Conda, Snakemake, and Jupyter: an OS tech-stack for reproducible research

Chris Gates (cgates@umich.edu)
UM Bioinformatic Core

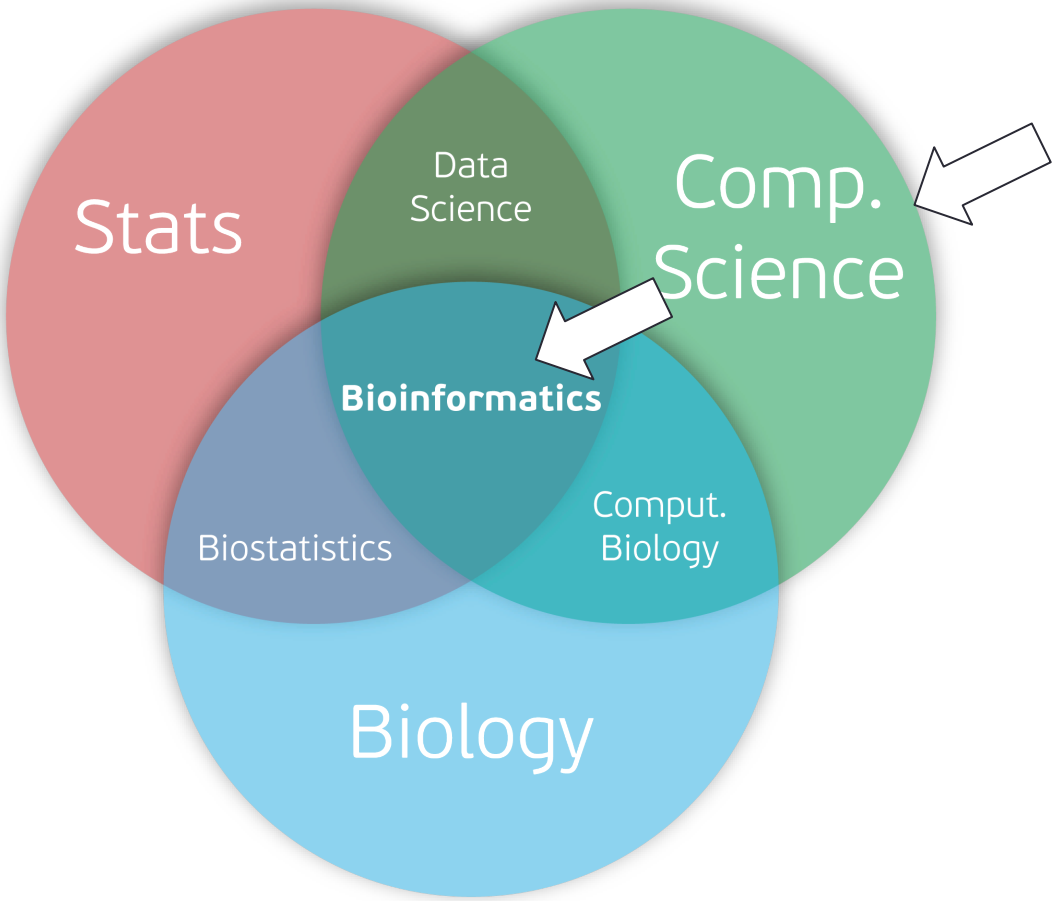
Conda, Snakemake, and Jupyter : An OS tech-stack for reproducible research

- Introductions
- Reproducibility in theory
- Reproducibility in action
  - Conda
  - Snakemake
  - Jupyter
- Reflections

About you

- How many use command line?
- Are you actively using Conda, Snakemake, or Jupyter?
- Name/describe more than three types of RNA
- Contrast a Bonferroni correction with a Benjamini–Hochberg adjustment
- What is the DRY principle?

About me


Stats
Data Science
Comp. Science
Bioinformatics
Biostatistics
Comput. Biology
Biology

Is Reproducibility good?

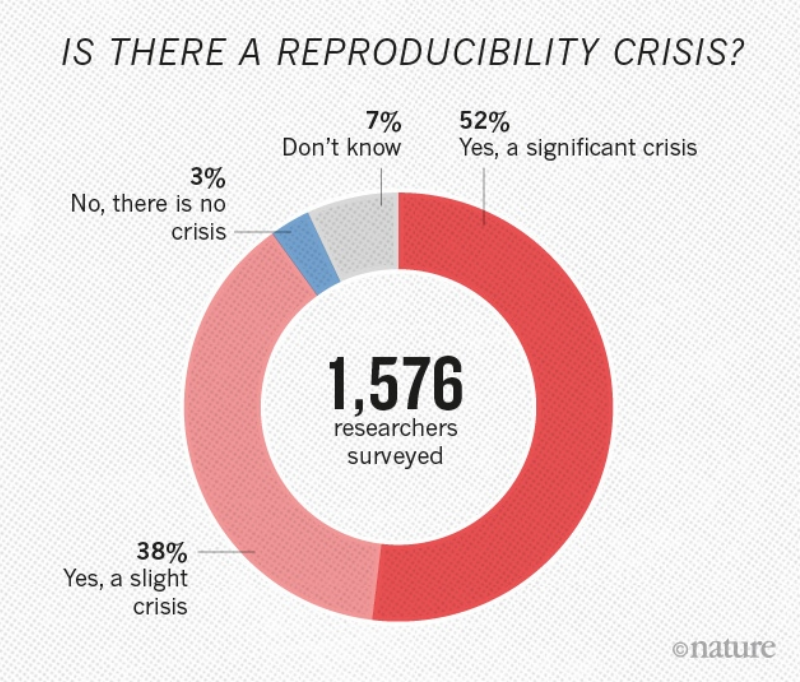
Is Reproducibility good?

- Show evidence of correctness
- Enable others* to make use of methods and results
- Epistemology of science

* including yourself in 4-6 months

http://ropensci.github.io/reproducibility-guide

Reproducibility is not a solved problem



- Psychology: Reproducibility project (2015): 39/100 ~39% replicated
- Pharmacology: phase I, phase II: 18%
- Cancer: Begley & Ellis: 6/53~11% replicated

- *Baker, M. 1,500 scientists lift the lid on reproducibility. (Nature 2016)*
- *Open Science Collaboration. Estimating the reproducibility of psychological science (Science 2015)*
- *Prinz, F. et al. Believe it or not: How much can we rely on published data on potential drug targets? (Nat Rev Drug Discovery 2011)*
- *Begley, C et al. Drug development: Raise standards for preclinical cancer research". (Nature 2012)*

What we mean when we say reproducible

- Empirical reproducibility
- Statistical reproducibility
- Computational reproducibility

> *"An article about computational results is advertising, not scholarship. The actual scholarship is the **full software environment**, **code** and **data**, that produced the result."*

- *http://ropensci.github.io/reproducibility*
- *Stodden et al. Setting the Default to Reproducible (2013)*
- *https://fivethirtyeight.com/features/science-isnt-broken/#part1*
- *Claerbout and Karrenbach. Electronic documents give reproducible research a new meaning (1992)*

Agenda

- Introductions
- Reproducibility in theory
- **Reproducibility in action**
  - **Conda**
  - Snakemake
  - Jupyter
- Reflections

