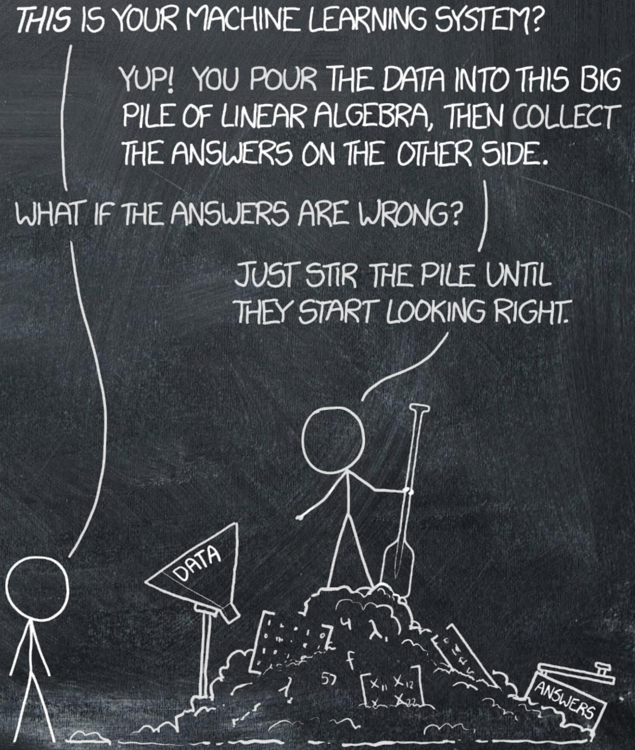
A desperate guide to data analysis workflow improvements

Peter Regner

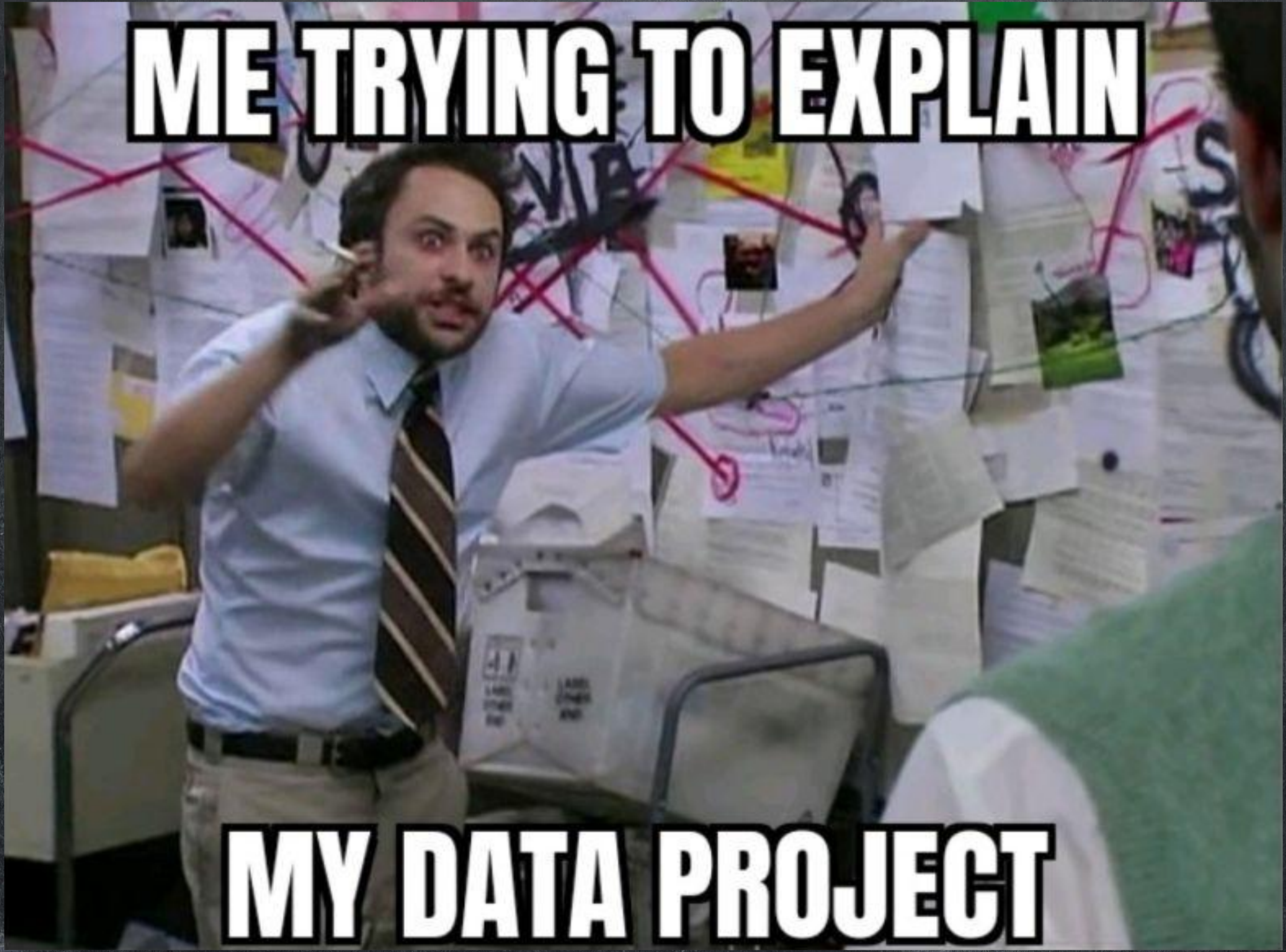
"Reproducible Data Analysis"
Data Science @ BOKU initiative

2024-02-29
BOKU Wien



THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

https://xkcd.com/1838/


ME TRYING TO EXPLAIN
MY DATA PROJECT

Our goal: reproducibility!

...but in practice:

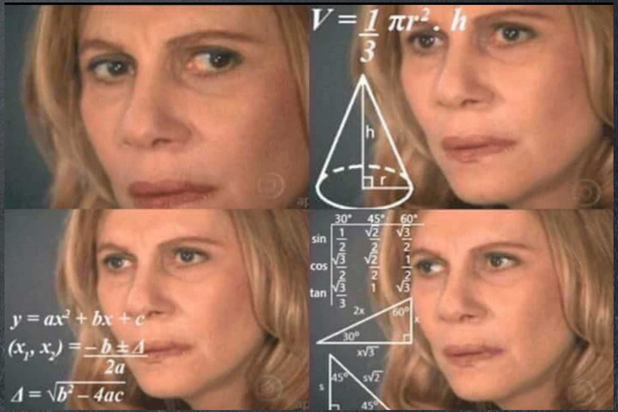
nobody re-runs it or it is broken

irreproducibility by hardware demands

irreproducibility by obscurity

traceability

comprehensibility


$V = \frac{1}{3}\pi r^2 \cdot h$
$y = ax^2 + bx + c$
$(x_1, x_2) = \frac{-b \pm \Delta}{2a}$
$\Delta = \sqrt{b^2 - 4ac}$
sin
cos
tan

some users might be interested mostly in results

Make things more usable - for yourself and others!

Reproducibility in practice: How easily can somebody use the thing later?

Think about your target audience to set priorities!

