Reproducible workflows in research computing

Lecture 11

MARN-5895, Fall 2021

As scientists who depend on computer code to generate, analyze and visualize results, we are all **professional programmers**, so we should take coding seriously (just like we take our lab experiments and our writing seriously).

Dumb programming mistakes often happen at the highest levels of scientific research —>

22 Aug 2018 in Research & Technology

The war over supercooled water

How a hidden coding error fueled a seven-year dispute between two of condensed matter's top theorists.

Ashley G. Smart

789 14 COMMENTS TOOLS PREV NEXT

Reproducible research

- “Authors provide all necessary information, including data and code, necessary for the replication and reproduction of relevant results” (Schwab et al, 2020).
- “The success and credibility of Science are anchored in the willingness of scientists to explore their ideas and results to **independent testing** and **replication by others**. This requires the open exchange of data, procedures and materials” (American Physical Association, Ethics & Values, 1999, https://www.aps.org/policy/statements/99_6.cfm)

In 1991, Jon Claerbout (Stanford University) began requiring graduate theses to conform to a standard of reproducibility. Some consider him the founding father of computational reproducibility in geosciences.

Schwab, Matthias, N. Karrenbach, and Jon Claerbout. "Making scientific computations reproducible." *Computing in Science & Engineering* 2.6 (2000): 61-67.

Ten simple rules for computational research

(adapted from Sandve et al., 2013)

Sandve, G. K. et al. (2013), Ten simple rules for reproducible computational research, PLOS Comp. Bio. (editorial), Vol. 9(10):1–4, doi: 10.1371/journal.pcbi.1003285

1

For every simple result, keep track of how it was produced

- How were the data collected, generated, pre-processed QCed?
- Keep a findable set of notes, with links to cruise/lab reports, model control files, etc.

Joint Ocean Ice Study /
Beaufort Gyre Exploration Project
2020 Cruise Report

Photo by Darcy McCabe

Report on the oceanographic research conducted aboard the *CCGS Louis S. St-Laurent,* September 14 to October 2, 2020
(Full cruise dates September 3rd to October 15th, 2020)
IOS Cruise ID 2020-79

Chief Scientist:
Sarah Zimmermann, Fisheries and Oceans Canada,
Institute of Ocean Sciences, Sidney, BC