Conda improves reproducibility

Conda is a package and environment management system

- Install, update, and remove packages
- Create, configure, and share compute environments

Create/activate a Conda environment

```
$ conda create --name 'reprod_tnt'
Solving environment: done

  environment location: /Users/cgates/miniconda3/envs/reprod_tnt


…
# To activate this environment, use:
# > source activate reprod_tnt
#
# To deactivate an active environment, use:
# > source deactivate
#

$ source activate reprod_tnt
(reprod_tnt) $
```

Install a package

```
(reprod_tnt) $ conda install jupyter
Solving environment: done
## Package Plan ##
  environment location: /Users/cgates/miniconda3/envs/reprod_tnt
  added / updated specs:
    - jupyter

The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    jupyter_console-5.2.0      |   py36hccf5b1c_1          35 KB
    jupyter-1.0.0              |           py36_4           5 KB
    …(6 omitted)…
    ------------------------------------------------------------
                                           Total:        82.5 MB
```

…

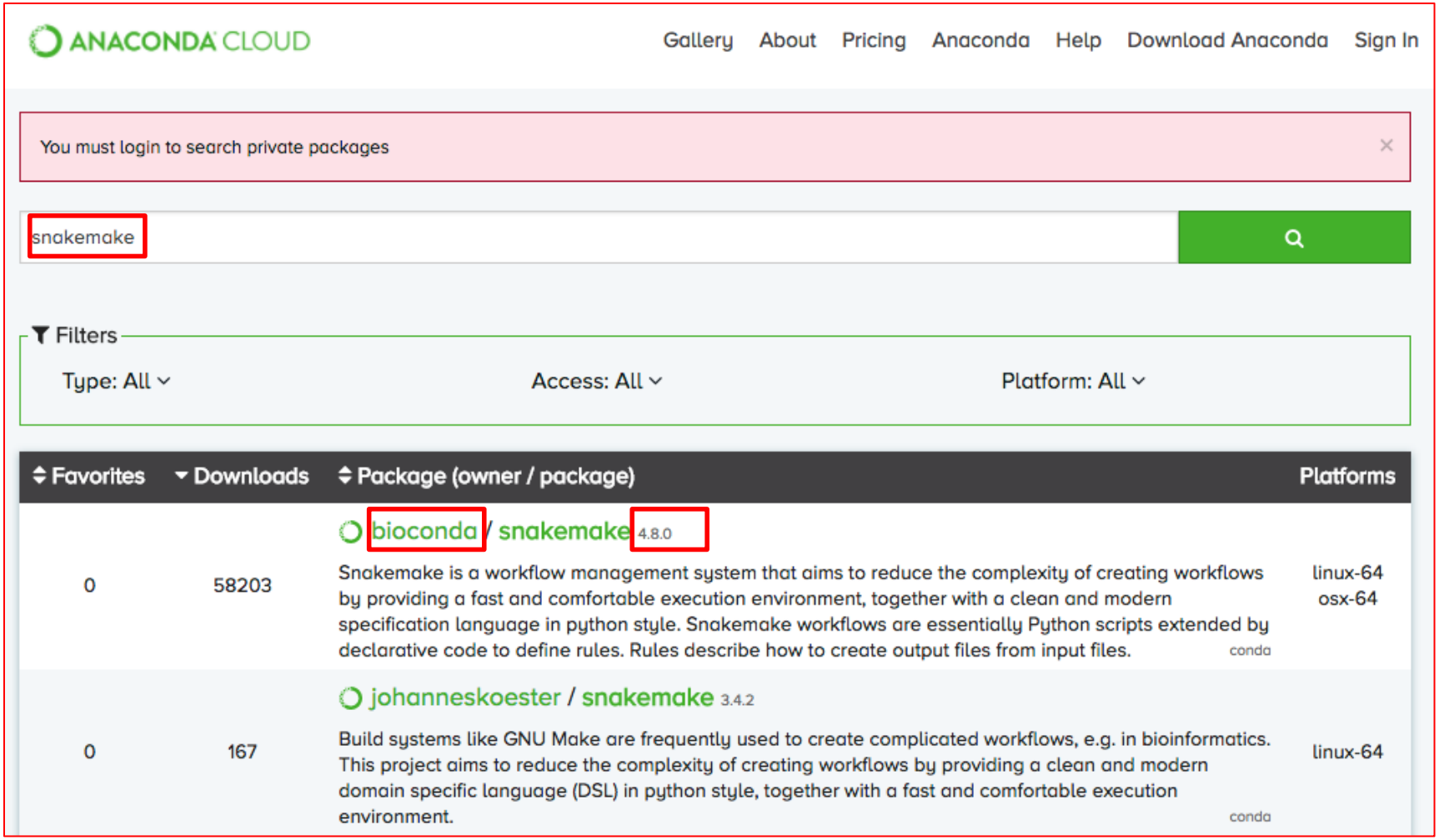
Install a package (cont.)

```
(cont.)
The following NEW packages will be INSTALLED:


    alabaster:                    0.7.10-py36h174008c_0
    appnope:                      0.1.0-py36hf537a9a_0
    asn1crypto:                   0.24.0-py36_0
    … (93 packages omitted) …
    zlib:                         1.2.11-hf3cbc9b_2


Downloading and Extracting Packages
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
(reprod_tnt) $ which jupyter
/Users/cgates/miniconda3/envs/reprod_tnt/bin/jupyter
```

Searching for a package

```
(reprod_tnt) $ conda search snakemake
Loading channels: done
PackagesNotFoundError: The following packages are not available from
current channels:
  - snakemake

Current channels:
  - https://repo.anaconda.com/pkgs/main/osx-64
  - https://repo.anaconda.com/pkgs/main/noarch
  - https://repo.anaconda.com/pkgs/free/osx-64
  …
  - https://repo.anaconda.com/pkgs/pro/noarch

To search for alternate channels that may provide the conda package
you're
looking for, navigate to
    https://anaconda.org
and use the search bar at the top of the page.
```

Searching for a package (cont.)



Installing a package from a channel

```
(reprod_tnt) $ conda search -c bioconda snakemake=4.8.0
Loading channels: done
# Name                  Version           Build  Channel
snakemake                 4.8.0          py35_0  bioconda
snakemake                 4.8.0          py36_0  bioconda

(reprod_tnt) $ conda install -c bioconda -c conda-forge snakemake
… (blah, blah, blah – lots of dependencies, lots of feedback)…
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(reprod_tnt) $ which snakemake
/Users/cgates/miniconda3/envs/reprod_tnt/bin/snakemake
```

Listing packages / revisions

```
(reprod_tnt) $ conda list
# packages in environment at /Users/cgates/miniconda3/envs/reprod_tnt:
## Name                    Version                   Build    Channel
 aioeasywebdav             2.2.0                     py36_0    conda-forge
 aiohttp                   3.1.3                     py36_0    conda-forge
 alabaster                 0.7.10           py36h174008c_0
 … (140 packages omitted) …
 yarl                      1.1.1                     py36_0    conda-forge
 zeromq                    4.2.5                  h378b8a2_0
 zlib                      1.2.11                 hf3cbc9b_2


(reprod_tnt) $ conda list -revisions
2018-04-18 09:46:08  (rev 0) (when I created the environment)

2018-04-18 09:55:20  (rev 1) (when I installed jupyter)
    +alabaster-0.7.10
    ... (other packages that were installed) ...

2018-04-18 11:10:39  (rev 2) (when I installed snakemake)
    +aioeasywebdav-2.2.0 (conda-forge)
     ... (other packages that were installed) ...
```