Do they want to...

- re-run all your scripts?
- run the code as piece of software?
- use the code and adapt it?
- use the computation results only?
- understand what and how we did it?

7 Steps towards more use-able data analysis projects

Gamification: count every ✓, if you follow the guideline already!

(1) Publish your result data sets ✓

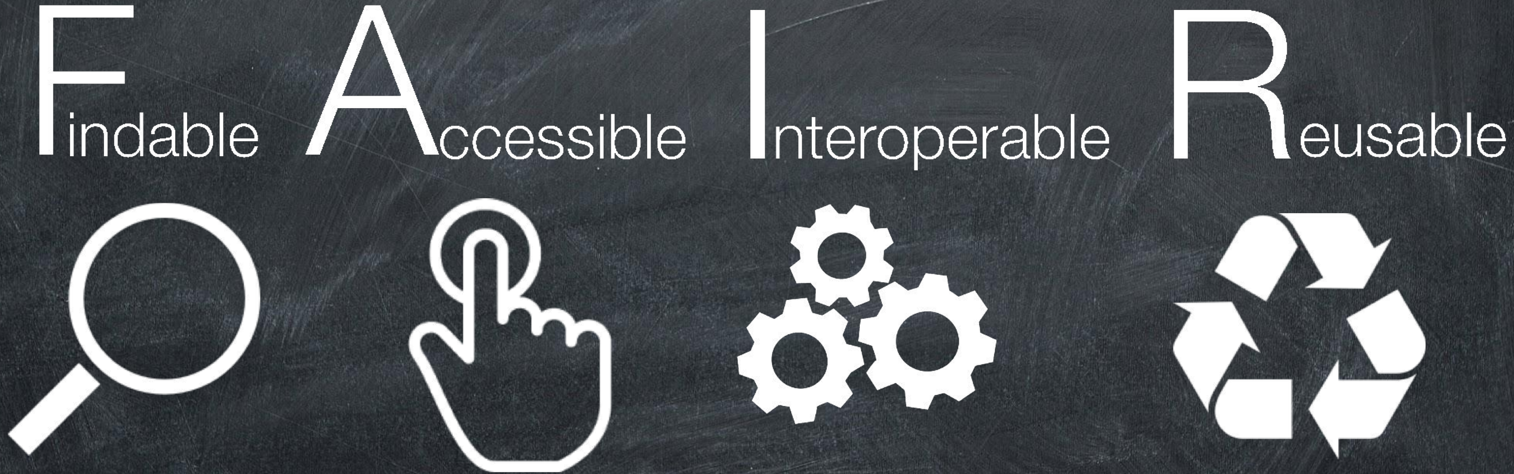
Wicherts, J. M., Bakker, M., & Molenaar, D. (2011). **Willingness to Share Research Data Is Related to the Strength of the Evidence and the Quality of Reporting of Statistical Results**. In R. E. Tractenberg (Ed.), PLoS ONE (Vol. 6, Issue 11, p. e26828). Public Library of Science (PLoS). https://doi.org/10.1371/journal.pone.0026828

Willingness to Share Research Data

is related to

Strength of the Evidence

(1) Publish your result data sets: in a fair way!



https://www.go-fair.org/fair-principles/
https://doi.org/10.1038/sdata.2016.18

(1) Publish your result data sets: how?

✓ Use an open license if possible! (Allows others to publish your results as input data!)

✓ Label and document the fields of your result data set properly

✓ A CSV or NetCDF file on Zenodo is already pretty good

✗ tables in the PDF of the supplementary material

✗ non-open or non-standard file formats like *.gdx (GAMS)

✗ units for quantities not documented

More data providers: Dataverse, FigShare, AUSSDA, Dryad, Mendeley Data, DataHub, DANS, EUDat, ...

(2) Publish your Code ✓

- use an open license, e.g. MIT license or CC-BY ✓ (unless you need GPL libraries)
- For maintained projects: Use GIT / Github ✓
- Include a README file ✓
  - What does it do? (summary & link to article) ✓
  - Requirements: hardware & software ✓
  - How to run / use the thing ✓

(3) Use reproducible virtual environments ✓

Virtual environments (e.g. Conda) allow you to:

- Multiple versions of a library on one machine
- Pretty portable (Windows, Linux, Mac OS)
- Easier to use than Docker

...and to export all used libraries & precise versions!

Alternatives to conda: renv, pipenv, ...

Export all used dependencies using conda: ✓

```
conda env export --no-builds | grep -v "^prefix: " > env.yml
```

Use micromamba - conda is too slow! ✓

https://mamba.readthedocs.io/en/latest/installation/micromamba-installation.html

env.yml:

```
name: my_project
channels:
  - conda-forge
  - defaults
dependencies:
  - ca-certificates=2023.11.17
  - liblapack=3.9.0
  - libzlib=1.2.13
  - ncurses=6.4
  - numpy=1.26.3
  - openssl=3.2.1
  - pip=23.3.2
  - python=3.10.13
     [...]
  - wheel=0.42.0
  - xz=5.2.6
  - zstd=1.5.5
  - pip:
    - cplex==22.1.1.0
```

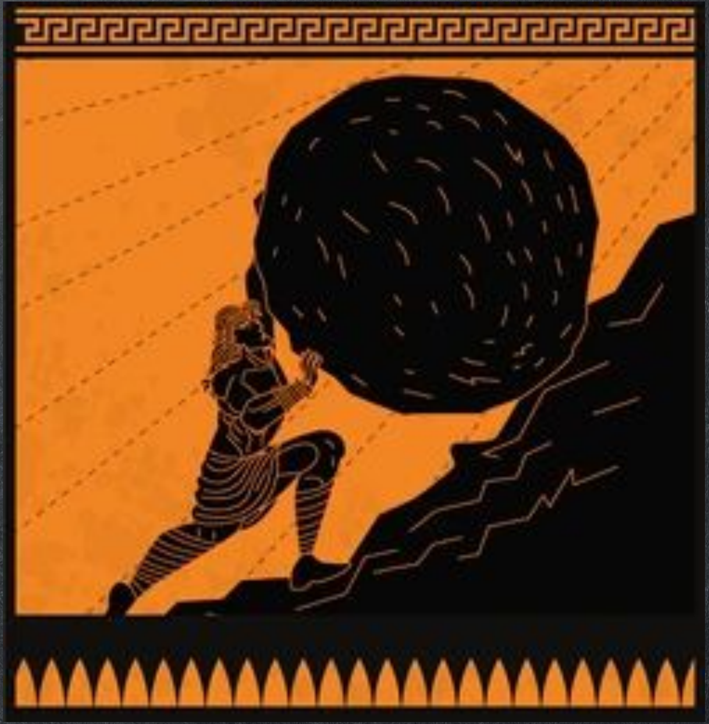
(4) Automate everything

Manual steps are...

- error prone
- time consuming
- not documented

Every command or click should be stored in a way, such that it can be executed again!

Be realistic! You will do things over and over again...



But you can automate all steps!

Please don't take this too literally!

