Best coding practices

Chenying Zhao

PennLINC lab meeting

11/28/2023

Acknowledgement

- Taylor Salo (a lot of feedbacks from him!)
- Matt Cieslak
- Tinashe Tapera
- Audrey Luo

Why we discuss coding practices?

- To enhance reproducibility
  - replicable by ourselves or others
- To enhance reusability
  - to make sure our code is readable and accessible to ourselves after several months, or by others, and even be reused in other projects
- To make our code less prone to mistakes
  - to boost our confidence in the correctness and accuracy of our code and results

Format of this meeting

- I list actions that we think should be encouraged
- Please discuss and/or raise different opinions!

Three stages of coding

Before you start coding for a project

During coding

After coding a bit

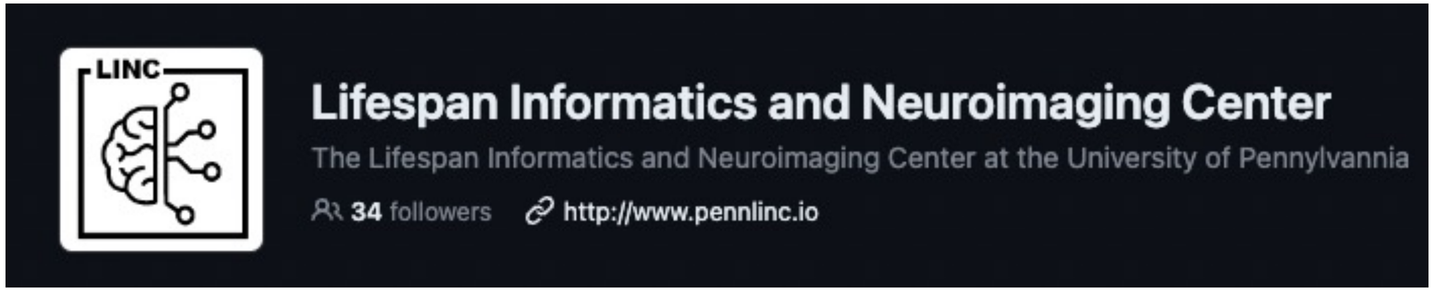
Three stages of coding

Before you start coding for a project

During coding

After coding a bit

Stage #1. Before starting to code for a project

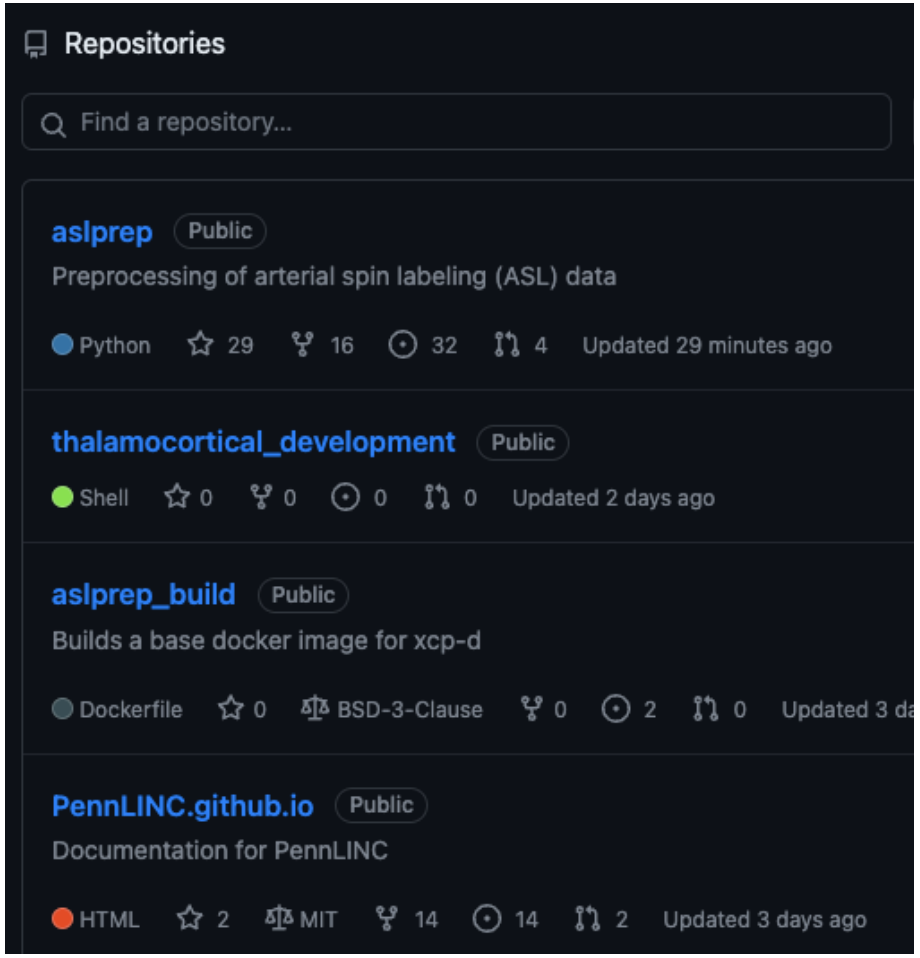


Lifespan Informatics and Neuroimaging Center

The Lifespan Informatics and Neuroimaging Center at the University of Pennylvannia

34 followers http://www.pennlinc.io

- Create a GitHub repository under "PennLINC GitHub group"
  - Why doing this at the very beginning, instead of the end of a project?
    - To start the version control from the very beginning
      - You'll appreciate the version control when you need to look back at changes you made to your code throughout the project's lifespan.
    - Also, it's easier to share and show your code to others, for feedbacks/questions/discussions
    - Finally, you may code & test locally, then move it onto clusters to apply the code to large datasets. This makes testing easier!



