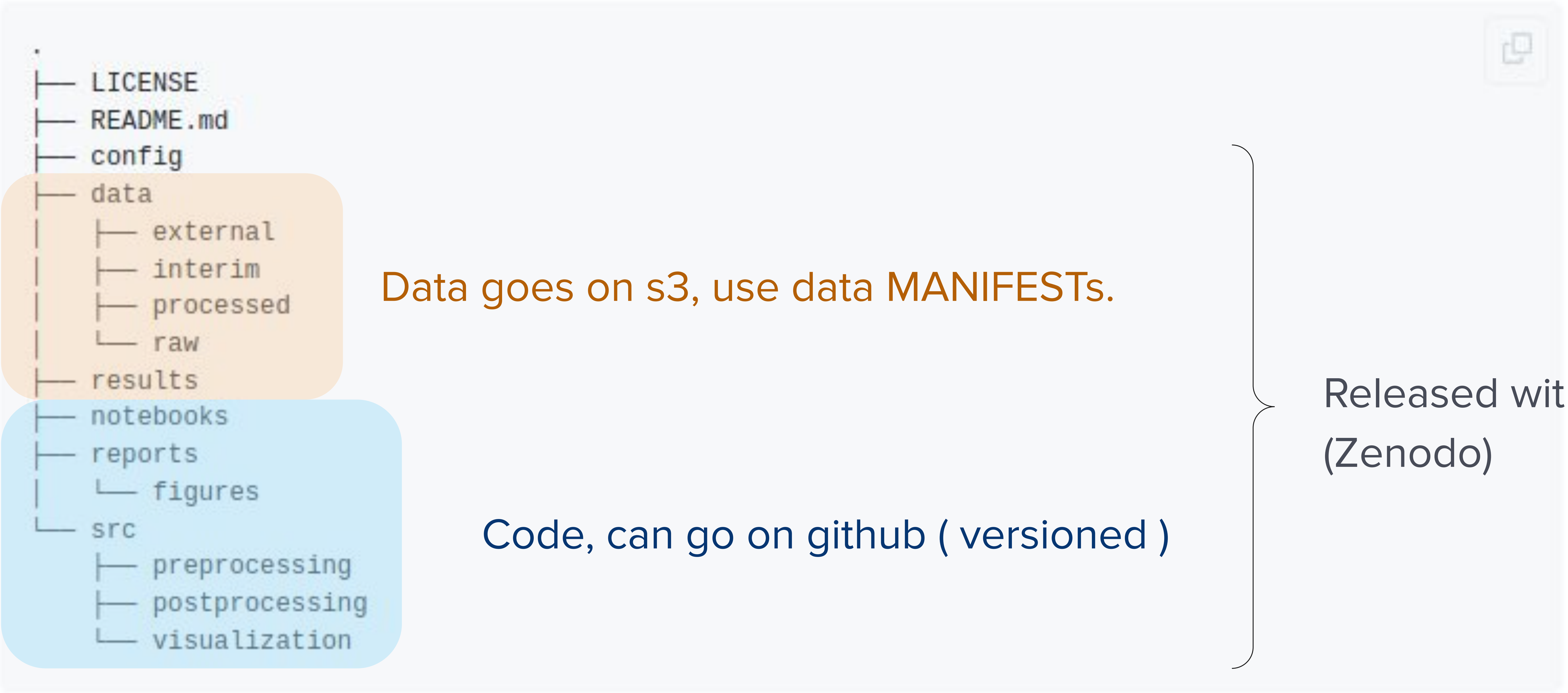
Code:

- reproducible
- portable
- (scalable)

git ANACONDA nextflow docker Snakemake

Organise the project folders

Consistent structure, which is easy to navigate, maintain, and release



```
.
├── LICENSE
├── README.md
├── config
├── data
│   ├── external
│   ├── interim
│   ├── processed
│   └── raw
├── results
├── notebooks
├── reports
│   └── figures
└── src
    ├── preprocessing
    ├── postprocessing
    └── visualization
```

Data goes on s3, use data MANIFESTs.