Listing environments

```
(reprod_tnt) $ conda info --envs
# conda environments:
#
base                     /Users/cgates/miniconda3
arhim_rs10               /Users/cgates/miniconda3/envs/arhim_rs10
jupyter                  /Users/cgates/miniconda3/envs/jupyter
py36_pysam13             /Users/cgates/miniconda3/envs/py36_pysam13
py36_pysam14             /Users/cgates/miniconda3/envs/py36_pysam14
reprod_tnt             *  /Users/cgates/miniconda3/envs/reprod_tnt
spike                    /Users/cgates/miniconda3/envs/spike
```

Sharing environments

```
(reprod_tnt) $ conda env export > environment.yaml
(reprod_tnt) $ head environment.yaml
name: reprod_tnt
channels:
  - bioconda
  - conda-forge
  - defaults
dependencies:
  - filechunkio=1.6=py36_0
  - ftputil=3.2=py36_0
  …
```

```
$ conda env create \
      --file environment.yaml \
      --name tools_and_tech
...

$ conda info --envs
# conda environments:
#
base                  *  /Users/vstodden/miniconda3
tools_and_tech           /Users/vstodden/miniconda3/envs/tools_and_tech
```

Conda can be ~~mysterious? oracular?~~ nuanced

- 80 wonderful / 20 inscrutable|infuriating
- Package management is essentially hard
- You become the repo maintainer

Conda enables you to manage and share your compute environment

Alternatives:

- VMWare/VBox, Amazon AMI/Google GCE
- Docker, Singularity
- Lmod/Environment modules
- Language/OS specific:
  - Virtualenv/PyPI (python)
  - packrat/CRAN (R)
  - rpm, brew, etc.

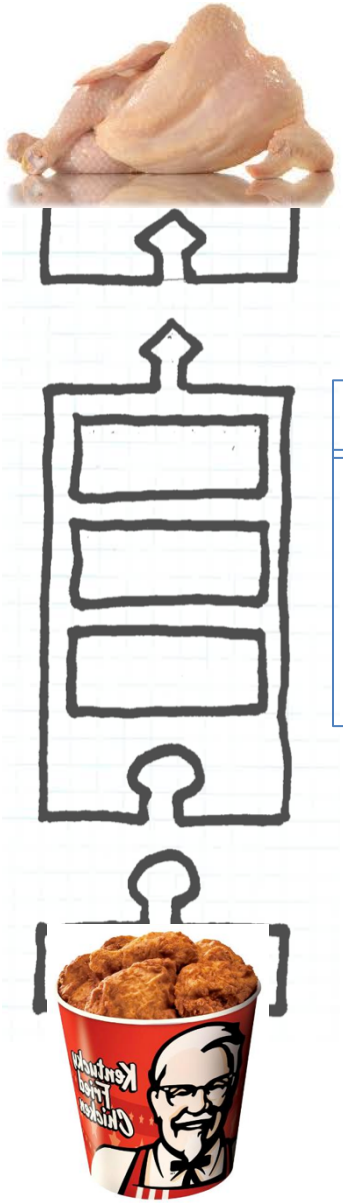
Can be used together (Conda + Docker + Packrat)

Conda supports: *Python, R, Ruby, Lua, Scala, Java, JavaScript, C/ C++, FORTRAN*

Agenda

- Introductions
- Reproducibility in theory
- Reproducibility in action
  - Conda
  - **Snakemake**
  - Jupyter
- Reflections

Consider a “traditional” data workflow



```
fried_chicken.sh

marinate chicken buttermilk > chicken.marinated

bread chicken.marinated > chicken.breaded

fry chicken.breaded > chicken.fried
```

An equivalent Snakemake workflow

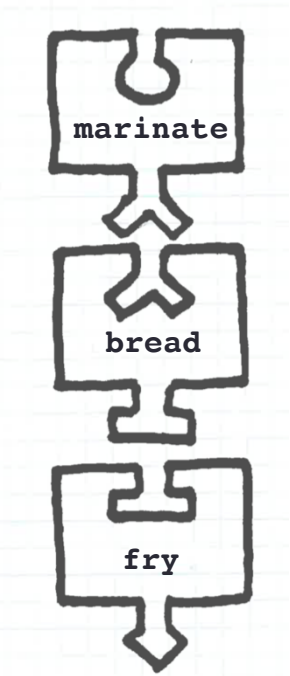
Snakefile

```
rule marinate:


    shell: 'marinate {input} > {output}'

rule bread:


    shell: 'bread {input} > {output}'

rule fry:


    shell: 'fry  {input} > {output}'
```