(4) Automate everything

Goal: a single executable script ✓

- download of input data
- preprocessing
- computation
- write results to files
- create figures

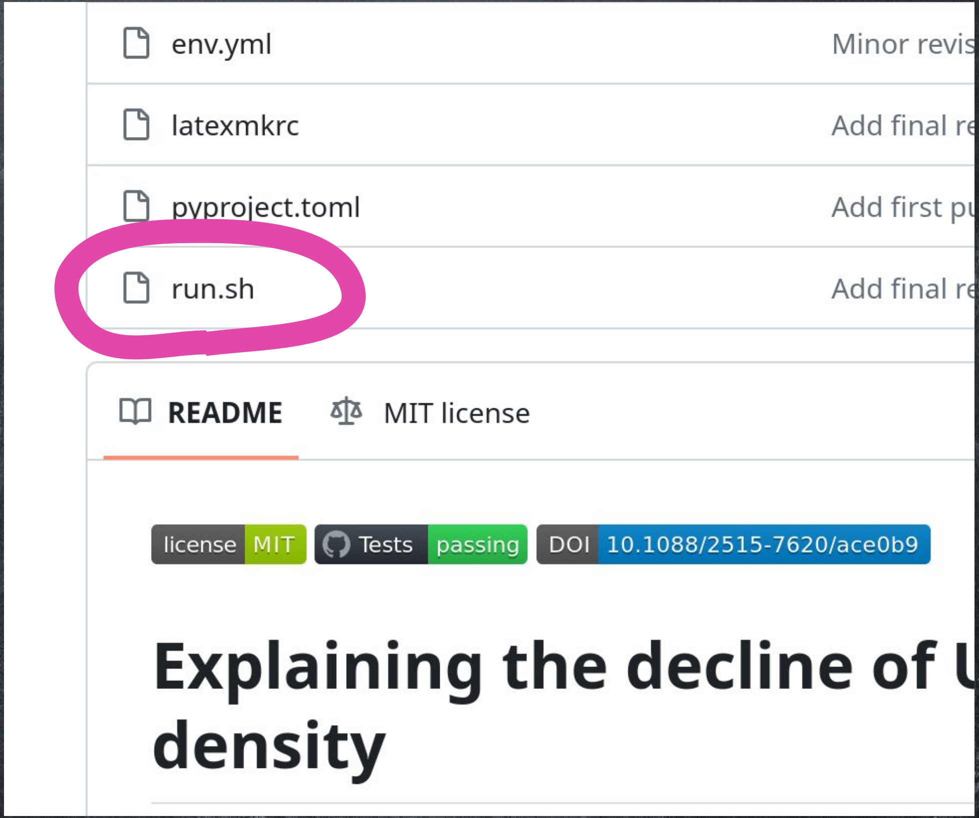
article

LATEX

or

quarto

how to & links in bonus slides


env.yml Minor revis
latexmkrc Add final re
pyproject.toml Add first pu
run.sh Add final re

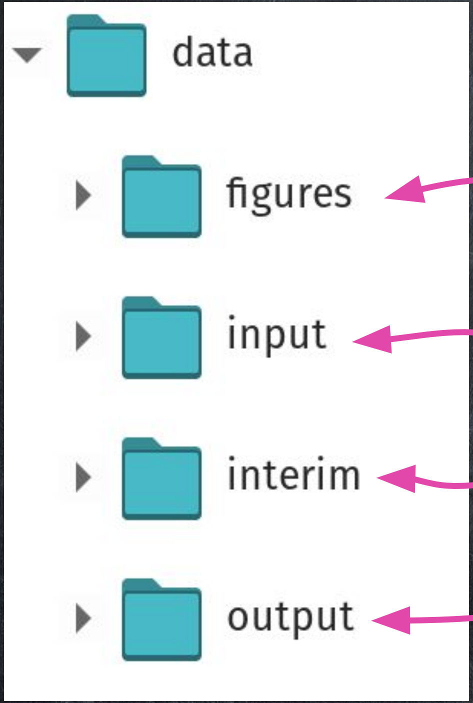
README MIT license

license MIT | Tests passing | DOI 10.1088/2515-7620/ace0b9

Explaining the decline of U density

https://github.com/inwe-boku/windpower-decomposition-usa

(5) Use a good folder structure ✓



ideally PDF files, directly used by LaTeX

raw input files, not modified after download

large files, less important interim results

results to be published

(5) Use a good folder structure

your project

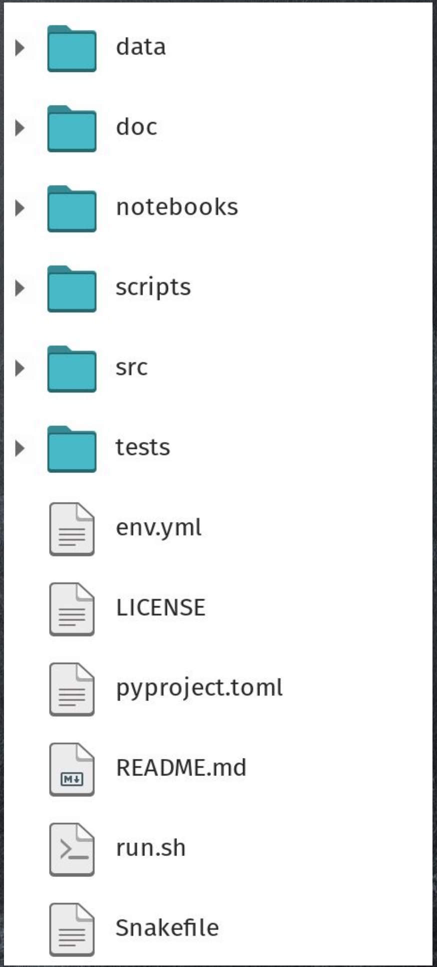
Follow common conventions!

- no spaces in file names ✓
- file names in lower case or ALL_CAPS ✓
- avoid CamelCase for files and folders ✓

auto-create from templates

COOKIECUTTER ✓

https://github.com/cookiecutter/cookiecutter



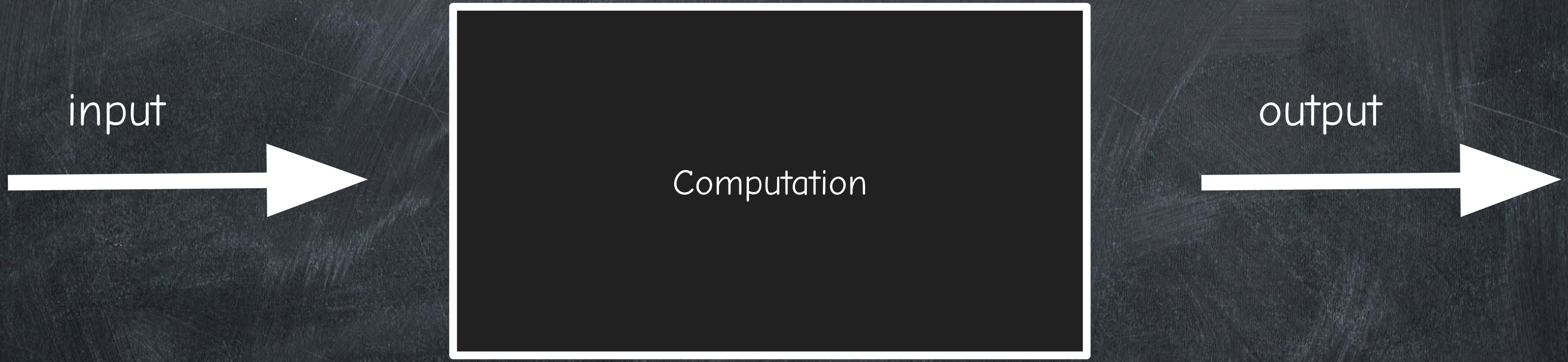
Literature:

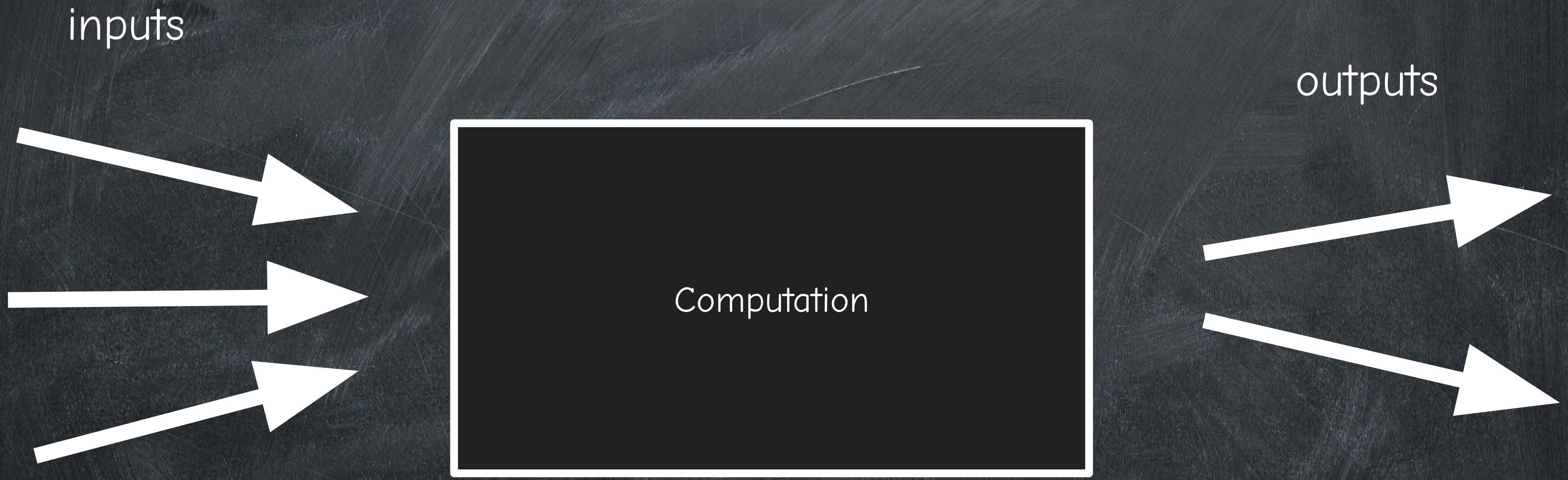
[1] G. Wilson, J. Bryan, K. Cranston, J. Kitzes, L. Nederbragt, and T. K. Teal, "**Good enough practices in scientific computing,**" PLOS Computational Biology, vol. 13, no. 6, p. e1005510, Jun. 2017, doi: 10.1371/journal.pcbi.1005510.

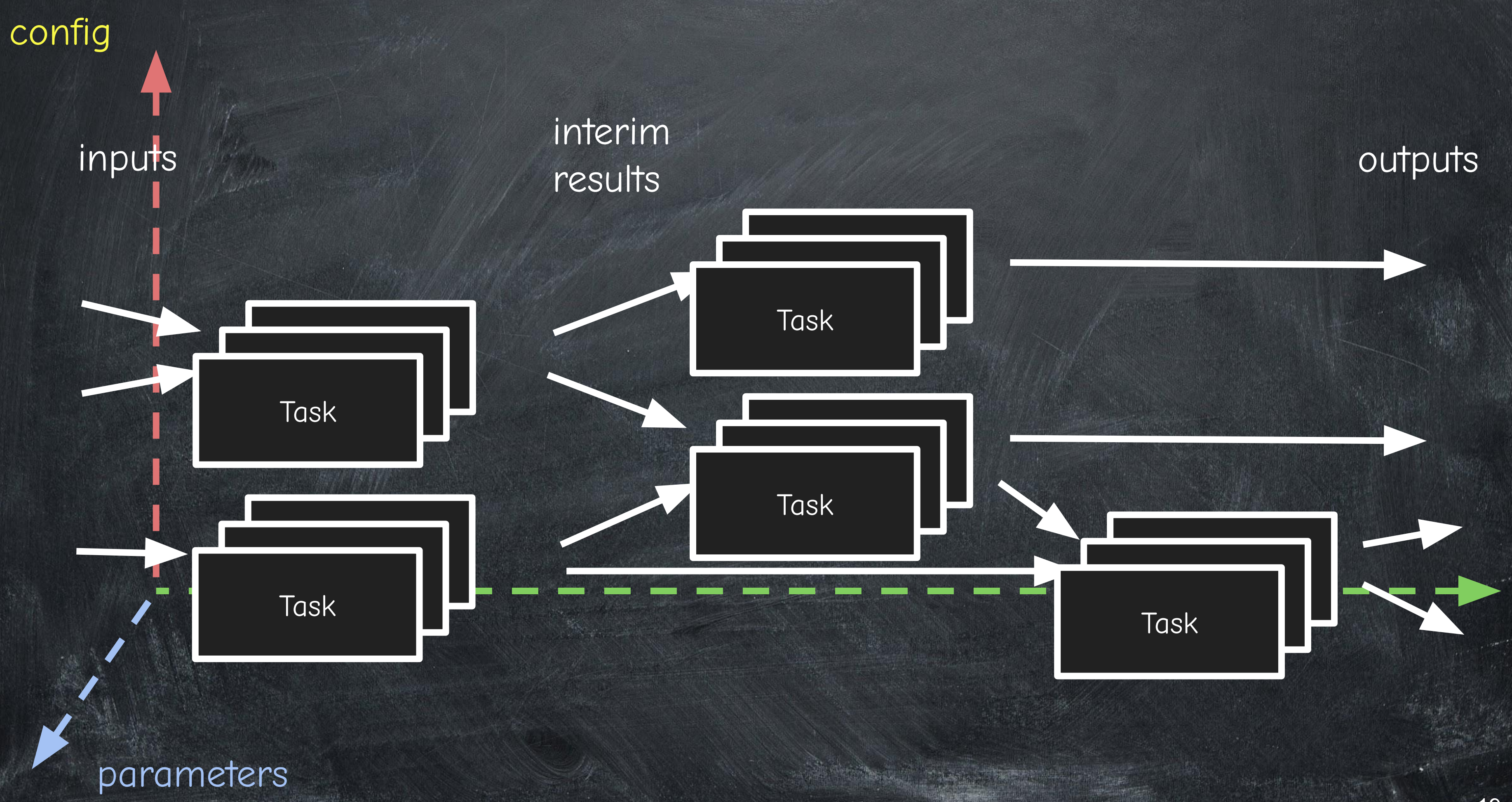
[2] W. S. Noble, "**A Quick Guide to Organizing Computational Biology Projects,**" PLOS Computational Biology, vol. 5, no. 7, p. e1000424, Jul. 2009, doi: 10.1371/journal.pcbi.1000424.