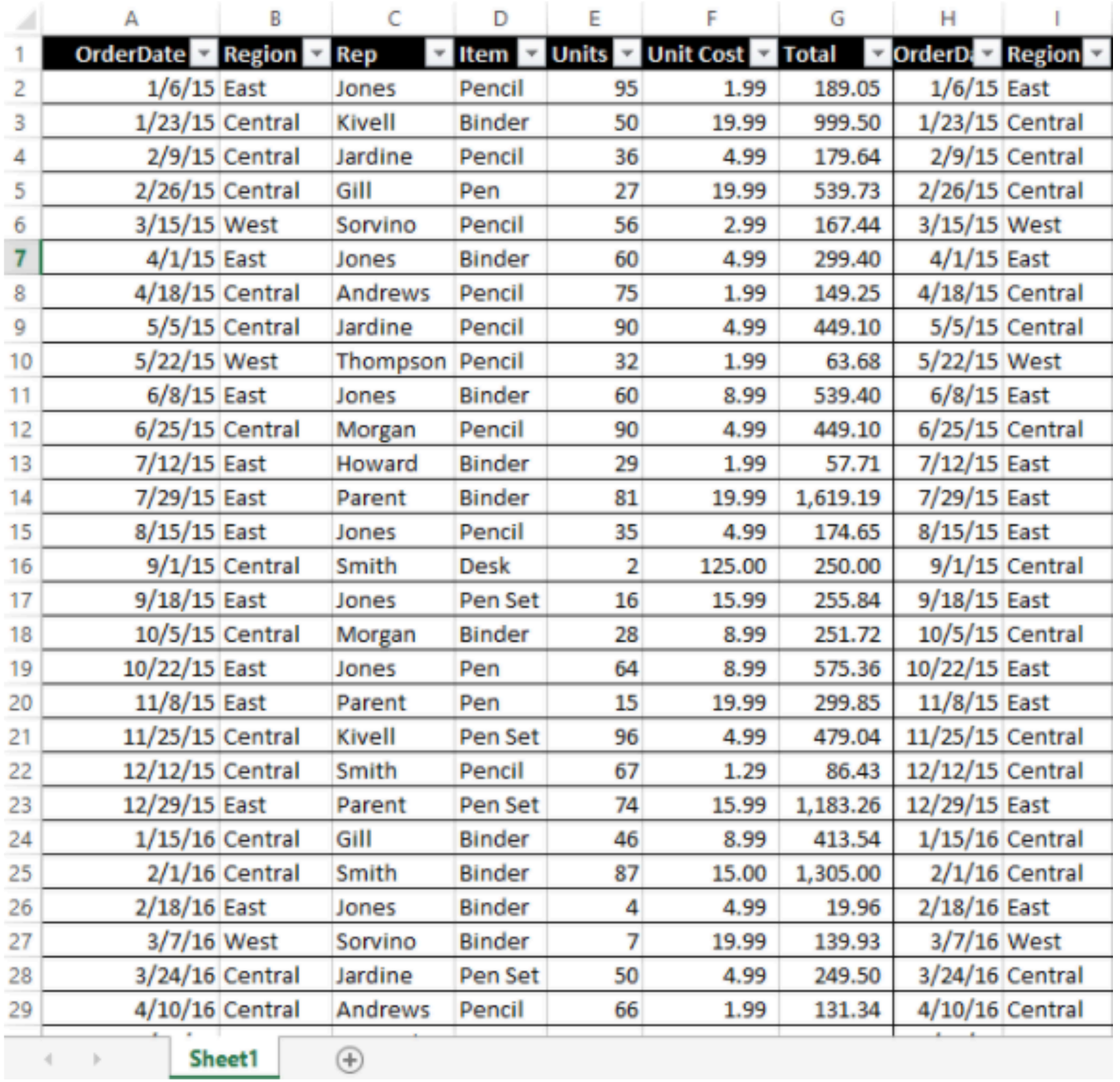
Principle Investigators:
Bill Williams, Fisheries and Oceans Canada
Andrey Proshutinsky, Richard Krishfield and Isabela Le Bras, Woods Hole Oceanographic Institution, USA
Mary-Louise Timmermans, Yale University, USA

PrinciPAL not PrinciPLE

2

Avoid manual data manipulation

- Use Python or your favorite programming language to make algorithmic correction to data, so that the changes can be replicated and reversed.
- In the extreme case that manual changes are necessary, the exact change and the reasoning behind it should be documented.



	A	B	C	D	E	F	G	H	I
1	OrderDate	Region	Rep	Item	Units	Unit Cost	Total	OrderD	Region
2	1/6/15	East	Jones	Pencil	95	1.99	189.05	1/6/15	East
3	1/23/15	Central	Kivell	Binder	50	19.99	999.50	1/23/15	Central
4	2/9/15	Central	Jardine	Pencil	36	4.99	179.64	2/9/15	Central
5	2/26/15	Central	Gill	Pen	27	19.99	539.73	2/26/15	Central
6	3/15/15	West	Sorvino	Pencil	56	2.99	167.44	3/15/15	West
7	4/1/15	East	Jones	Binder	60	4.99	299.40	4/1/15	East
8	4/18/15	Central	Andrews	Pencil	75	1.99	149.25	4/18/15	Central
9	5/5/15	Central	Jardine	Pencil	90	4.99	449.10	5/5/15	Central
10	5/22/15	West	Thompson	Pencil	32	1.99	63.68	5/22/15	West
11	6/8/15	East	Jones	Binder	60	8.99	539.40	6/8/15	East
12	6/25/15	Central	Morgan	Pencil	90	4.99	449.10	6/25/15	Central
13	7/12/15	East	Howard	Binder	29	1.99	57.71	7/12/15	East
14	7/29/15	East	Parent	Binder	81	19.99	1,619.19	7/29/15	East
15	8/15/15	East	Jones	Pencil	35	4.99	174.65	8/15/15	East
16	9/1/15	Central	Smith	Desk	2	125.00	250.00	9/1/15	Central
17	9/18/15	East	Jones	Pen Set	16	15.99	255.84	9/18/15	East
18	10/5/15	Central	Morgan	Binder	28	8.99	251.72	10/5/15	Central
19	10/22/15	East	Jones	Pen	64	8.99	575.36	10/22/15	East
20	11/8/15	East	Parent	Pen	15	19.99	299.85	11/8/15	East
21	11/25/15	Central	Kivell	Pen Set	96	4.99	479.04	11/25/15	Central
22	12/12/15	Central	Smith	Pencil	67	1.29	86.43	12/12/15	Central
23	12/29/15	East	Parent	Pen Set	74	15.99	1,183.26	12/29/15	East
24	1/15/16	Central	Gill	Binder	46	8.99	413.54	1/15/16	Central
25	2/1/16	Central	Smith	Binder	87	15.00	1,305.00	2/1/16	Central
26	2/18/16	East	Jones	Binder	4	4.99	19.96	2/18/16	East
27	3/7/16	West	Sorvino	Binder	7	19.99	139.93	3/7/16	West
28	3/24/16	Central	Jardine	Pen Set	50	4.99	249.50	3/24/16	Central
29	4/10/16	Central	Andrews	Pencil	66	1.99	131.34	4/10/16	Central