Repositories

Find a repository...

aslprep Public

Preprocessing of arterial spin labeling (ASL) data

Python 29 16 32 4 Updated 29 minutes ago

thalamocortical_development Public

Shell 0 0 0 0 Updated 2 days ago

aslprep_build Public

Builds a base docker image for xcp-d

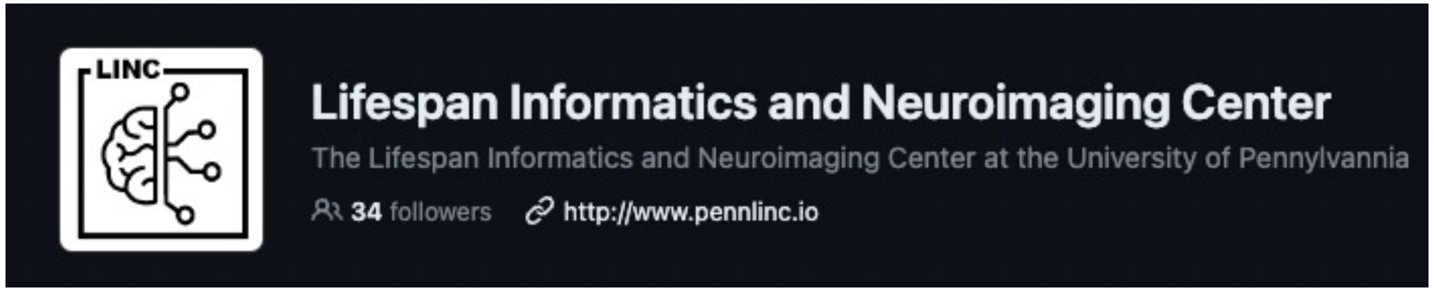
Dockerfile 0 BSD-3-Clause 0 2 0 Updated 3 da

PennLINC.github.io Public

Documentation for PennLINC

HTML 2 MIT 14 14 2 Updated 3 days ago

Stage #1. Before starting to code for a project

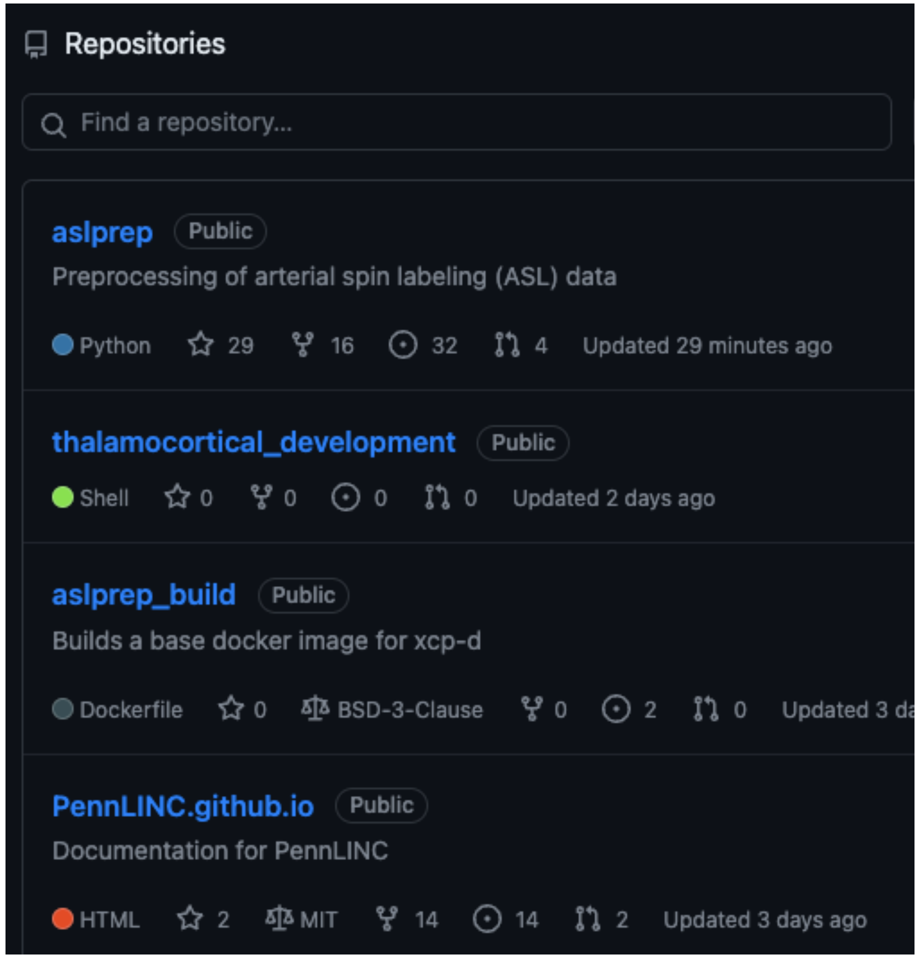


Lifespan Informatics and Neuroimaging Center

The Lifespan Informatics and Neuroimaging Center at the University of Pennylvannia