[3] "**Cookiecutter Data Science.**" https://drivendata.github.io/cookiecutter-data-science/

Organize your computation pipeline


input
Computation
output


inputs
Computation
outputs


config
inputs
interim results
outputs
Task
Task
Task
Task
Task
parameters

(6) Use a pipeline tool ✓

- Caching / incremental builds
  → skip tasks if nothing changed (inputs, code, config, parameters)

- Parametrizing tasks
  → specify (multi dimensional) parameter spaces via ranges or lists

- Run tasks in parallel
  → on different CPUs or on different nodes on a cluster (e.g. VSC)

Ruffus Rabix KNIME Analytics Platform Fission Workflows Polynote Gwf
Dask Sushi Stpipe ActivePapers Workflow Patterns Initiative
Reflow Pydra Common Workflow Language
OpenMOLE DSL Mario Remake Anduril Yet Another Workflow Language Workflow Patterns Library Redun
pyperator Zeppelin Flyte
TaskGraph Factual's Drake SciLuigi Pentaho Kettle Makeflow Arvados Balsam noWorkflow
Sake Adage Joblib Antha Argo
LinkedPipes ETL Swift Just Snakemake
pyppl Cadence Drake R package Nextflow Clusterjob Pinball
Compss ecFlow Zenaton VisTrails Seqtools Bistro Ophidia
Mistral Workflow Description Language Substation Consecution
Cosmos
NiPype BioMake Moa Dagster Bioshake Luigi Signac Digdag Domino Quilt OpenGE
Ruigi
Pathomx d6tflow Ketrew
Kepler Clubber Cluster Flow Pegasus PipEngine Toil
SoS Orchest Hamilton Kestra ResearchObject.org
ActionChain Martian Dbt Apache Iravata Workflow 4 Ever workflow research object model pypyr AWE
Watchdog Sundial Workflow 4 Ever Initiative Conan2
Galaxy Targets Reprozip SciPipe
DataJoint doit
Dockerflow Yap Shake Mara Drake
Qsubsec Binder Porcupine Parsl Tibanna Wallaroo
lakeFS APE Scons HyperLoom Cromwell Wings
Longbow Spiff MetaFlow
Bazel Compi FlowHub Couler Dataform Bpipe
BioQueue Rubra SmartPipeline
Ploomber Nessie Stolos Sure DVC Steppy Yabi ZenML Yapp
omega|ml DataOps Platform AiiDA StreamPipes RedPoint Notebooks Ray
Rain Fireworks Briefly Kedro NextflowWorkbench Kronos WorldMake Polyaxon OpenMOLE
Reana CDS Qdo Loom PyFlow Popper Flex Kiba ETL Prodmodel
Gc3pie Maestro Temporal Guix Workflow Language
Chipster Covalent Piper StreamFlow Arteria Flowr
Pachyderm Make DataLad Dagr
Butler Scoop Cloudgene Workflow Language Anvi'o
Rmake Dray
Cuneiform Taverna Prefect Gradle
HPC Grid Runner MD Studio

https://github.com/pditommaso/awesome-pipeline

(7) Keep old versions of your data ✓

Re-running computations might be difficult or take long time... Keep old versions!

It would be nice to...

- know how artefacts (figures, output files, ...) were generated
  → record Git hash / code, all inputs, parameters, conda environment, config, ...
- delete large files later when disk space is an issue
- compare between different versions of data

Git alone is not enough to fulfill these requirements!

Note that Git offers more features:

- Sync to other machines / collaboration with others
- Publish the repository - but data is not always publishable due to licence issues

Tools for large data: DVC, GIT-LFS, git-annex, DataLad, ...

My poor-man data versioning

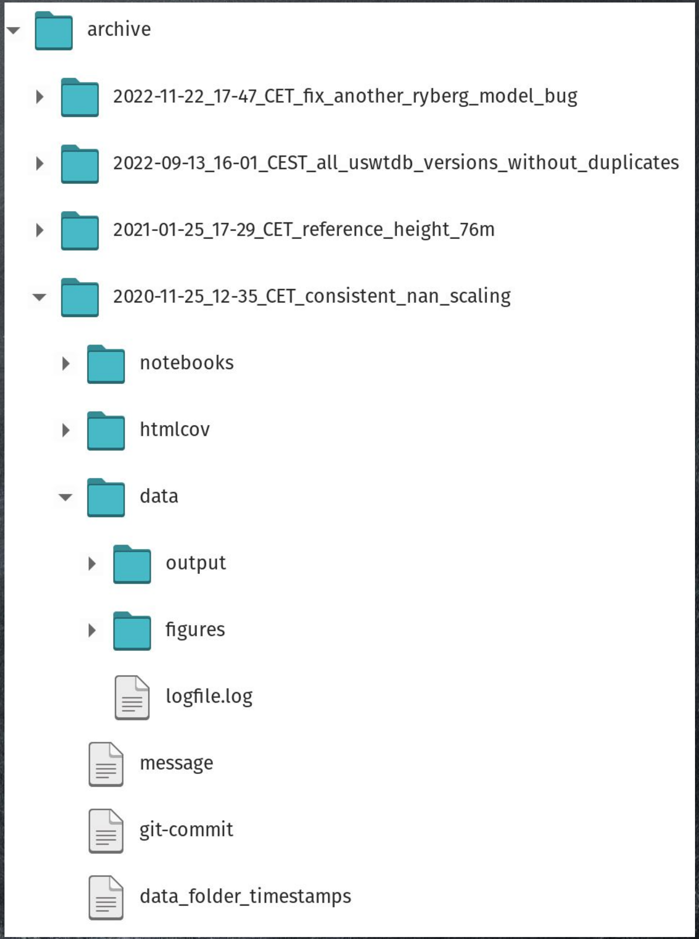
`run.sh` script:

- runs the computation
- creates a Git tag
- copies outputs to an archive folder

Open issues:

- How to sync data between machines? Use Unison?
- How to collaborate with others using the same data?
- How to sync figures to Overleaf without Git?

https://github.com/inwe-boku/windpower-decomposition-usa/blob/main/run.sh



```
archive
  2022-11-22_17-47_CET_fix_another_ryberg_model_bug
  2022-09-13_16-01_CEST_all_uswtdb_versions_without_duplicates
  2021-01-25_17-29_CET_reference_height_76m
  2020-11-25_12-35_CET_consistent_nan_scaling
    notebooks
    htmlcov
    data
      output
      figures
      logfile.log
    message
    git-commit
    data_folder_timestamps
```

Cookies for all of you!

Everything we have seen today in a cookiecutter template:

- folder structure
- code snippet for including results in LaTeX
- Script for building a zip file for arXiv.org
- run.sh script
- Snakemake file
- README.md, .gitignore, LICENSE, ...
- ...

→ https://github.com/inwe-boku/cookiecutter-data-research

Reproduce all the things!