Sheet1

3

Archive the exact versions of all software used

- In Python, work with conda environments.
- Keep a copy of the original .yml file.
- At the end of a project, export the versions of all python packages in the environment and their dependencies.

```
(marn5895) [cer19004@cn01 ~]$ conda env export
name: marn5895
channels:
  - conda-forge
  - defaults
dependencies:
  - _libgcc_mutex=0.1=conda_forge
  - _openmp_mutex=4.5=1_gnu
  - aiohttp=3.7.4.post0=py39h3811e60_0
  - alabaster=0.7.12=py_0
  - alsa-lib=1.2.3=h516909a_0
  - anyio=3.3.0=py39hf3d152e_0
  - appdirs=1.4.4=pyh9f0ad1d_0
  - argon2-cffi=20.1.0=py39h3811e60_2
  - argopy=0.1.6=pyhd8ed1ab_1
  - asciitree=0.3.3=py_2
  - async-timeout=3.0.1=py_1000
  - async_generator=1.10=py_0
  - attrs=21.2.0=pyhd8ed1ab_0
  - babel=2.9.1=pyh44b312d_0
  - backcall=0.2.0=pyh9f0ad1d_0
  - backports=1.0=py_2
  - backports.functools_lru_cache=1.6.4=pyhd8ed1ab_0
  - bleach=4.1.0=pyhd8ed1ab_0
  - bokeh=2.3.3=py39hf3d152e_0
  - brotlipy=0.7.0=py39h3811e60_1001
  - bzip2=1.0.8=h7f98852_4
  - c-ares=1.17.2=h7f98852_0
  - ca-certificates=2021.5.30=ha878542_0
  - cached-property=1.5.2=hd8ed1ab_1
```

4

Version-control all custom scripts

- Use git to keep track of relevant changes to code.
- Push the code to Github for easy tracking and back-up.
- If working on a major change, create a git branch.
- When a few major changes are implemented, “release a version” on Github.

STAGE & SNAPSHOT

Working with snapshots and the Git staging area

`git status`	show modified files in working directory, staged for you
`git add [file]`	add a file as it looks now to your next commit (stage)
`git reset [file]`	unstage a file while retaining the changes in working dir
`git diff`	diff of what is changed but not staged
`git diff --staged`	diff of what is staged but not yet committed
`git commit -m “[descriptive message]”`	commit your staged content as a new commit snapshot

SETUP & INIT

Configuring user information, initializing and cloning repositories

`git init`	initialize an existing directory as a Git repository
`git clone [url]`	retrieve an entire repository from a hosted location via URL

BRANCH & MERGE

Isolating work in branches, changing context, and integrating changes

`git branch`	list your branches. a * will appear next to the currently active branch
`git branch [branch-name]`	create a new branch at the current commit
`git checkout`	switch to another branch and check it out into your working directory
`git merge [branch]`	merge the specified branch’s history into the current one
`git log`	show all commits in the current branch’s history