Code, can go on github (versioned)

Released with paper (Zenodo)

https://drivendata.github.io/cookiecutter-data-science/

Cookiecutter

Reproducible analysis

ANACONDA

Conda environments allow to specify and to keep track of the packages and their versions. Whoever tries to run the same code should have the same output and you should be able to reinstall everything on any machine. They also work with jupyter notebooks.

environment.yml

```
name: pygna
channels:
  - defaults
  - bioconda
  - conda-forge
dependencies:
  - python>=3.7
  - pandas
- numpy
  - scipy
  - matplotlib
  - pyyaml
  - pytables>=3.4.4
  - seaborn>=0.9
  - networkx=2.3
```

Reproducible analysis

Conda environments allow to specify and to keep track of the packages and their versions. Whoever tries to run the same code should have the same output and you should be able to reinstall everything on any machine. They also work with jupyter notebooks.

Docker containers allow to create ‘virtual machines’ with all the software you need. Not only you have all the conda packages, but you also set the system where the tools are installed. They make the analysis reproducible top to bottom.

Running jupyter projects with docker:
https://towardsdatascience.com/dockerizing-jupyter-projects-39aad547484a

Reproducible workflows

nextflow

Snakemake

Workflow management tools allow to specify a pipeline for the analysis: entire analysis reproducible with 1 command. Portable: it uses docker containers and works by using configuration files, the same analysis can be done on any dataset and any machine.

*cookiecutter

Nextflow AWS ready: launches jobs by itself, reads and saves files in S3, installs the whole image every time (no need to keep 100 AMIs)

Workflow Management Tools

nextflow

Metadata
Input files
computation.C
docker
generate_table.py
Output files
preprocess.R
analysis.py
plot.R
Configurations
Textual code