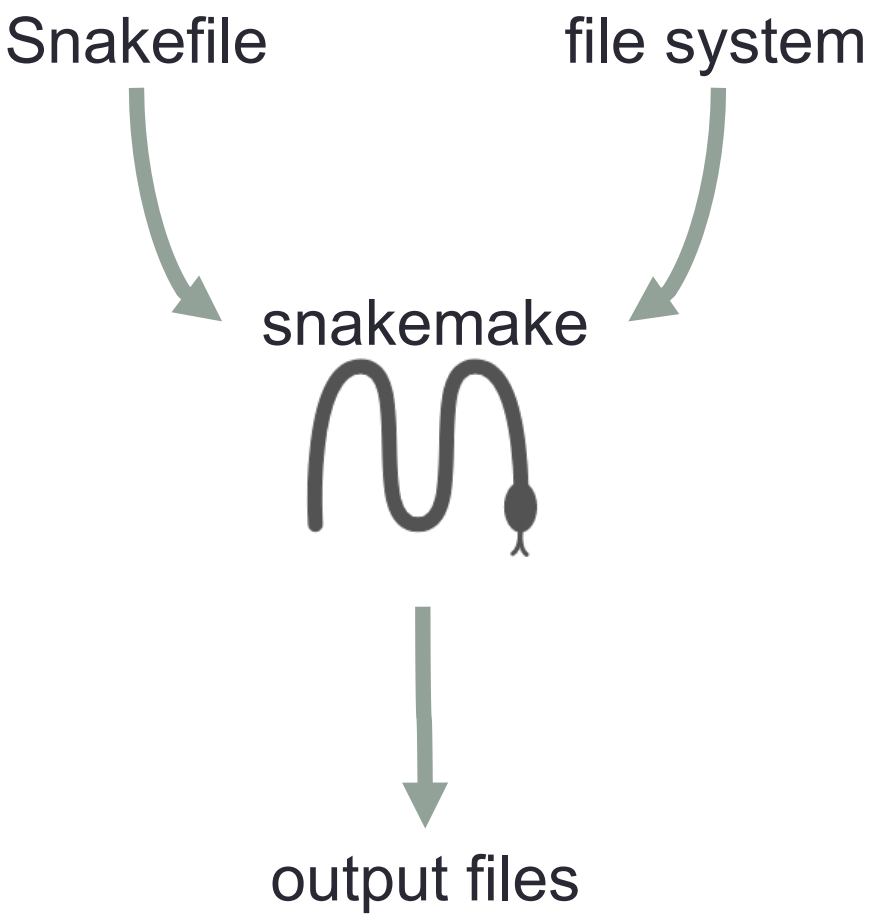
An equivalent Snakemake workflow

Snakefile

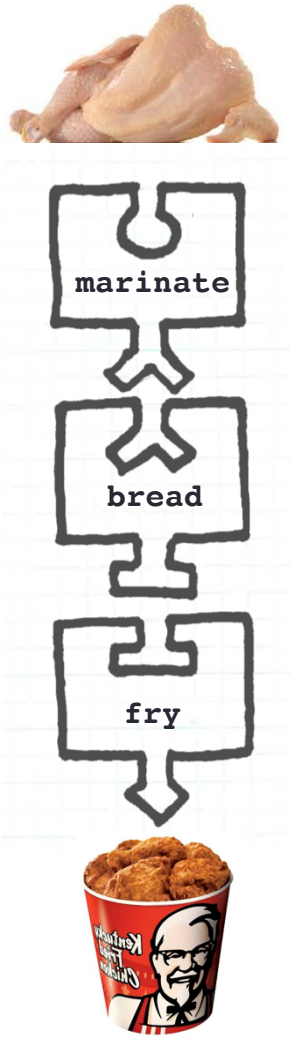
```
rule marinate:
    input: 'chicken', 'buttermilk'
    output: 'chicken.marinated'
    shell: 'marinate {input} > {output}'

rule bread:
    input: 'chicken.marinated'
    output: 'chicken.breaded'
    shell: 'bread {input} > {output}'

rule fry:
    input: 'chicken.breaded'
    output: 'chicken.fried'
    shell: 'fry  {input} > {output}'
```



Snakemake interprets a Snakefile to transform inputs to outputs

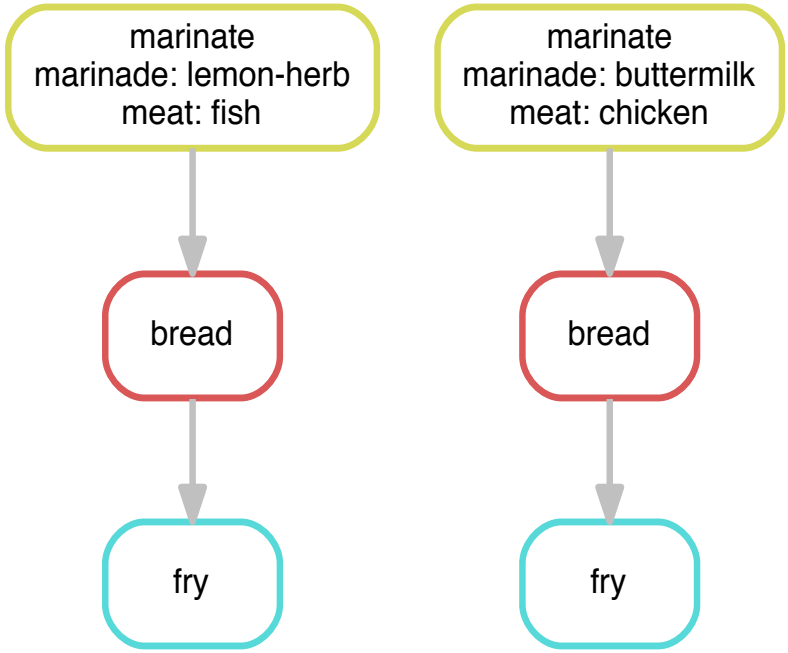



```
$ snakemake
```



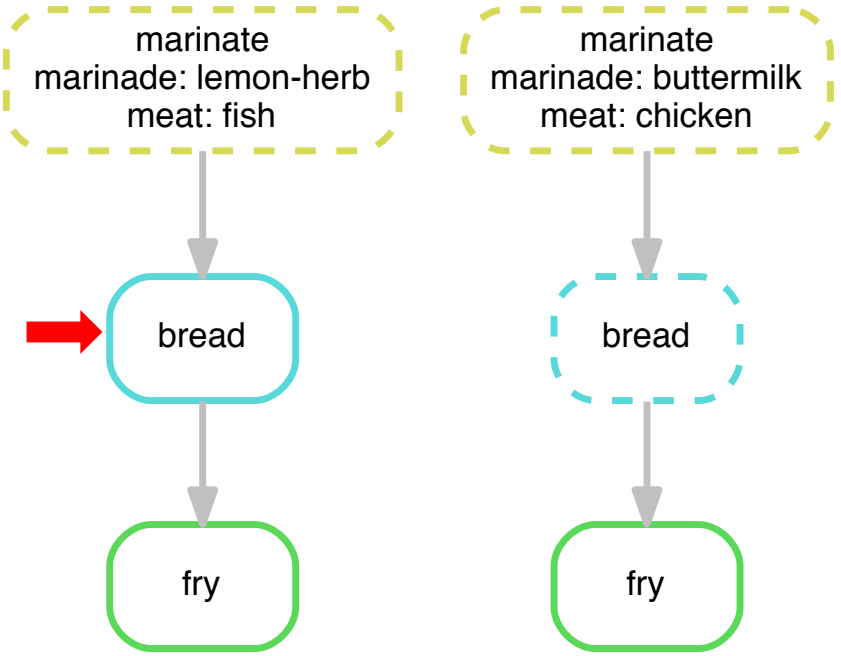
Snakemake extends to new inputs

```
$ snakemake

...
```



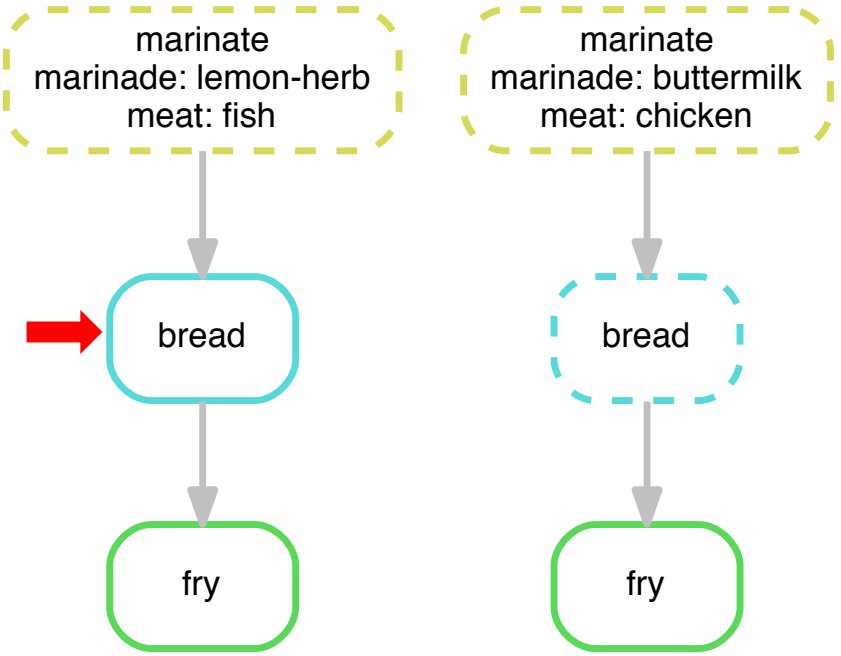
Snakemake is durable

```
$ snakemake
…
Error in job bread while
creating output file
lemon-herb.fish.breaded.
…
```