SHARE & UPDATE

Retrieving updates from another repository and updating local repos

`git remote add [alias] [url]`	add a git URL as an alias
`git fetch [alias]`	fetch down all the branches from that Git remote
`git merge [alias]/[branch]`	merge a remote branch into your current branch to bring it up to date
`git push [alias] [branch]`	Transmit local branch commits to the remote repository branch
`git pull`	fetch and merge any commits from the tracking remote branch

5

Record all intermediate results, if possible in standard formats

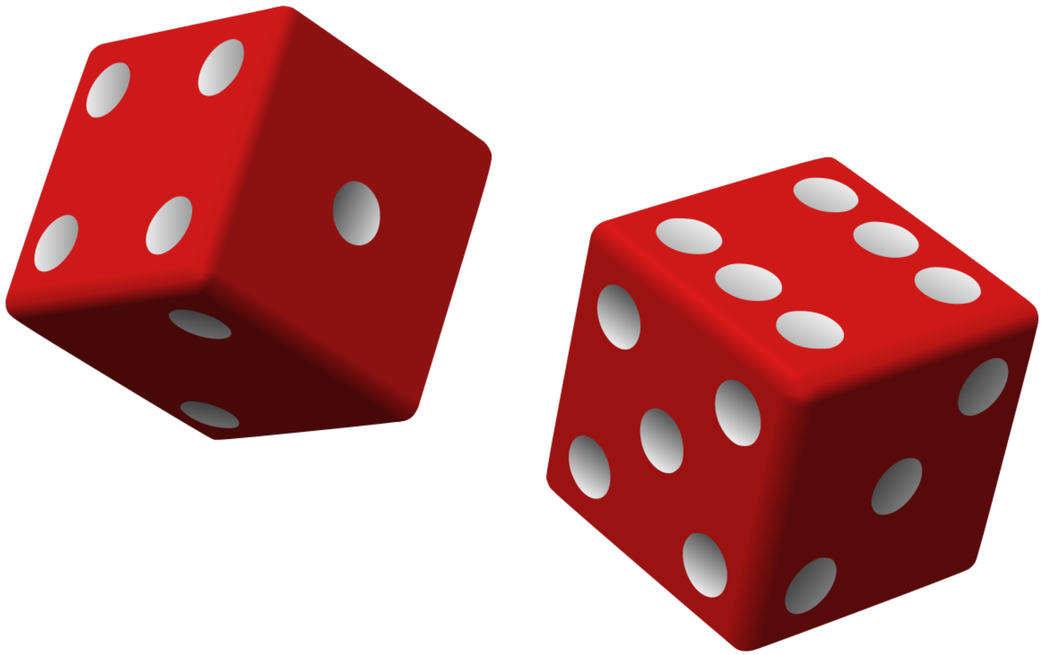
- If workflow involves several scripts, with inputs and outputs, export the intermediate files with appropriate file names.
- Prioritize netCDF files over npz (Python) or mat (Matlab) binary files.

```
# Load data from cruise

data = pd.read_csv('MAB_CruiseData.csv')

# Run specific function to calculate some statistics

stats = CalculateBasicStatistics(data)

# Export the statistics dataframe to an xarray dataset

stats = stats.to_xarray()

# Save intermediate results for further analysis and visualization

stats.to_netcdf('intermediate_results/Statistics_MAB_CruiseData.nc')
```

6

For analyses that include randomness, note the underlying seed

- Random number algorithms are often used in computational research, especially in statistical analyses.
- Every time random generator is run, a different random number is generated.
- To ensure exact replication of results, use the pseudo-random generator's seeding system.