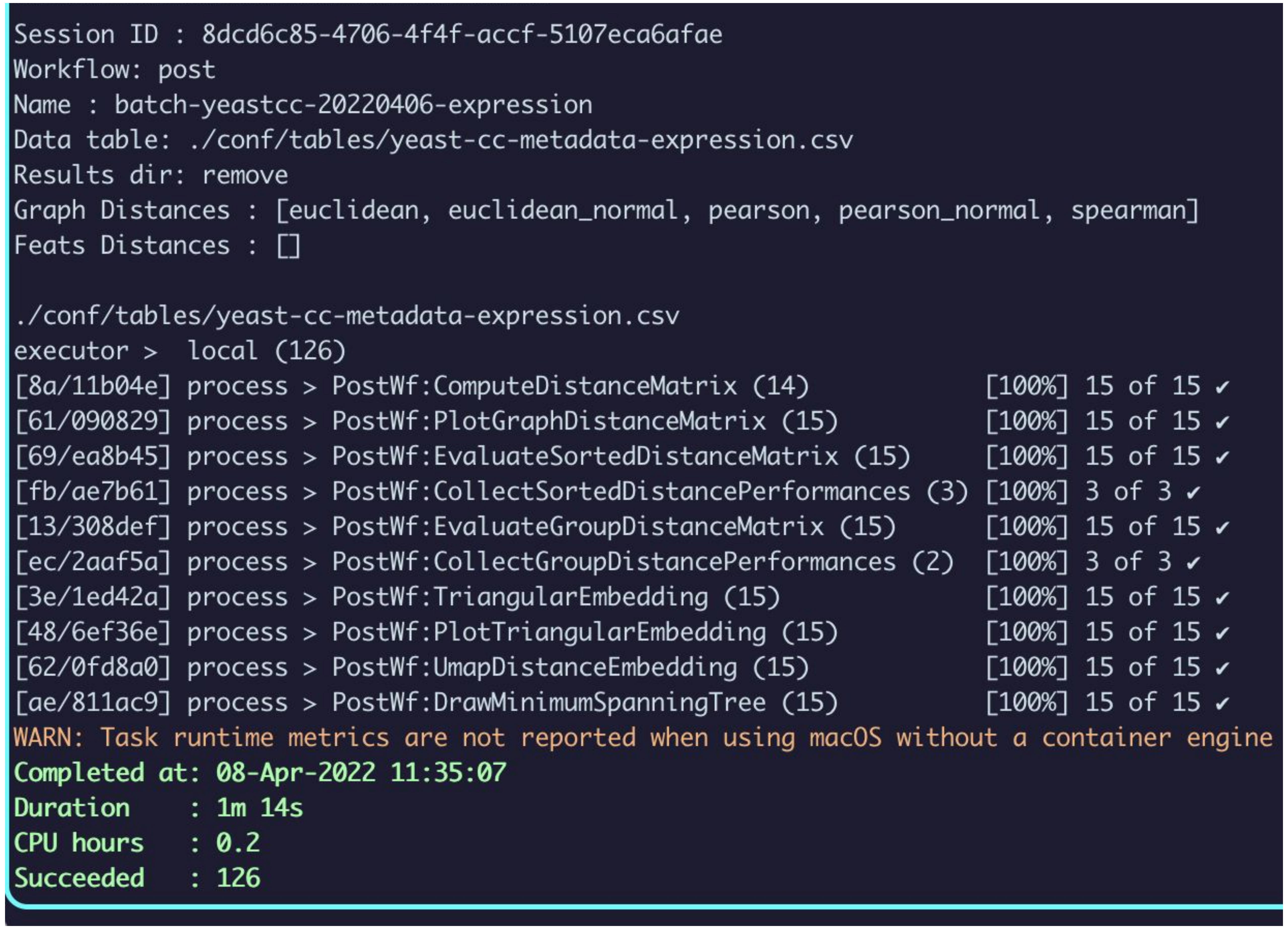
My workflow

```
> nextflow run yeast-cell-cycle-nf/main.nf -param <param-value> -resultsDir batch-yeast-20220406-expression ....
```

Channel.fromPath
Channel.from
featsDist
splitCsv
map
Channel.from
data
graphDist
combine
PostWf:ComputeDistanceMatrix
PostWf:PlotGraphDistanceMatrix
PostWf:TriangularEmbedding
PostWf:UmapDistanceEmbedding
PostWf:DrawMinimumSpanningTree
combine
combine
PostWf:PlotTriangularEmbedding
PostWf:EvaluateGroupDistanceMatrix
PostWf:EvaluateSortedDistanceMatrix
map
map
map
map
groupTuple
groupTuple
groupTuple
groupTuple
PostWf:CollectGroupDistancePerformances
PostWf:CollectSortedDistancePerformances