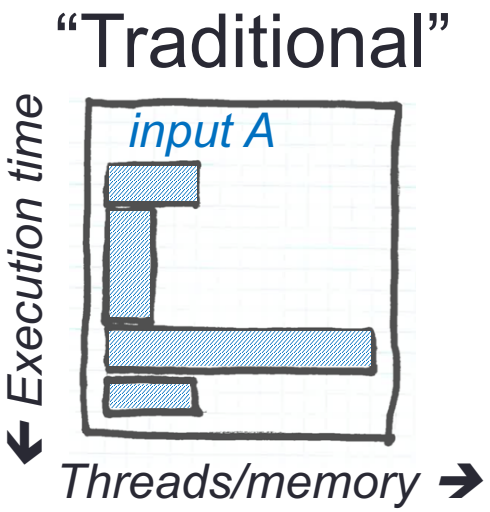


Snakemake is durable

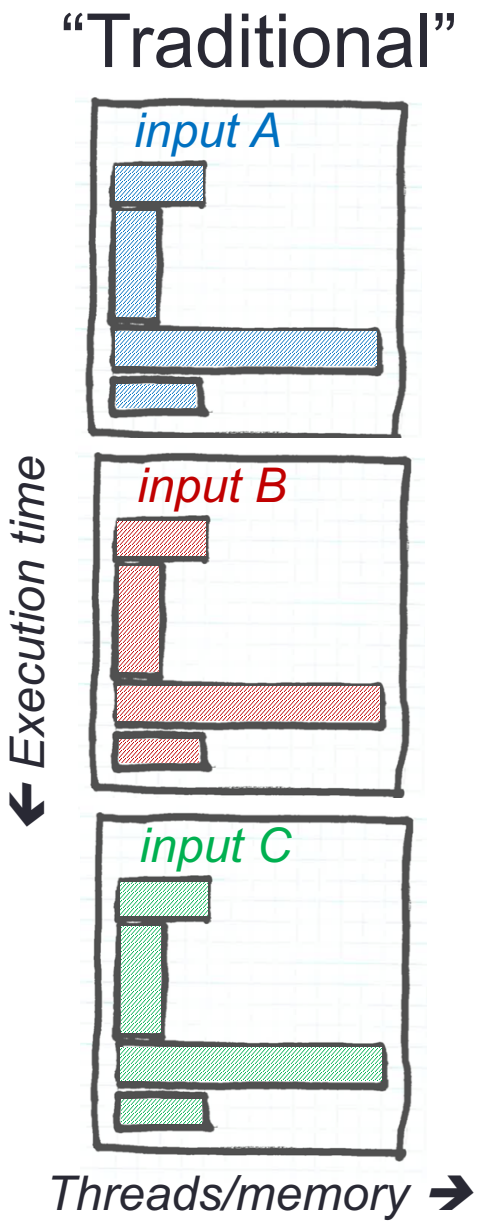
```
$ snakemake
…
Error in job bread while
creating output file
lemon-herb.fish.breaded.
…
(fix something)
$ snakemake
…
```