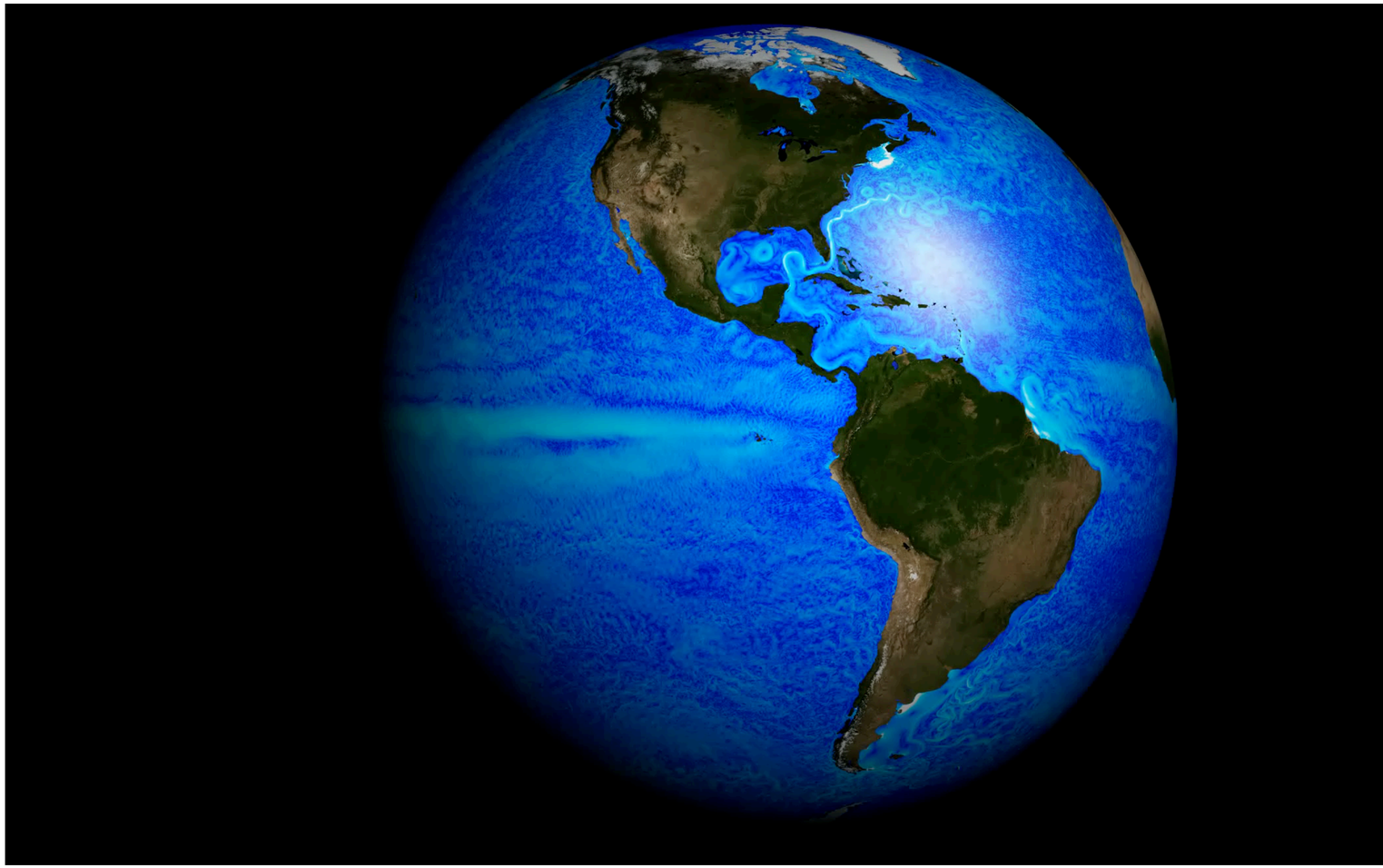
https://en.wikipedia.org/wiki/Random_number_generation

7

Always store raw data behind plots

- Never delete original data files.
- Exception: massive model outputs could be deleted (if really needed), but all files necessary to re-run the model must be stored; storing smaller subsets may be useful.

This simulation generated 10s of Petabytes of data, which was hard to store.