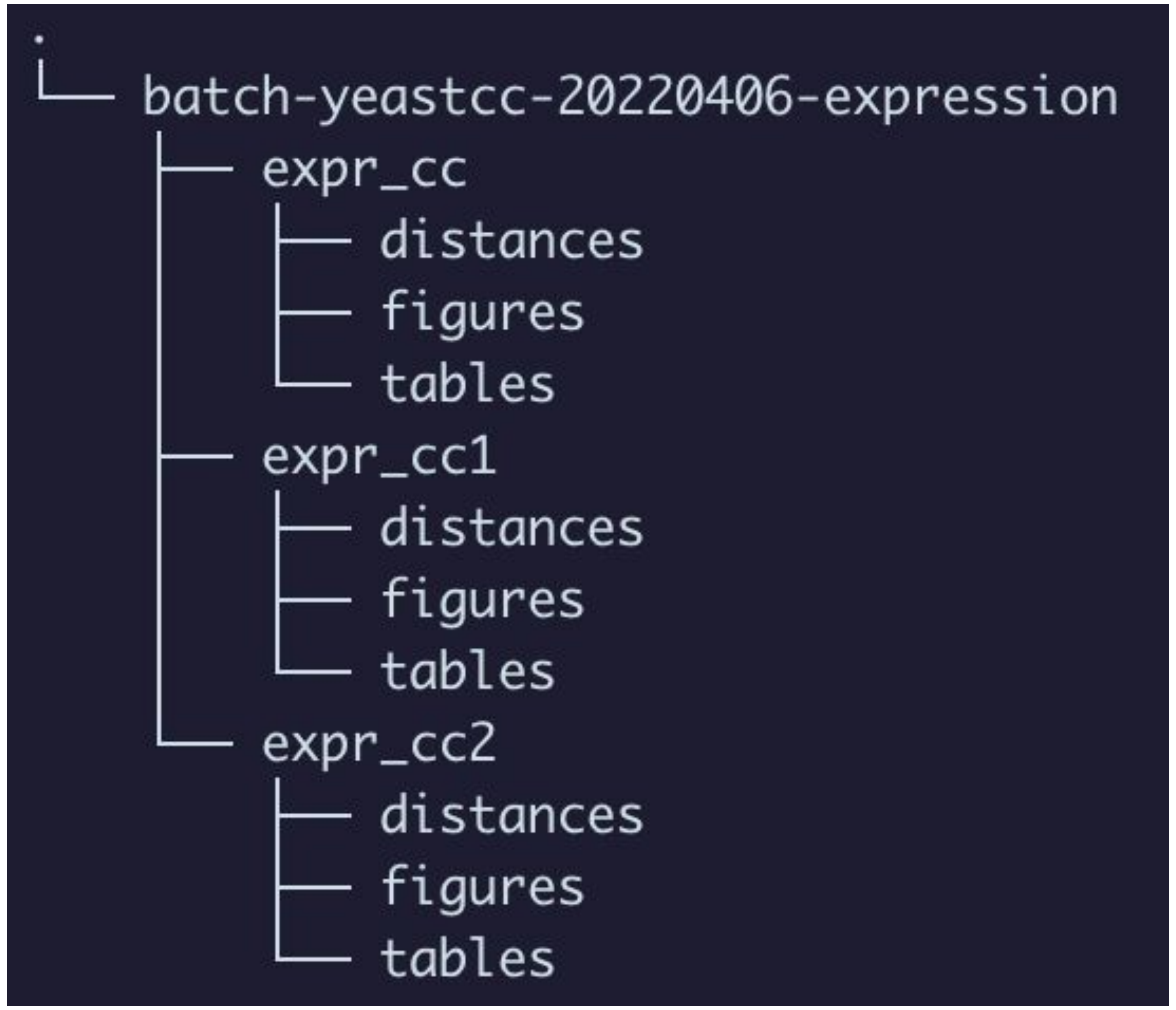
My workflow

> nextflow run yeast-cell-cycle-nf/main.nf -param <param-value> -resultsDir batch-yeast-20220406-expression ....