Download slides:

https://bit.ly/desperate-guide

peter.regner@boku.ac.at

https://github.com/inwe-boku/

https://refuel.world/

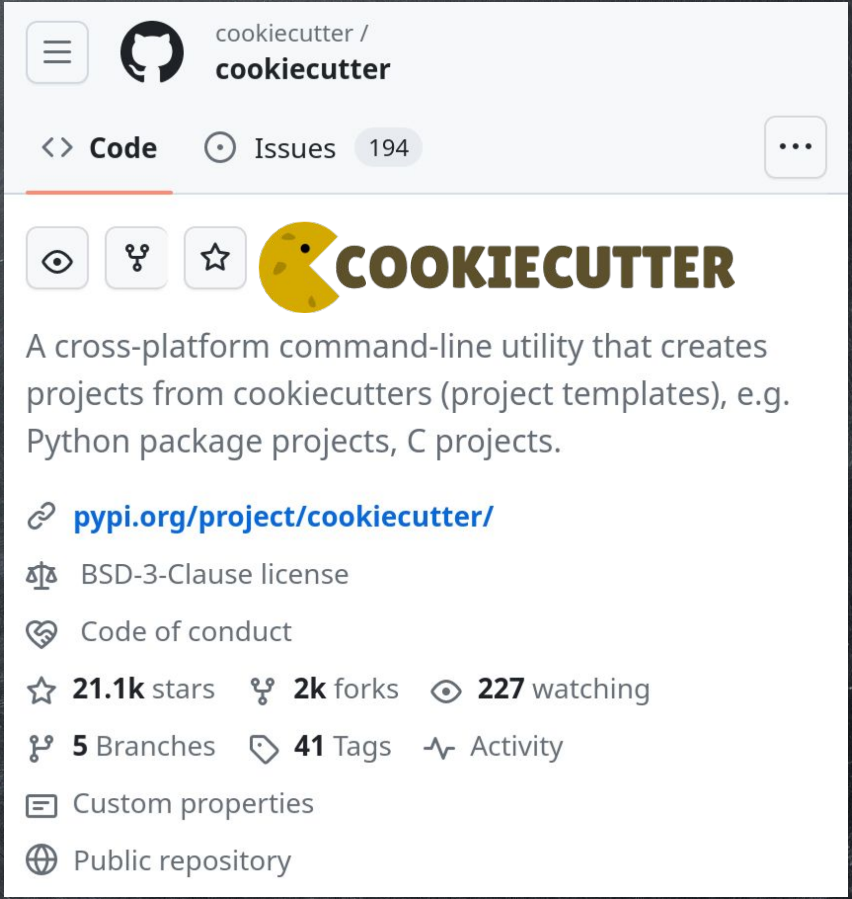
We gratefully acknowledge support from the European Research Council ("reFUEL" ERC-2017-STG 758149).

Automate project creation ✓

Create a new folder from a template:

- folder structure
- LICENSE file
- .gitignore file
- code snippets
- ...

https://github.com/cookiecutter/cookiecutter



A cross-platform command-line utility that creates projects from cookiecutters (project templates), e.g. Python package projects, C projects.

pypi.org/project/cookiecutter/

BSD-3-Clause license

Code of conduct

21.1k stars 2k forks 227 watching

5 Branches 41 Tags Activity

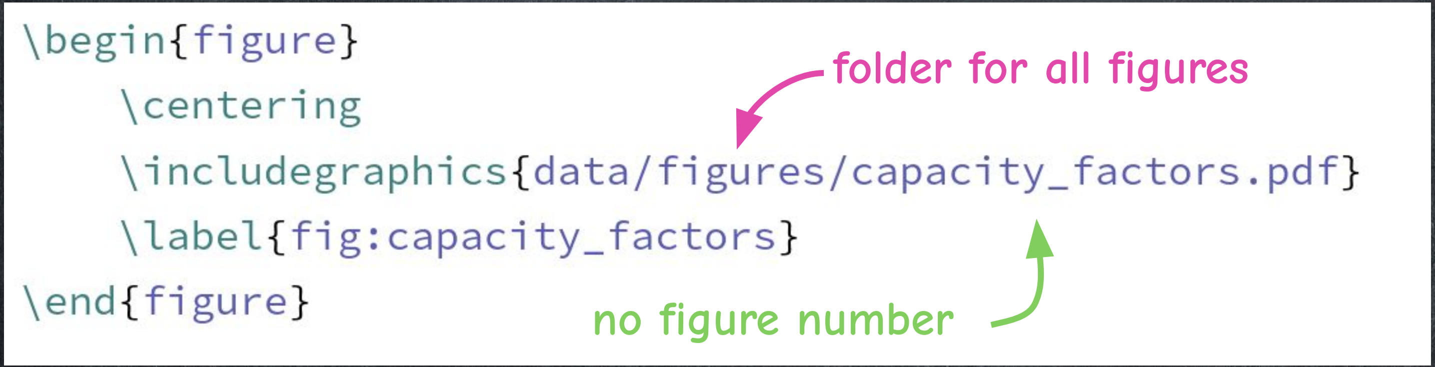
Custom properties

Public repository

Include figures in LaTex documents ✓

MS Word: use linked images

Latex:



```
\begin{figure}
    \centering
    \includegraphics{data/figures/capacity_factors.pdf}
    \label{fig:capacity_factors}
\end{figure}
```

Overleaf: use GIT, Github or dropbox to sync figures - premium only… :-/

https://www.overleaf.com/learn/how-to/Dropbox_Synchronization https://www.overleaf.com/learn/how-to/Git_Integration_and_GitHub_Synchronization

Include results in LaTex documents ✓

Turns out to be surprisingly difficult! 😱

```
result_values = {}

meaning_of_life = 42
result_values['meaning_of_life'] = f'{meaning_of_life:d}'

gravity_ms2 = 9.80665
result_values['gravity'] = f'{gravity_ms2:.2f}', 'm/s^2'

write_result_values(result_values)
```

script writes results to a file with custom LaTeX commands

```
\newcommand{\meaning_of_life}{42}
\newcommand{\gravity}{\qty{9.81}{m/s^2}}
```

auto-generated LaTeX commands

More details and other Solutions like Knitr:
https://tex.stackexchange.com/a/711627/8964

Do more good things

- Write good code
  - Don't do no magic numbers ✓
  - No absolute file paths ✓
  - No unnecessary Abbrev. ✓
  - Follow code conventions (e.g. pep8) ✓
- Write unit / functional tests ✓

  Automatically check if things are behaving as expected!
- Use a code linter (e.g. as GIT hook) ✓

  Tools to automatically check code for errors and style violations, for Python: flake8, black (auto-formatter),
- Use Continuous integration ✓