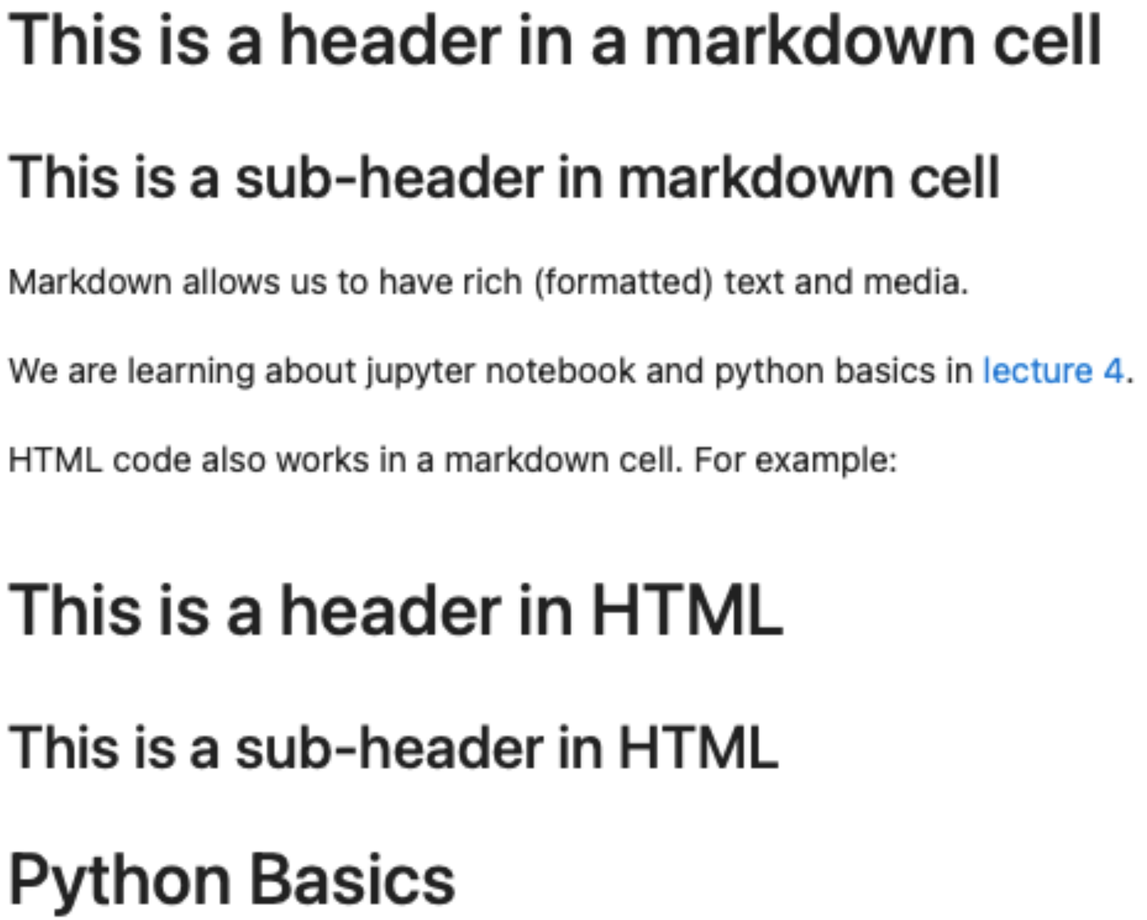
8

Comment your code: connect textual statements to code and results

- Jupyter notebooks are excellent for blending code with comments and figures (and it supports several languages besides Python).

- In simple scripts, use simple comments: short and punchy statements about each small block of code, e.g.: # loop over datasets.