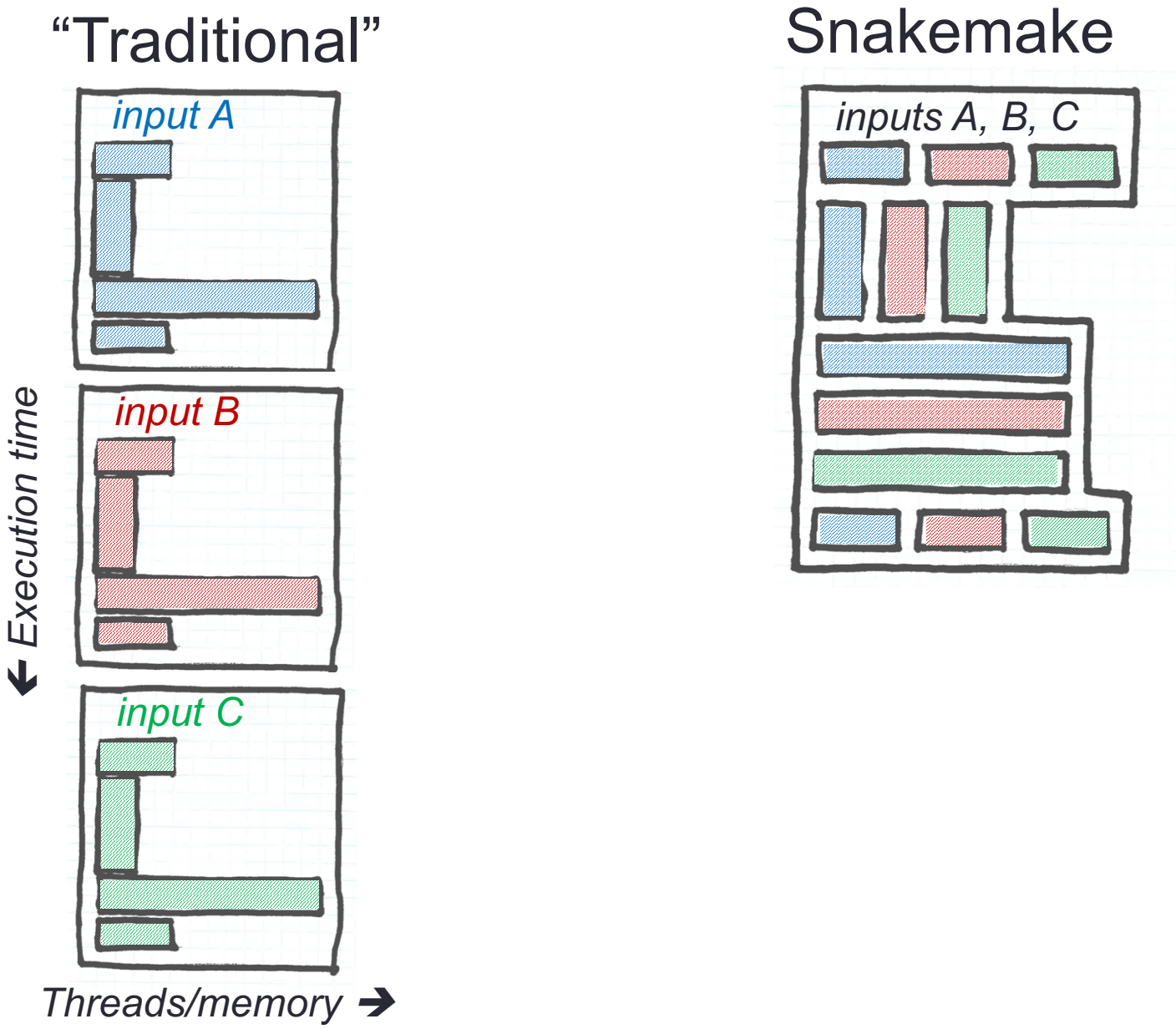
Snakemake is computationally efficient



Snakemake is computationally efficient



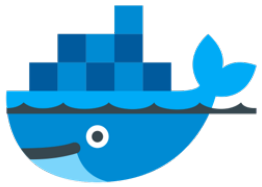
Snakemake is computationally efficient



Snakemake is portable

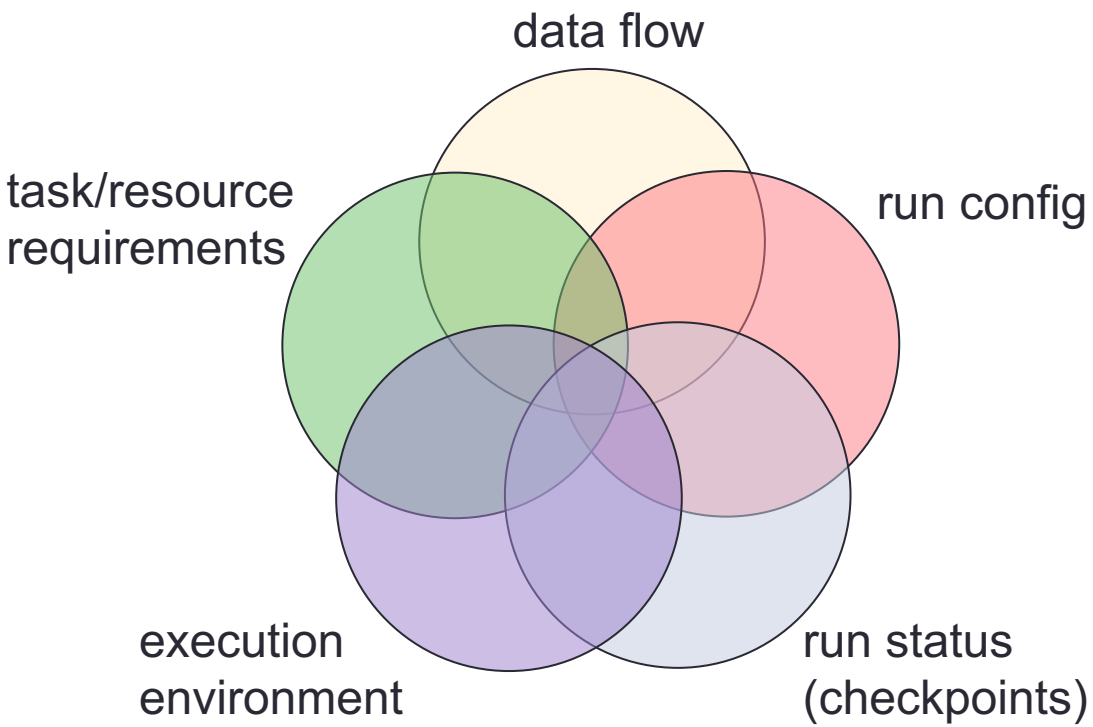
```
$ snakemake # to run on workstation

$ snakemake --cores=20 # to run on server

$ snakemake --drmaa # on HPC

$ snakemake --kubernetes
```



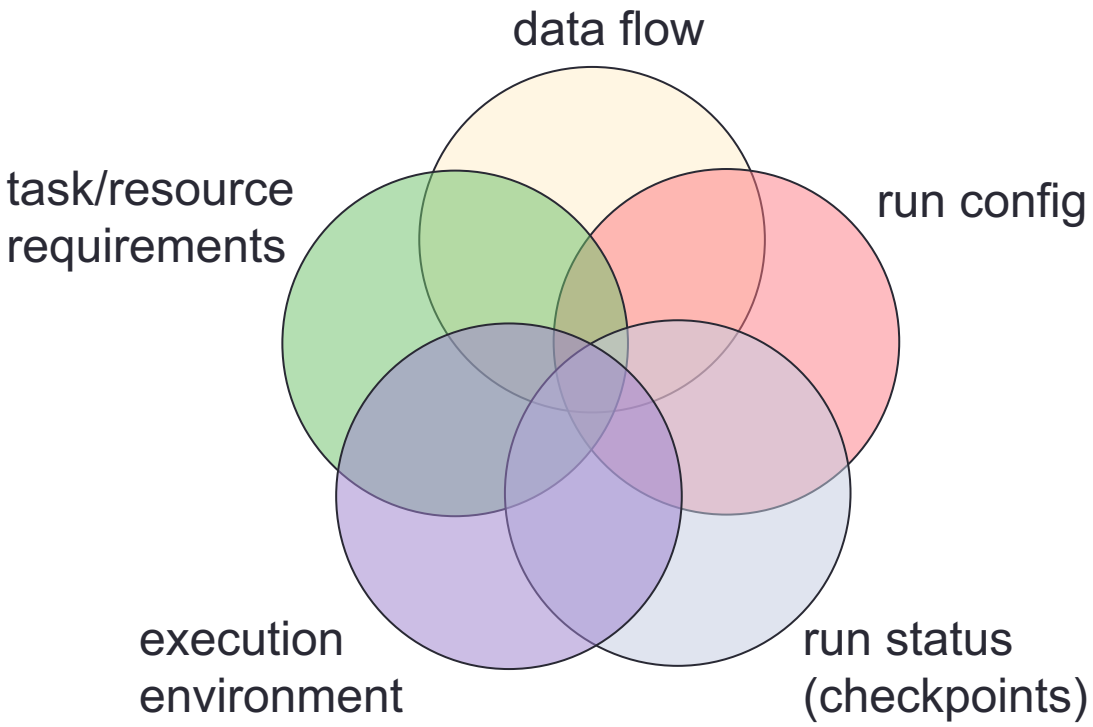

Snakemake enables me to focus on the data

"Traditional" workflow

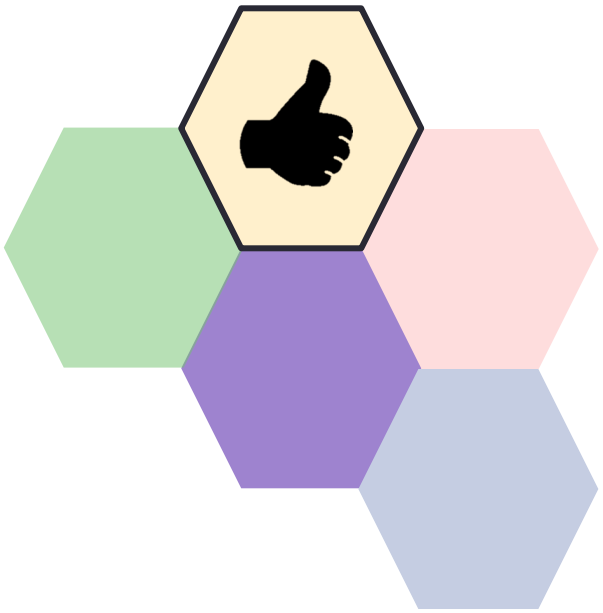


Snakemake enables me to focus on the data

"Traditional" workflow



Snakemake



Snakemake makes it easier to build and share good data workflows

Agenda

- Introductions
- Reproducibility in theory
- Reproducibility in action
  - Conda
  - Snakemake
  - **Jupyter**
- Reflections

Jupyter supports literate programming using a notebook idiom

"Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do."

Could we combine and share?