34 followers http://www.pennlinc.io

- Create a GitHub repository under “PennLINC GitHub group”
  - Hmmm, I’m hesitated to show my code publicly from the very beginning…
    - That's okay - you can always choose to make the GitHub repository "Private”, then switch to “Public” later



Repositories

Find a repository...

aslprep Public

Preprocessing of arterial spin labeling (ASL) data

Python 29 16 32 4 Updated 29 minutes ago

thalamocortical_development Public

Shell 0 0 0 0 Updated 2 days ago

aslprep_build Public

Builds a base docker image for xcp-d

Dockerfile 0 BSD-3-Clause 0 2 0 Updated 3 da

PennLINC.github.io Public

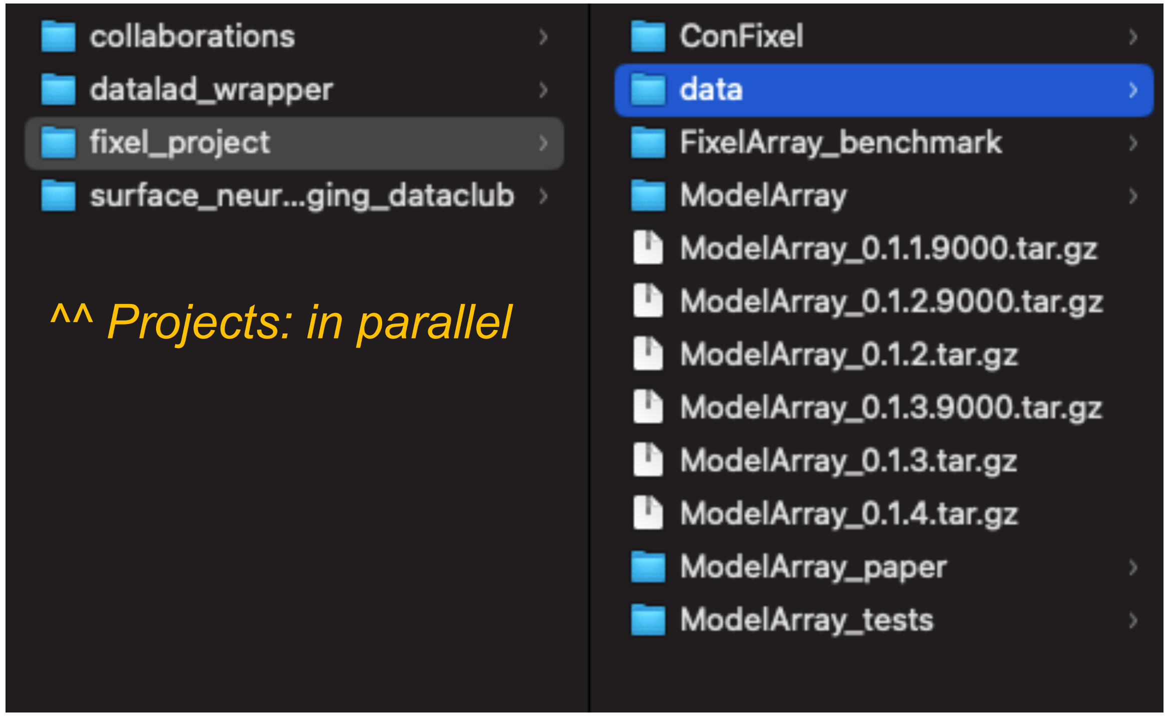
Documentation for PennLINC

HTML 2 MIT 14 14 2 Updated 3 days ago

Stage #1. Before starting to code for a project
Think about organization of data & code

- Projects: in parallel
  - So that you can reuse scripts or functions from other projects by sourcing them (via relative paths) ☺
- Separate code from data!
- Code: tracked by git, shared/backup on GitHub
- Data: tracked by DataLad, shared on OSF etc platforms

Example: Chenying's projects:



collaborations
datalad_wrapper
fixel_project
surface_neur...ging_dataclub

ConFixel
data
FixelArray_benchmark
ModelArray
ModelArray_0.1.1.9000.tar.gz
ModelArray_0.1.2.9000.tar.gz
ModelArray_0.1.2.tar.gz
ModelArray_0.1.3.9000.tar.gz
ModelArray_0.1.3.tar.gz
ModelArray_0.1.4.tar.gz
ModelArray_paper
ModelArray_tests