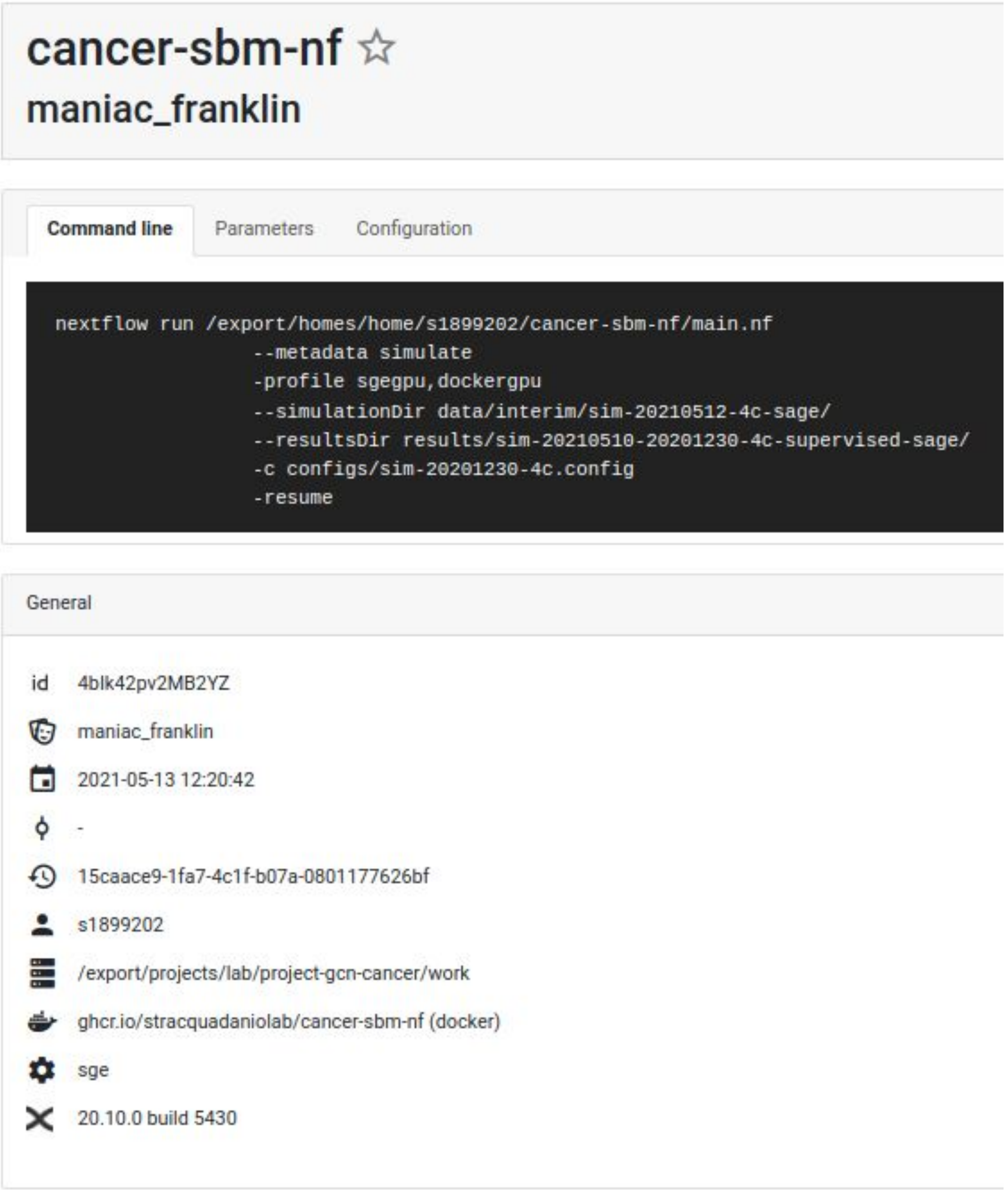
```
Session ID : 8dcd6c85-4706-4f4f-accf-5107eca6afae
Workflow: post
Name : batch-yeastcc-20220406-expression
Data table: ./conf/tables/yeast-cc-metadata-expression.csv
Results dir: remove
Graph Distances : [euclidean, euclidean_normal, pearson, pearson_normal, spearman]
Feats Distances : []

./conf/tables/yeast-cc-metadata-expression.csv
executor >  local (126)
[8a/11b04e] process > PostWf:ComputeDistanceMatrix (14)           [100%] 15 of 15 ✔
[61/090829] process > PostWf:PlotGraphDistanceMatrix (15)         [100%] 15 of 15 ✔
[69/ea8b45] process > PostWf:EvaluateSortedDistanceMatrix (15)    [100%] 15 of 15 ✔
[fb/ae7b61] process > PostWf:CollectSortedDistancePerformances (3) [100%] 3 of 3 ✔
[13/308def] process > PostWf:EvaluateGroupDistanceMatrix (15)     [100%] 15 of 15 ✔
[ec/2aaf5a] process > PostWf:CollectGroupDistancePerformances (2)  [100%] 3 of 3 ✔
[3e/1ed42a] process > PostWf:TriangularEmbedding (15)             [100%] 15 of 15 ✔
[48/6ef36e] process > PostWf:PlotTriangularEmbedding (15)         [100%] 15 of 15 ✔
[62/0fd8a0] process > PostWf:UmapDistanceEmbedding (15)           [100%] 15 of 15 ✔
[ae/811ac9] process > PostWf:DrawMinimumSpanningTree (15)         [100%] 15 of 15 ✔
WARN: Task runtime metrics are not reported when using macOS without a container engine
Completed at: 08-Apr-2022 11:35:07
Duration    : 1m 14s
CPU hours   : 0.2
Succeeded   : 126
```



```
.
└── batch-yeastcc-20220406-expression
    ├── expr_cc
    │   ├── distances
    │   ├── figures
    │   └── tables
    ├── expr_cc1
    │   ├── distances
    │   ├── figures
    │   └── tables
    └── expr_cc2
        ├── distances
        ├── figures
        └── tables
```