- Narrative
- Code
- Data
- Quantitative results
- Visualization

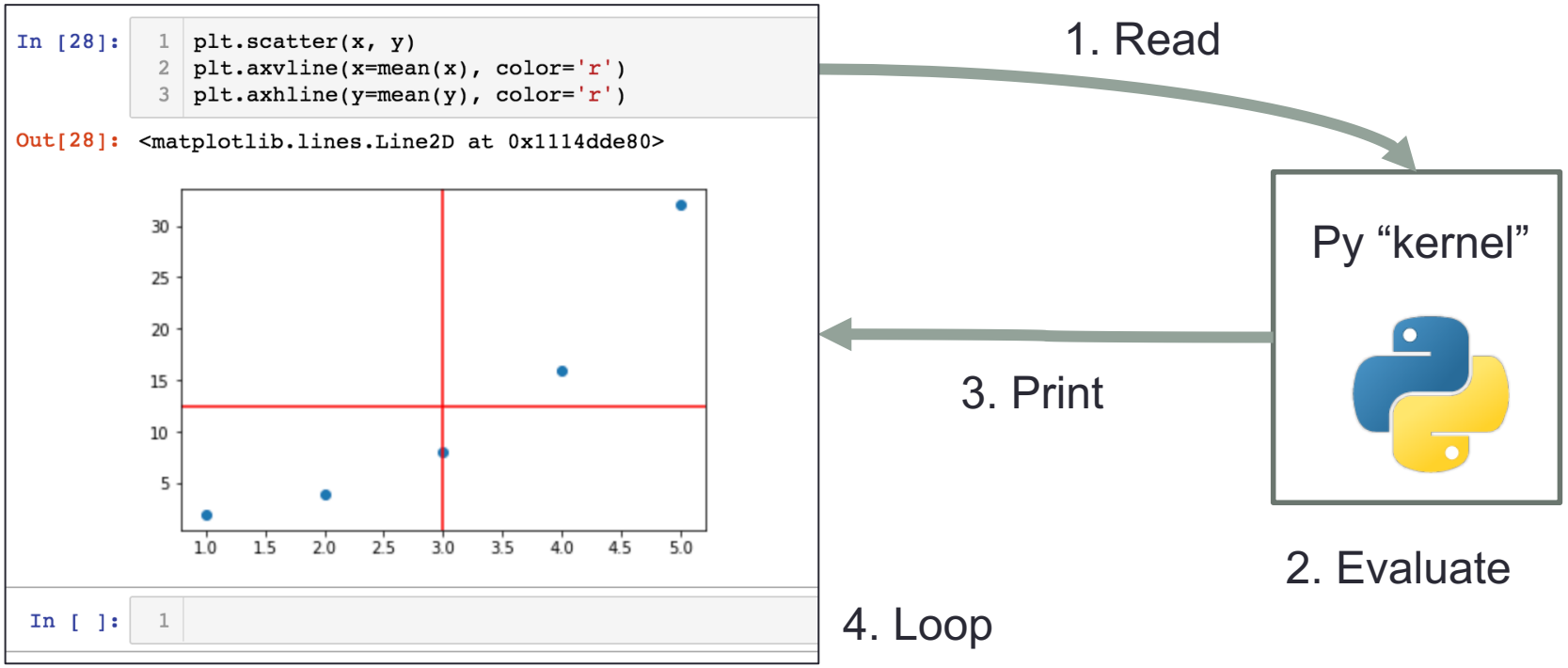
- *Knuth. Literate Programming (1984)*

Jupyter supports literate programming using a notebook idiom

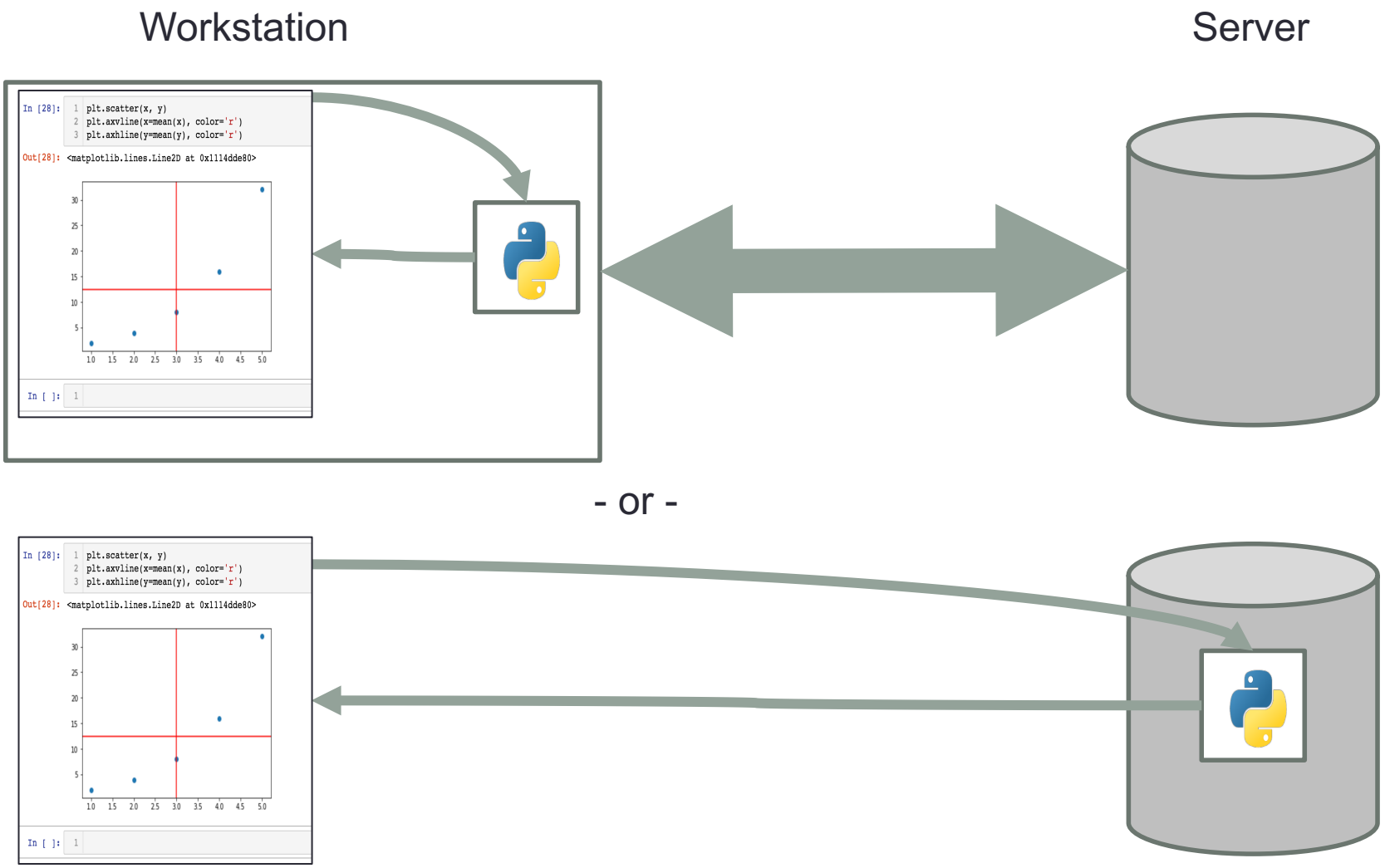
Mean

KMeans

Jupyter partitions “beauty” and “brains”