^^ Projects: in parallel

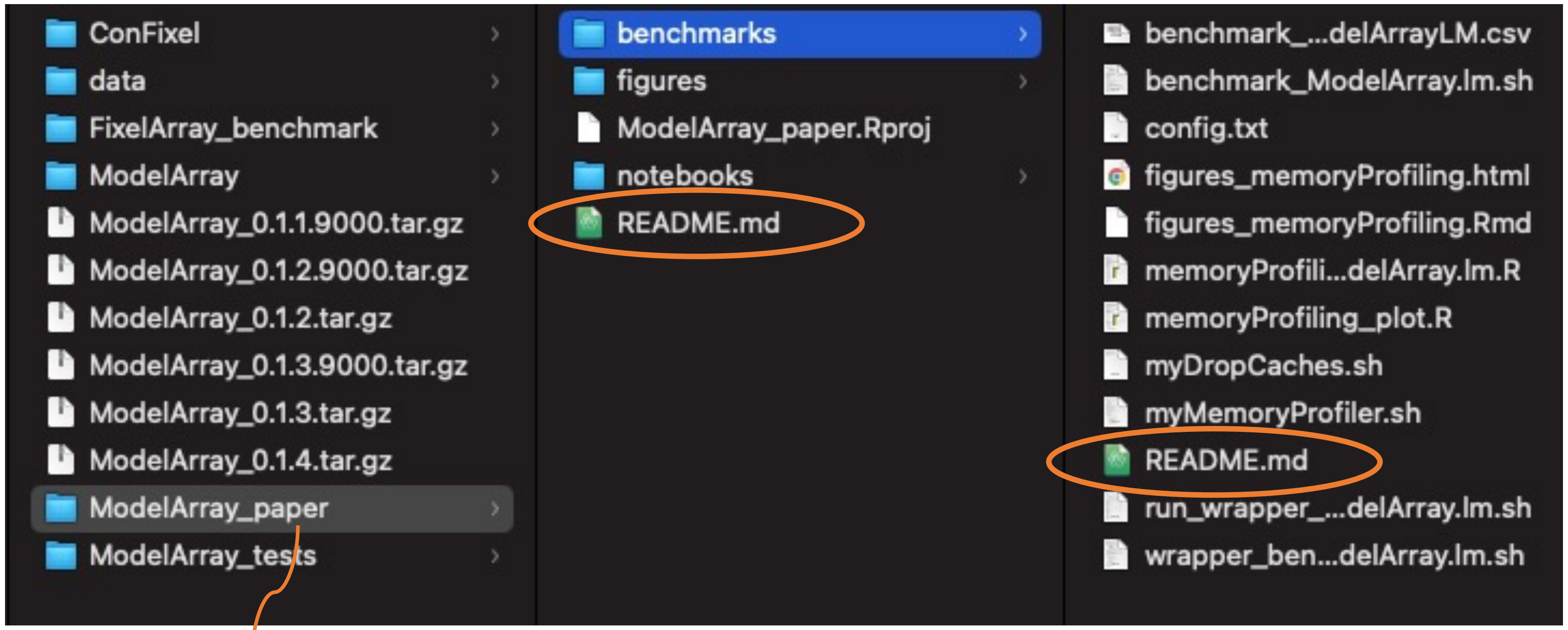
- *For ModelArray project:*
  - *Folders except "data" folder are mostly code repositories*
  - *"data" folder includes data for testing ModelArray*

Stage #1. Before starting to code for a project

Think about organization of data & code

- How to organize code folders?

Different folders for different purposes/functions/steps



ConFixel
data
FixelArray_benchmark
ModelArray
ModelArray_0.1.1.9000.tar.gz
ModelArray_0.1.2.9000.tar.gz
ModelArray_0.1.2.tar.gz
ModelArray_0.1.3.9000.tar.gz
ModelArray_0.1.3.tar.gz
ModelArray_0.1.4.tar.gz
ModelArray_paper
ModelArray_tests
benchmarks
figures
ModelArray_paper.Rproj
notebooks
README.md
benchmark_...delArrayLM.csv
benchmark_ModelArray.lm.sh
config.txt
figures_memoryProfiling.html
figures_memoryProfiling.Rmd
memoryProfili...delArray.lm.R
memoryProfiling_plot.R
myDropCaches.sh
myMemoryProfiler.sh
README.md
run_wrapper_...delArray.lm.sh
wrapper_ben...delArray.lm.sh