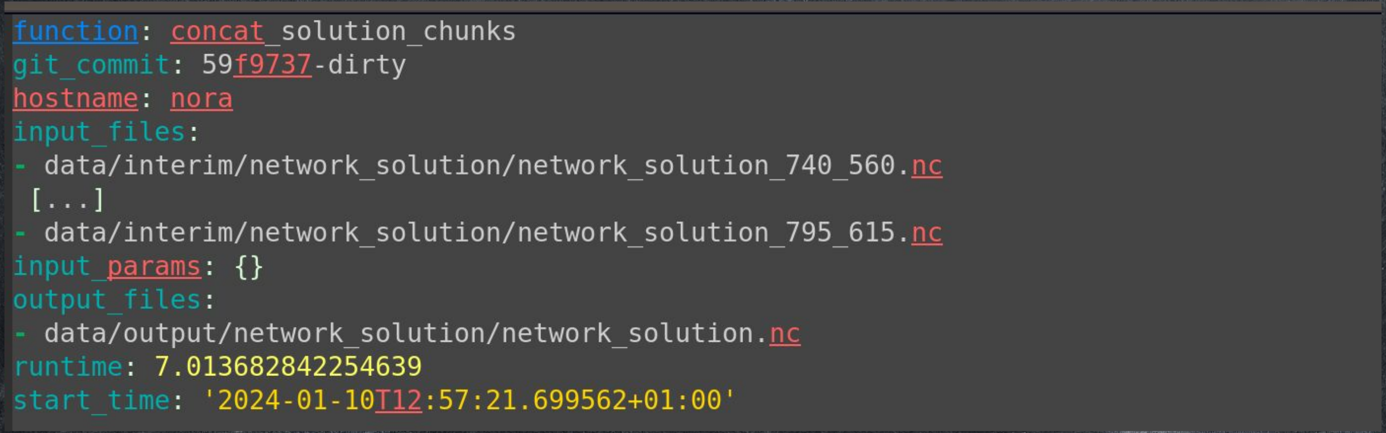
  Run tests, code linter or computations automatically, e.g. via Github Actions
- Do code review ✓

  A great way to improve quality and spread knowledge in your team!

Tracking additional computation information

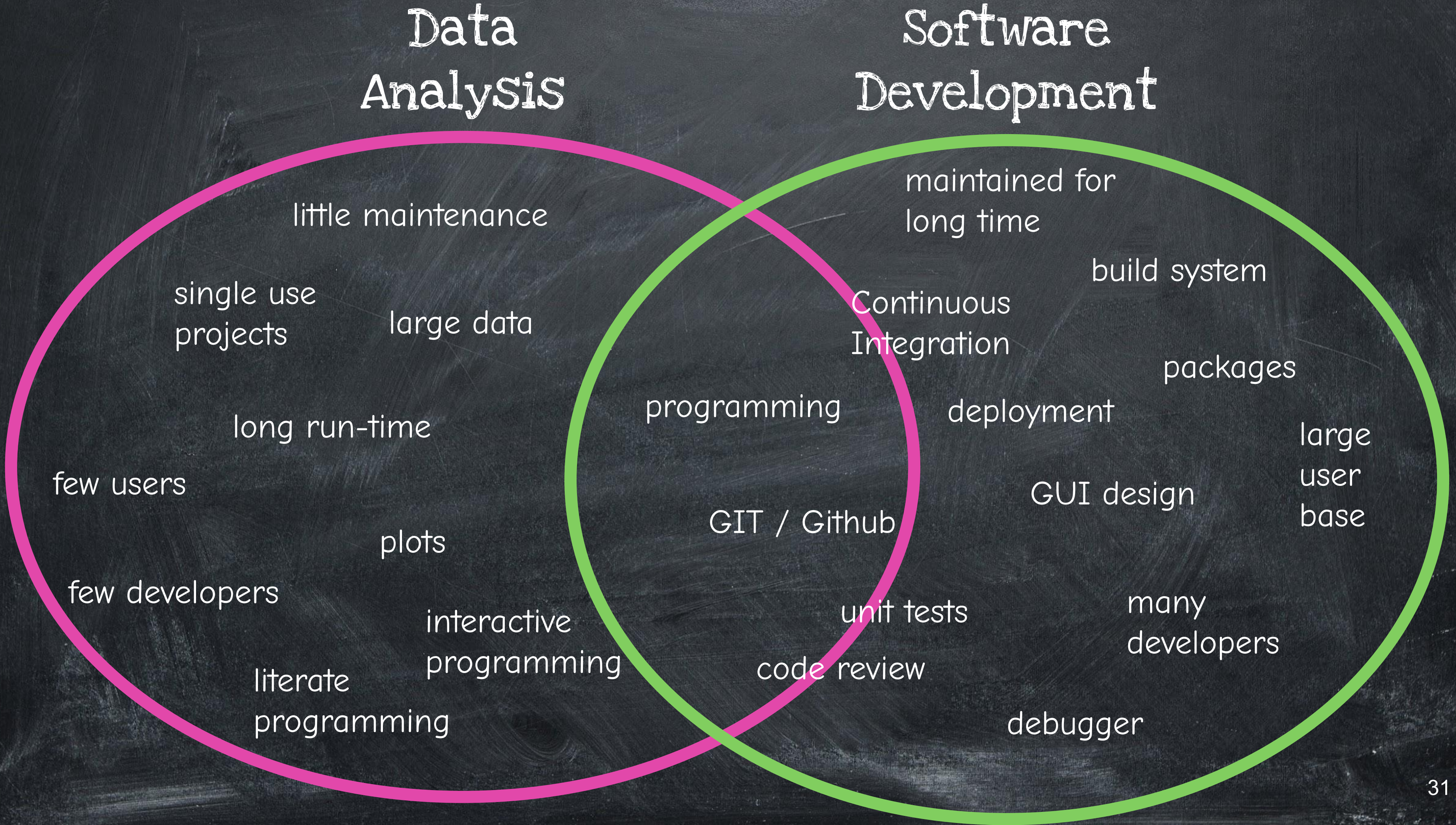
```
@task
def my_fancy_func(some_param, inputs, outputs):
    # do something here
    ...
```