Nextflow



cancer-sbm-nf ☆

maniac_franklin

```
nextflow run /export/homes/home/s1899202/cancer-sbm-nf/main.nf
                 --metadata simulate
                 -profile sgegpu,dockergpu
                 --simulationDir data/interim/sim-20210512-4c-sage/
                 --resultsDir results/sim-20210510-20201230-4c-supervised-sage/
                 -c configs/sim-20201230-4c.config
                 -resume
```

General

id 4blk42pv2MB2YZ

maniac_franklin

2021-05-13 12:20:42

-

15caace9-1fa7-4c1f-b07a-0801177626bf

s1899202

/export/projects/lab/project-gcn-cancer/work

ghcr.io/stracquadaniolab/cancer-sbm-nf (docker)

sge

20.10.0 build 5430

Possible workflow for the lab

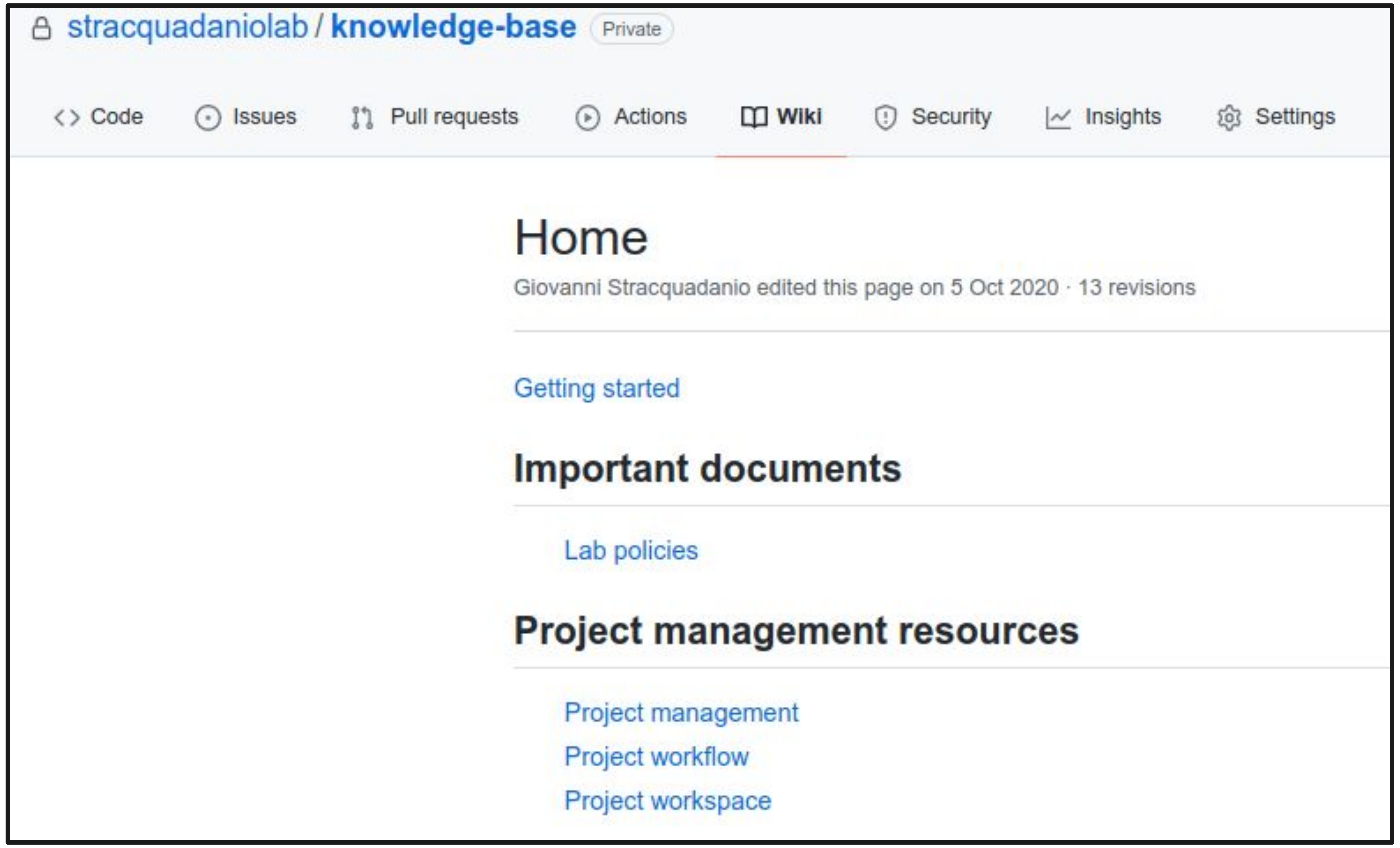
- Create data folder: data sync on S3 and code sync on github. Preprocessing, exploratory analyses, final postprocessing, examples all in here.

- Create workflow for the analysis: testable on your own computer and then usable on AWS

- Run analysis on AWS (possibly with AWS batches)

Lab wiki

Keep lab knowledge and expertise in a ‘centralised’ wiki.