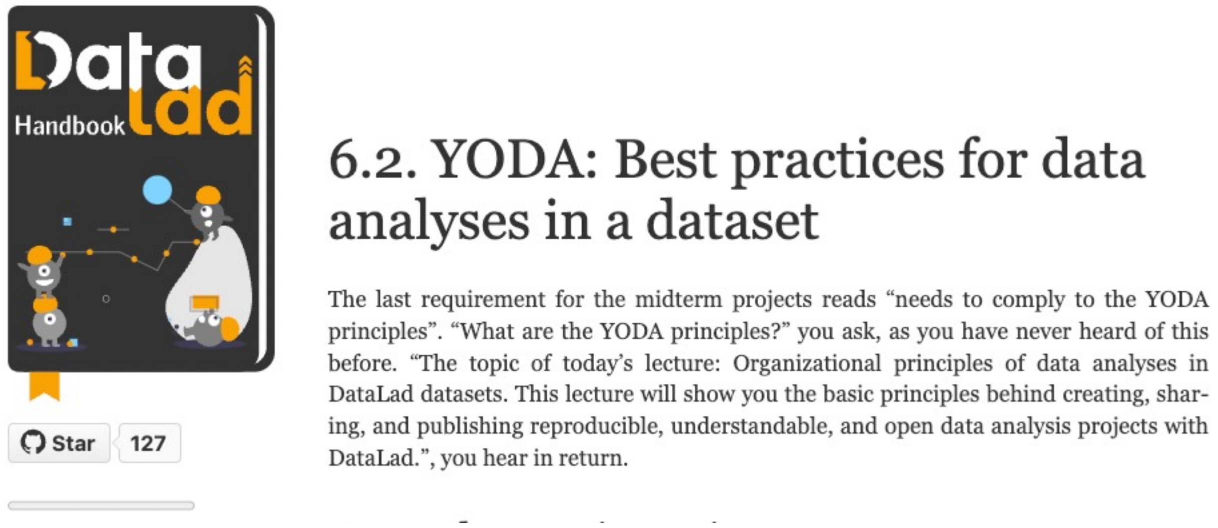
The code repo for ModelArray paper

Add README files to explain (or using a separate doc page): overview of scripts, order to run, etc

Stage #1. Before starting to code for a project
Think about organization of data & code

- How to organize data folders?
  - Separate data derivatives from raw data

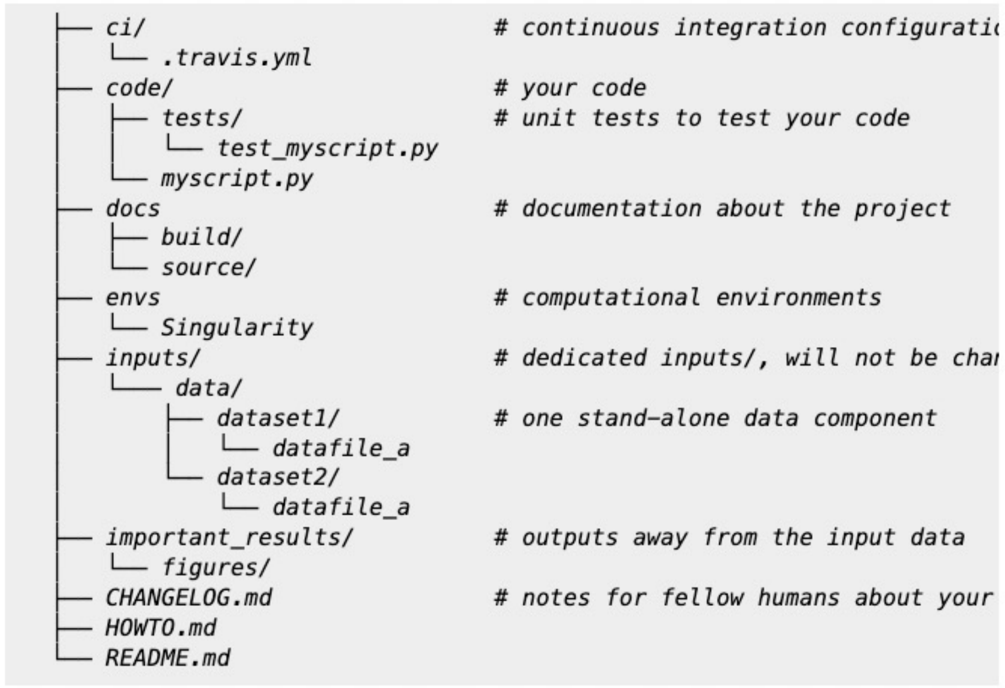
For more?



6.2. YODA: Best practices for data analyses in a dataset

The last requirement for the midterm projects reads "needs to comply to the YODA principles". "What are the YODA principles?" you ask, as you have never heard of this before. "The topic of today's lecture: Organizational principles of data analyses in DataLad datasets. This lecture will show you the basic principles behind creating, sharing, and publishing reproducible, understandable, and open data analysis projects with DataLad.", you hear in return.