Screenshot from notebook used in lecture 4

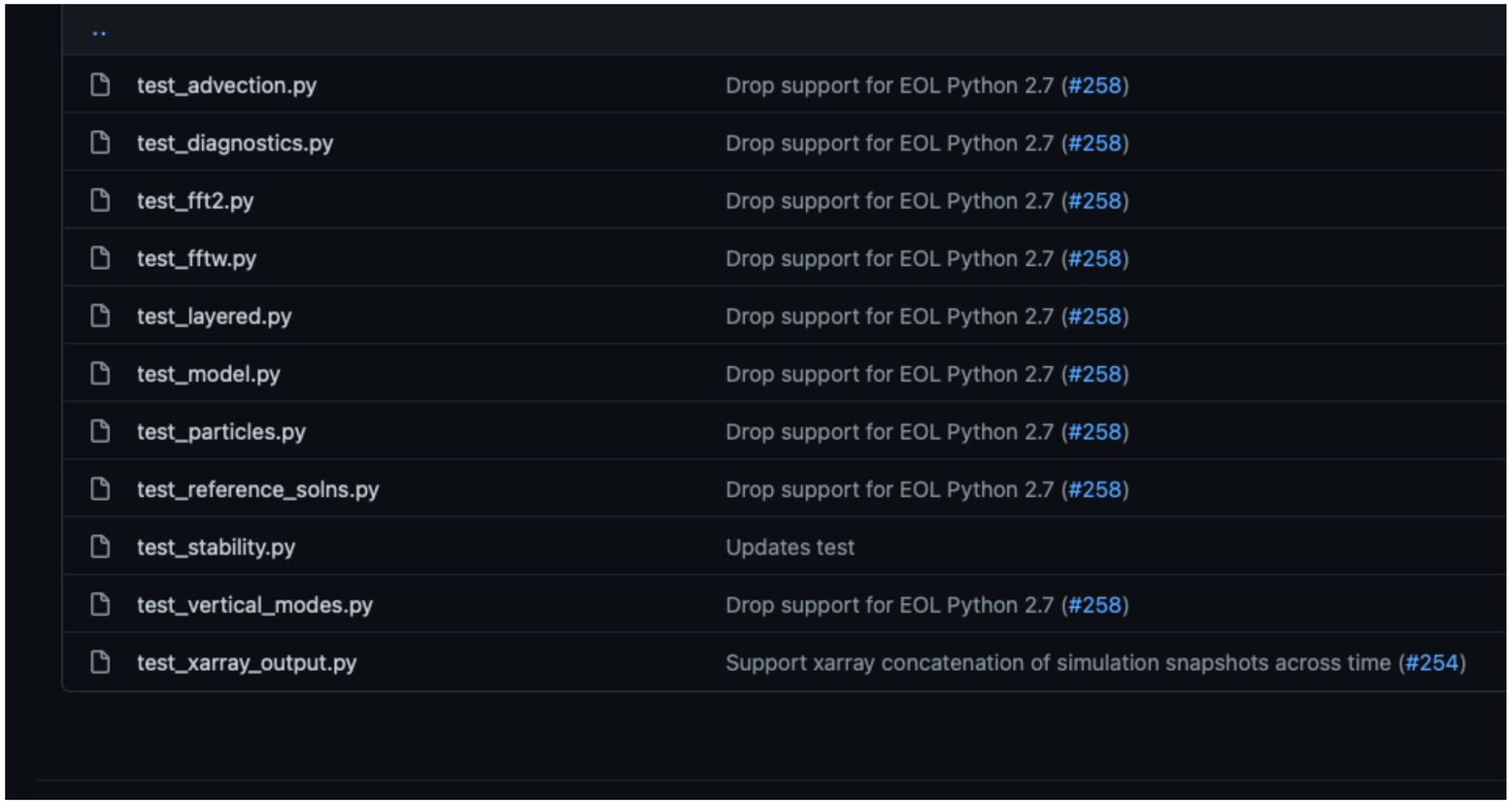


9

Test your code

- Come up with tests that ensure your code yields the right result in simple situations you know the answer.
- Write tests for every relevant block of code or analysis script.
- Re-run the tests whenever you make relevant changes to code.

Tests developed for pyqg project



10

Provide public access to scripts, runs, and results

- Make your Github repositories public when you submit a paper (or before if you are comfortable).
- To get credit for your code, use Zenodo to generate a DOI.
- Choose an appropriate license (MIT License, Creative Commons BY 4.0, etc.)
- Add License and Zenodo “badges” to GitHub repository.

Ten simple rules for computational research

Adapted from Sandve et al., 2013

1. For every result, keep track of how it was produced.
2. Avoid manual data manipulation steps.
3. Archive the exact versions of all external programs used.
4. Version-control all custom scripts.
5. Record all intermediate results, when possible in standard formats.
6. For analyses that include randomness, note underlying random seeds.
7. Always store raw data behind plots.
8. Comment your code
9. Test your code.
10. Provide public access to scripts, runs, and results.

Sandve, G. K. et al. (2013), Ten simple rules for reproducible computational research, PLOS Comp. Bio. (editorial), Vol. 9(10):1–4, doi: 10.1371/journal.pcbi.1003285