Home

Giovanni Stracquadanio edited this page on 5 Oct 2020 · 13 revisions

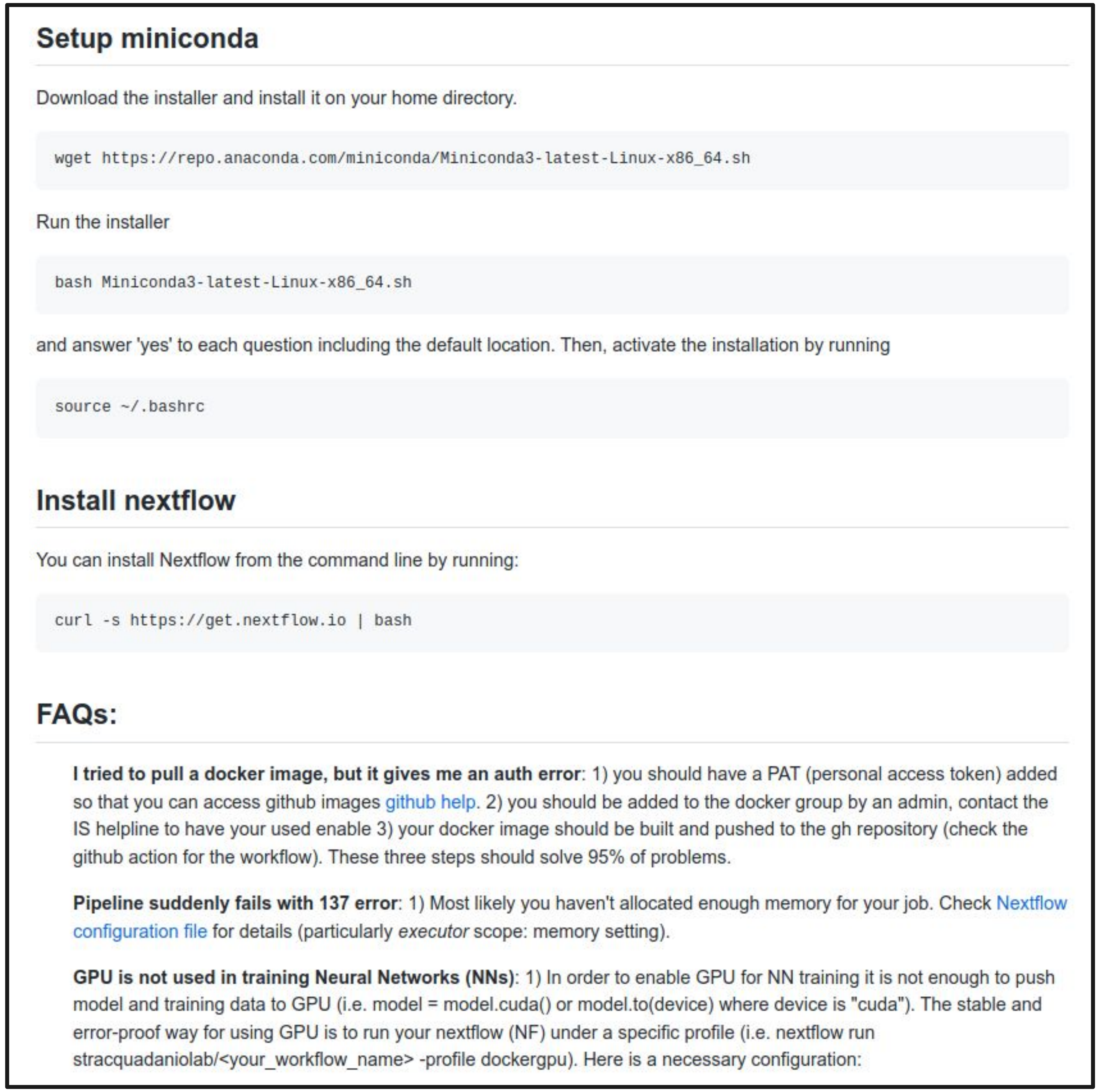
Getting started

Important documents

- Lab policies

Project management resources

- Project management
- Project workflow
- Project workspace



Setup miniconda

Download the installer and install it on your home directory.

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

Run the installer

```
bash Miniconda3-latest-Linux-x86_64.sh
```

and answer 'yes' to each question including the default location. Then, activate the installation by running

```
source ~/.bashrc
```

Install nextflow

You can install Nextflow from the command line by running:

```
curl -s https://get.nextflow.io | bash
```

FAQs:

I tried to pull a docker image, but it gives me an auth error: 1) you should have a PAT (personal access token) added so that you can access github images github help. 2) you should be added to the docker group by an admin, contact the IS helpline to have your used enable 3) your docker image should be built and pushed to the gh repository (check the github action for the workflow). These three steps should solve 95% of problems.

Pipeline suddenly fails with 137 error: 1) Most likely you haven't allocated enough memory for your job. Check Nextflow configuration file for details (particularly *executor* scope: memory setting).

GPU is not used in training Neural Networks (NNs): 1) In order to enable GPU for NN training it is not enough to push model and training data to GPU (i.e. model = model.cuda() or model.to(device) where device is "cuda"). The stable and error-proof way for using GPU is to run your nextflow (NF) under a specific profile (i.e. nextflow run stracquadaniolab/<your_workflow_name> -profile dockergpu). Here is a necessary configuration:

Reproducible science

Tools and good practices

Viola Fanfani

January 2022