https://handbook.datalad.org/en/latest/basics/101-127-yoda.html

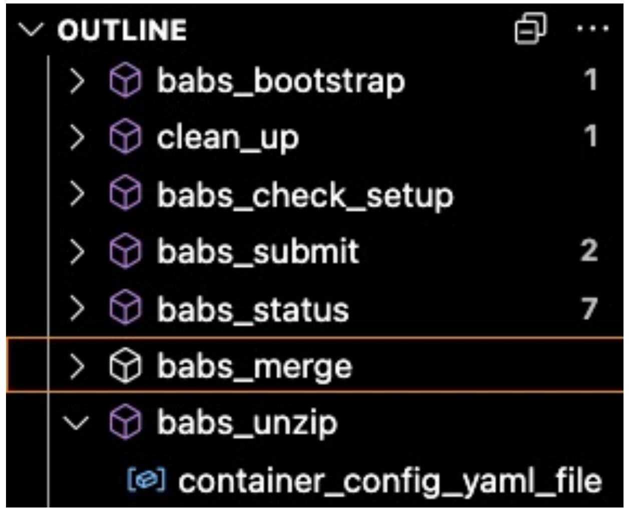


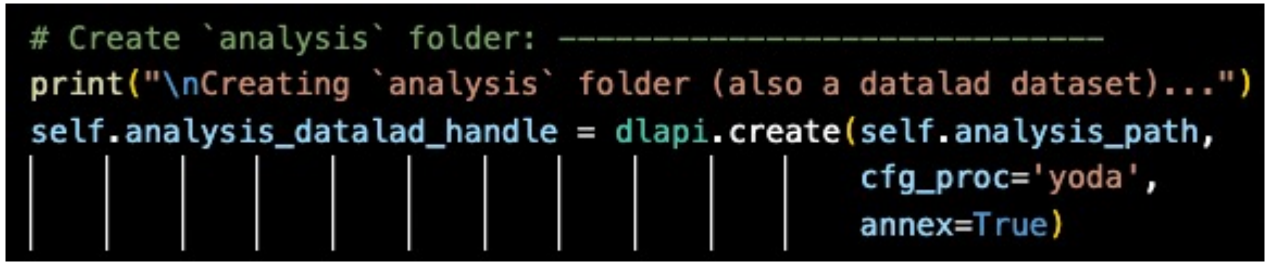
```
├── ci/                          # continuous integration configurati
│   └── .travis.yml
├── code/                        # your code
│   ├── tests/                   # unit tests to test your code
│   │   └── test_myscript.py
│   └── myscript.py
├── docs                         # documentation about the project
│   ├── build/
│   └── source/
├── envs                         # computational environments
│   └── Singularity
├── inputs/                      # dedicated inputs/, will not be char
│   └── data/
│       ├── dataset1/            # one stand-alone data component
│       │   └── datafile_a
│       └── dataset2/
│           └── datafile_a
├── important_results/           # outputs away from the input data
│   └── figures/
├── CHANGELOG.md                 # notes for fellow humans about your
├── HOWTO.md
└── README.md
```

Stage #1. Before starting to code for a project
Tools for coding

- Coding UI:
  - VS Code for Python, bash, etc
  - R Studio for R
  - …
- Autoformatting and linting tools:
  - Why we need them? Make code more readable!
  - For Python, for example: (recommended by Taylor Salo):
    - autoformatting tools: \`black\` and \`isort\`
    - linting tools: \`flake8\`
  - For R: TODO: need to google to find good R linting tools in R studio

VS Code outline of \`babs.py\`:





```
# Create `analysis` folder: ----------------------------
print("\nCreating `analysis` folder (also a datalad dataset)...")
self.analysis_datalad_handle = dlapi.create(self.analysis_path,
                                            cfg_proc='yoda',
                                            annex=True)
```

- Each line: Not too long
- Aligned
- …

Three stages of coding

Before you start coding for a project

During coding

After coding a bit

Step 2. Coding itself
Right before coding

- Think about the purpose of the code you want to achieve
- Which language will be efficient to achieve your goal here?
- You can have more than one language of code in one project!
    - Python is good for image processing, data wrangling, and figure generation;
    - R is good for data wrangling, statistical analysis, and figure generation;
    - Bash scripts are best for submitting jobs on clusters;
    - Some people may prefer Matlab as well

Incorporated suggestions from Taylor Salo

Step 2. Coding itself
Right before coding

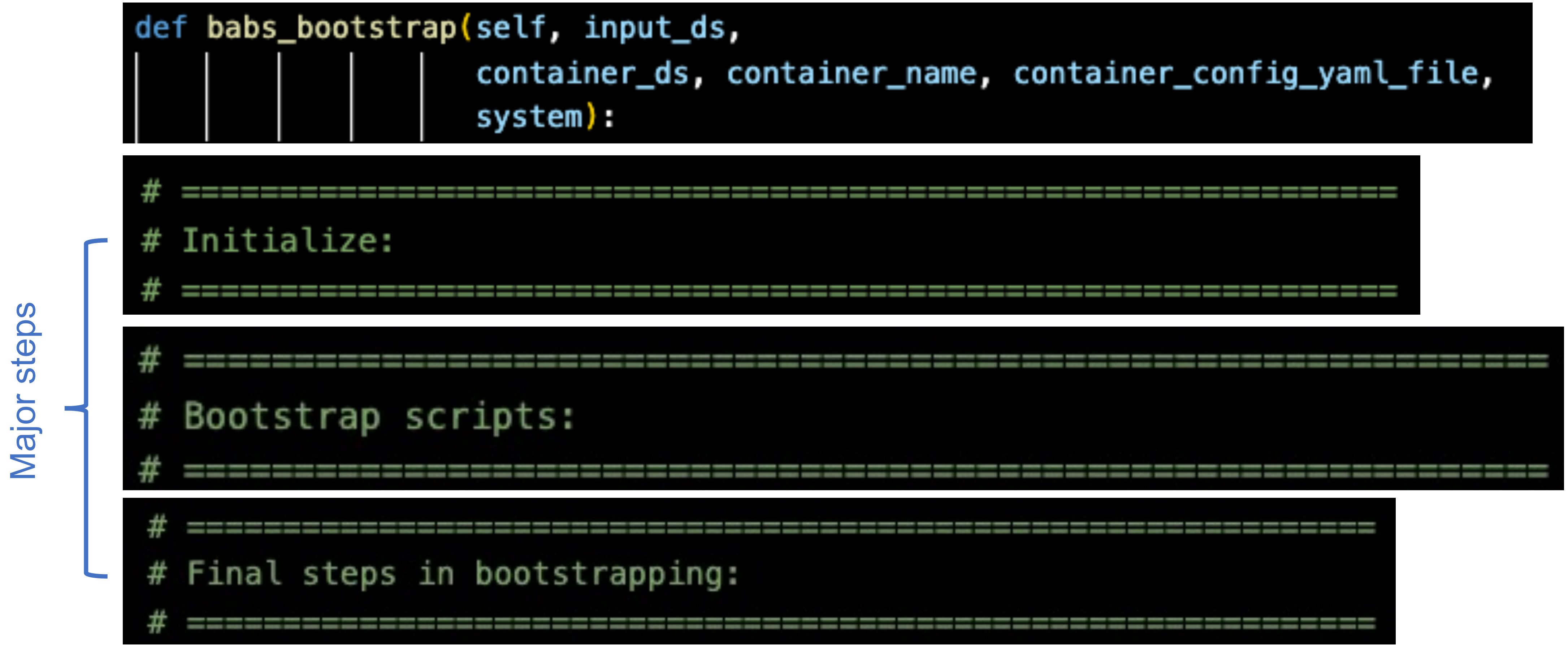
- If the code involves a few steps, you might get lost when coding
- Instead, break down the coding into a few steps
    - Focus on one step each time
    - Consider writing down the list of the steps as comments before you start coding it.