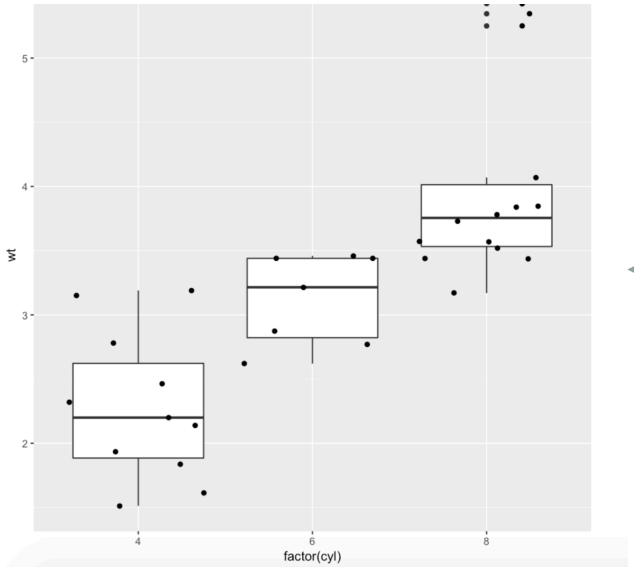
Jupyter partitions “beauty” and “brains”



Jupyter speaks many languages

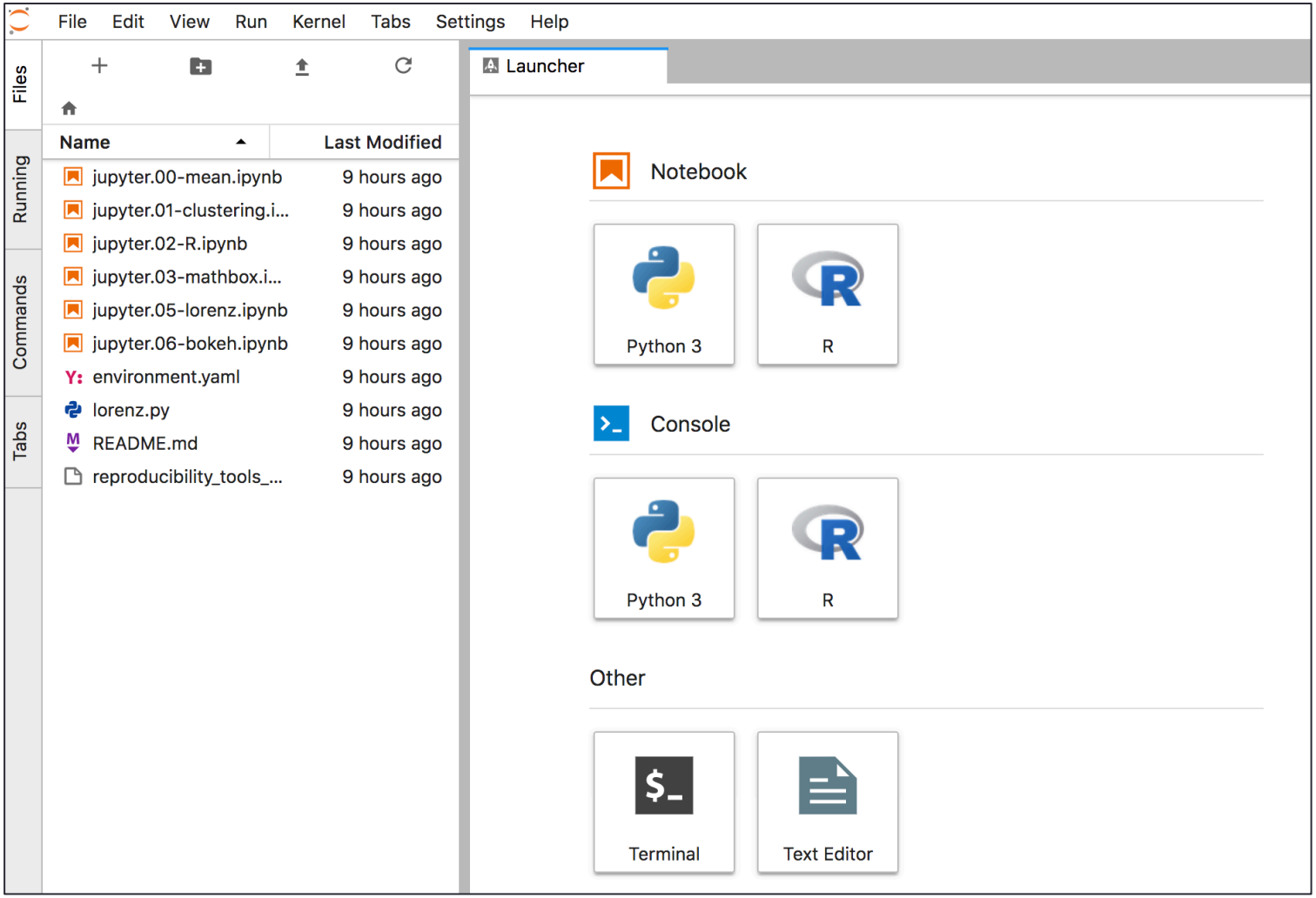
In [1]:
```
1 library(ggplot2)
```

In [2]:
```
1 qplot(factor(cyl), wt, data = mtcars, geom = c("boxplot", "jitter"))
```



R “kernel”

JupyterLab introduces new look and feel and widget architecture



Jupyter ecosystem

- IPython -> Jupyter -> JupyterLab (Lab is still quite new)
- JupyterHub
- Git
- Binder

- Jupyter is not
  - A software IDE
  - A powerful text editor

Agenda

- Introductions
- Reproducibility in theory
- Reproducibility in action
  - Conda
  - Snakemake
  - Jupyter
- Reflections

"Joel test" for computational reproducibility

- Have you repeated these results inside your lab (2nd party)?
- Do you use source control for your code and data workflow?
- Have you shared your data?
- Have you shared your code?
- Have you shared the automated workflow that generates all derived data?
- Have you shared your compute environment?
- Is your code/workflow/data versioned?
- Have you shared the automated tests of your code/workflow?
- Is your code published under an OS license?
- Is your code documented/literate?
- Have you repeated these results outside your lab (3rd party)?

Reproducibility is good

- Reproducibility::Research as Usability::Software
  - Harder
  - Better user/researcher experience
  - Broader adoption

- [Research] advances by extending the number of important operations which we can perform without thinking about them.
  [apologies to] A.N. Whitehead

For more info

- This deck (plus notebooks et al.)
  - https://github.com/cgates/tech_stack_for_reproducibility
- Open Science and Reproducibility
  - *http://ropensci.github.io/reproducibility-guide/*
  - *https://software-carpentry.org/blog/2016/11/reproducibility-reading-list.html*
- Maintainers
  - Conda : https://conda.io/docs/
  - Snakemake: https://snakemake.readthedocs.io/en/stable/
  - Jupyter: http://jupyter.org/
- Software Carpentry
  - https://software-carpentry.org/
  - UM: https://umswc.github.io/

Thanks & Questions