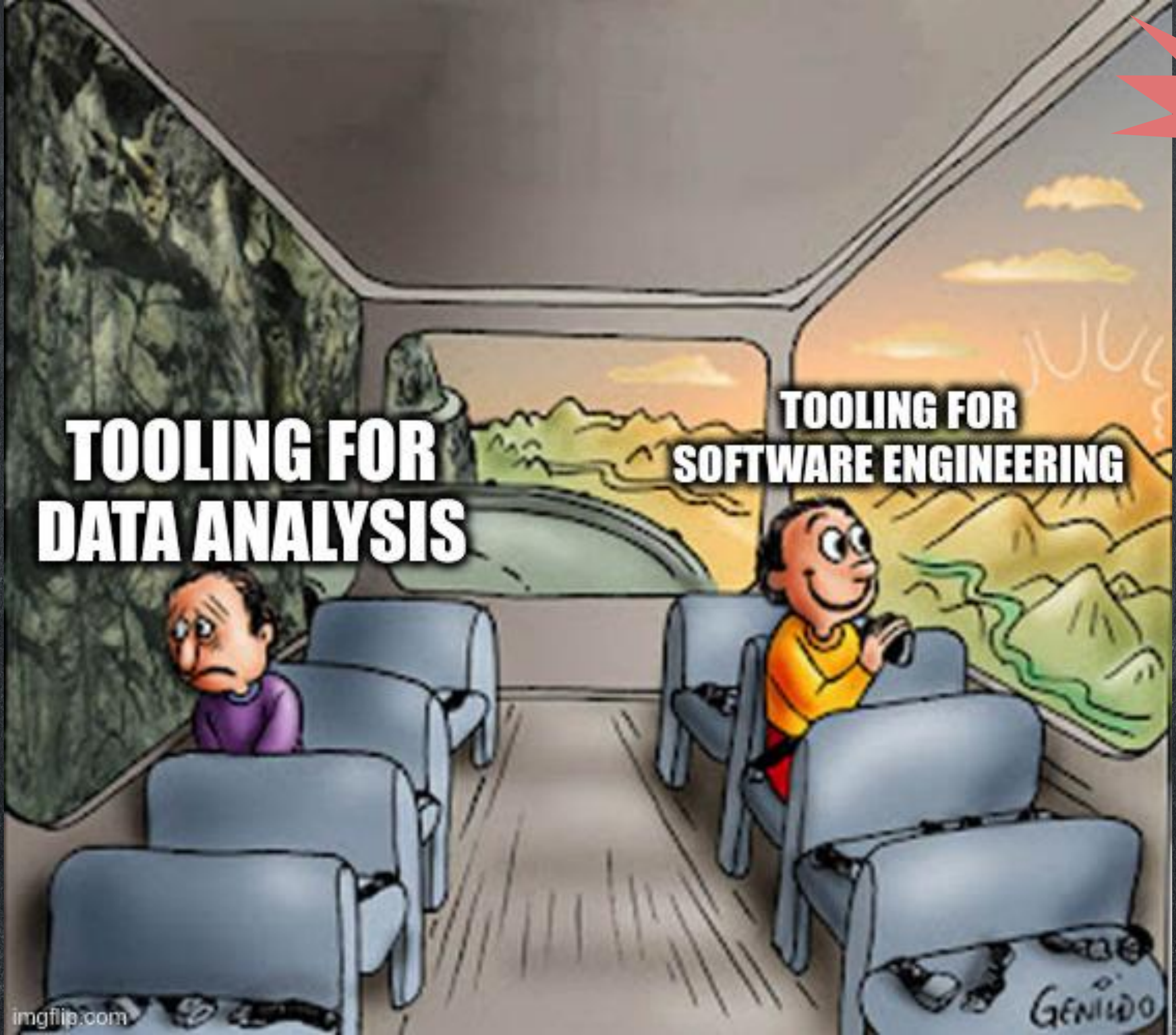
The `@task` decorator creates a file with metaparameters for each output:



```
function: concat_solution_chunks
git_commit: 59f9737-dirty
hostname: nora
input_files:
- data/interim/network_solution/network_solution_740_560.nc
 [...]
- data/interim/network_solution/network_solution_795_615.nc
input_params: {}
output_files:
- data/output/network_solution/network_solution.nc
runtime: 7.013682842254639
start_time: '2024-01-10T12:57:21.699562+01:00'
```

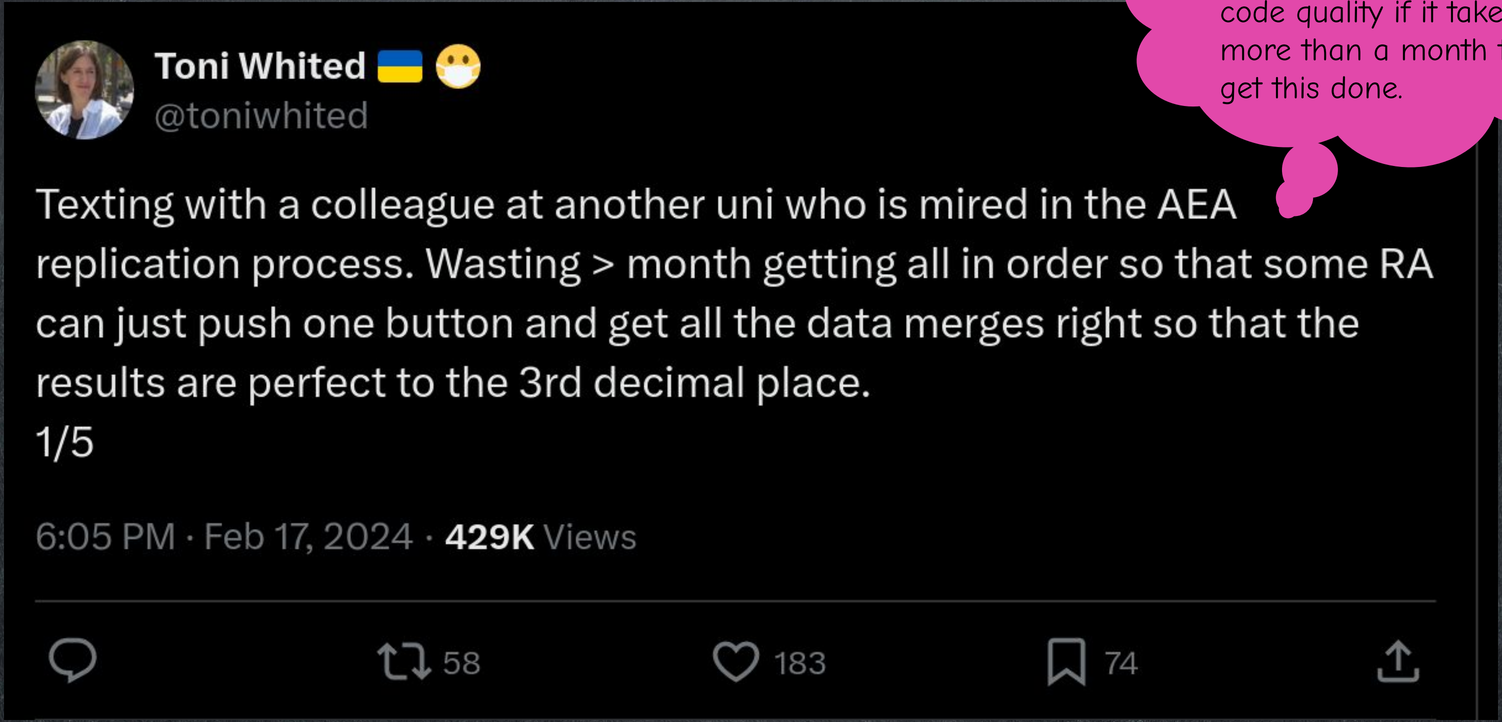
Link to source: https://github.com/inwe-boku/cookiecutter-data-research/blob/main/%7B%7B%20cookiecutter.repo_name%20%7D%7D/src/task.py


Data Analysis
Software Development
little maintenance
single use projects
large data
long run-time
few users
plots
few developers
interactive programming
literate programming
programming
GIT / Github
unit tests
code review
maintained for long time
build system
Continuous Integration
packages
deployment
large user base
GUI design
many developers
debugger



opinionated!

Reproducibility is a hot topic...



Toni Whited 🇺🇦 😷
@toniwhited

Texting with a colleague at another uni who is mired in the AEA replication process. Wasting > month getting all in order so that some RA can just push one button and get all the data merges right so that the results are perfect to the 3rd decimal place.
1/5

6:05 PM · Feb 17, 2024 · **429K** Views

58 183 74

It should not be the only priority, but I'd be a bit skeptical about code quality if it takes more than a month to get this done.

https://twitter.com/toniwhited/status/1758900448596291953