Step 2. Coding itself

Right before coding

- Break down the coding into a few steps

The method `babs_bootstrap()` is a long one, and includes a lot of steps:



```
def babs_bootstrap(self, input_ds,
                   container_ds, container_name, container_config_yaml_file,
                   system):
```

```
# ========================================================
# Initialize:
# ========================================================
```

```
# ========================================================
# Bootstrap scripts:
# ========================================================
```

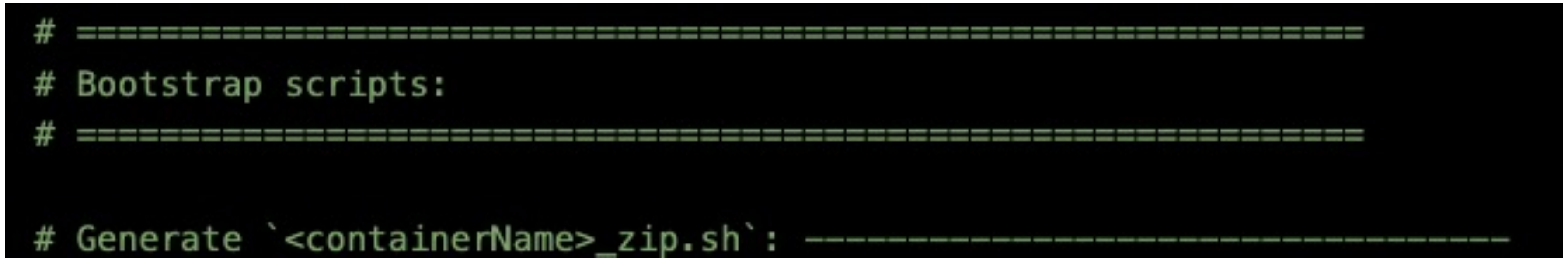
```
# ========================================================
# Final steps in bootstrapping:
# ========================================================
```

Step 2. Coding itself

Right before coding

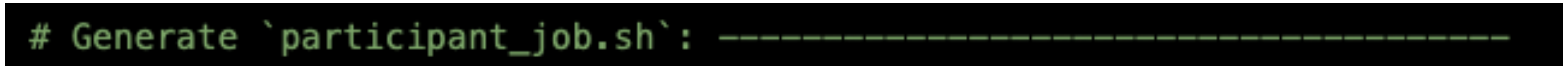
- Break down the coding into a few steps
- If the code involves a few steps, you might get lost when coding/focusing on one step. Consider writing down the list of the steps as comments before you start coding it.

For each major step, there are a few smaller steps, too:

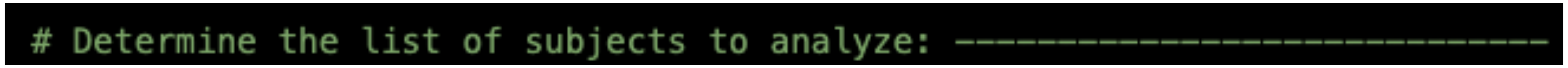


```
# =============================================================
# Bootstrap scripts:
# =============================================================

# Generate `<containerName>_zip.sh`: ----------------------------------
```



```
# Generate `participant_job.sh`: -----------------------------------
```



```
# Determine the list of subjects to analyze: ---------------------------
```

…

Then fill each step out

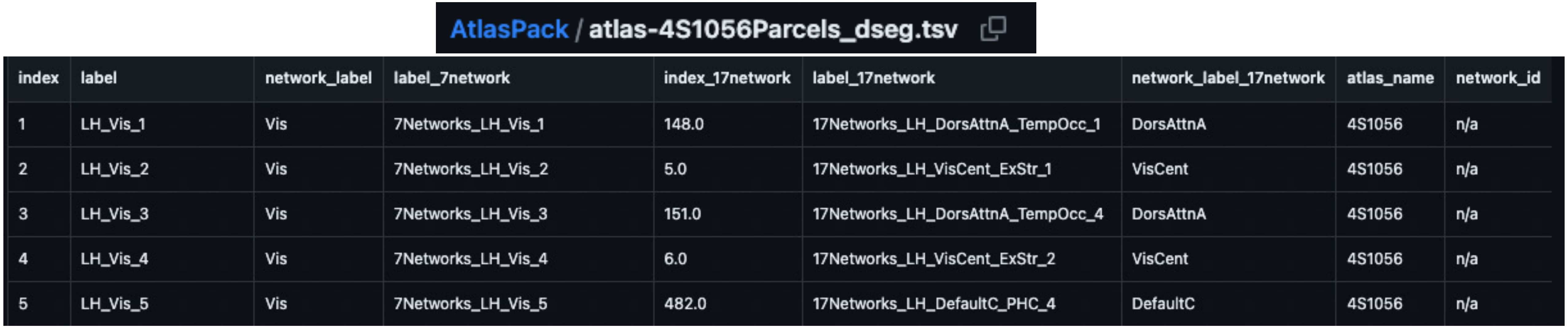
Step 2. Coding itself
Reduce duplicated code

- If you have duplicated code, it might be a good time to translate them into a function.
- Why? It will be easier for version control ☺
  - So that in the future, you only need to edit the function once.
  - Otherwise, if you want to edit one of the versions, you will also have to update the other one.

Step 2. Coding itself

Reduce duplicated list or table as input

- Sometimes you hope to define a list of brain regions in your analysis, and you'll use it repeatedly in other scripts too.
    - For lists: may define the list in a text file.
    - For tables (e.g., full names, abbreviations, color schemes, etc of brain regions): you can define it in a TSV file
- Then save the text file or TSV file in the Git repo if the file size is not big
- When you want to use this, just `source` the text file or directly load the TSV file.



AtlasPack / atlas-4S1056Parcels_dseg.tsv

index	label	network_label	label_7network	index_17network	label_17network	network_label_17network	atlas_name	network_id
1	LH_Vis_1	Vis	7Networks_LH_Vis_1	148.0	17Networks_LH_DorsAttnA_TempOcc_1	DorsAttnA	4S1056	n/a
2	LH_Vis_2	Vis	7Networks_LH_Vis_2	5.0	17Networks_LH_VisCent_ExStr_1	VisCent	4S1056	n/a
3	LH_Vis_3	Vis	7Networks_LH_Vis_3	151.0	17Networks_LH_DorsAttnA_TempOcc_4	DorsAttnA	4S1056	n/a
4	LH_Vis_4	Vis	7Networks_LH_Vis_4	6.0	17Networks_LH_VisCent_ExStr_2	VisCent	4S1056	n/a
5	LH_Vis_5	Vis	7Networks_LH_Vis_5	482.0	17Networks_LH_DefaultC_PHC_4	DefaultC	4S1056	n/a

https://github.com/PennLINC/AtlasPack/blob/main/atlas-4S1056Parcels_dseg.tsv

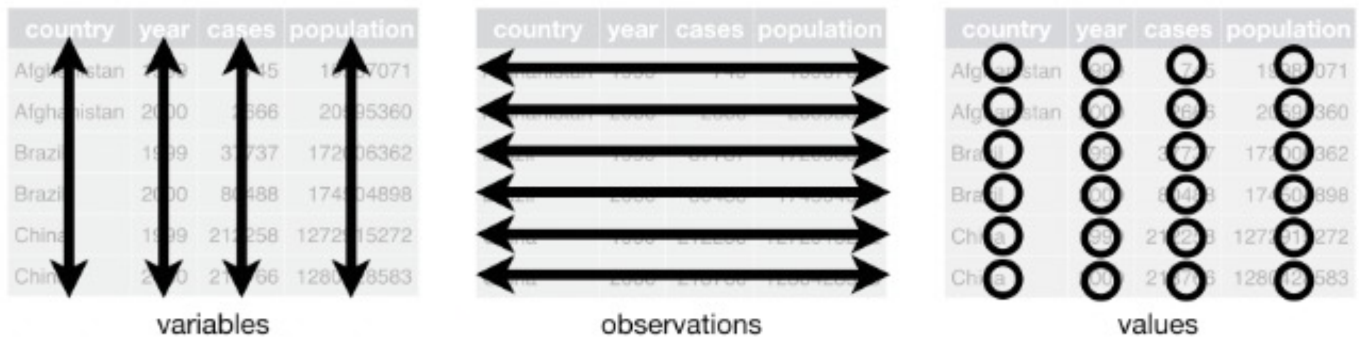
Step 2. Coding itself

Data wrangling

- The data analysis part after (pre)processing often involve data wrangling.
- Highly encourage to use "Tidy Data"

There are three interrelated rules which make a dataset tidy:

1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.



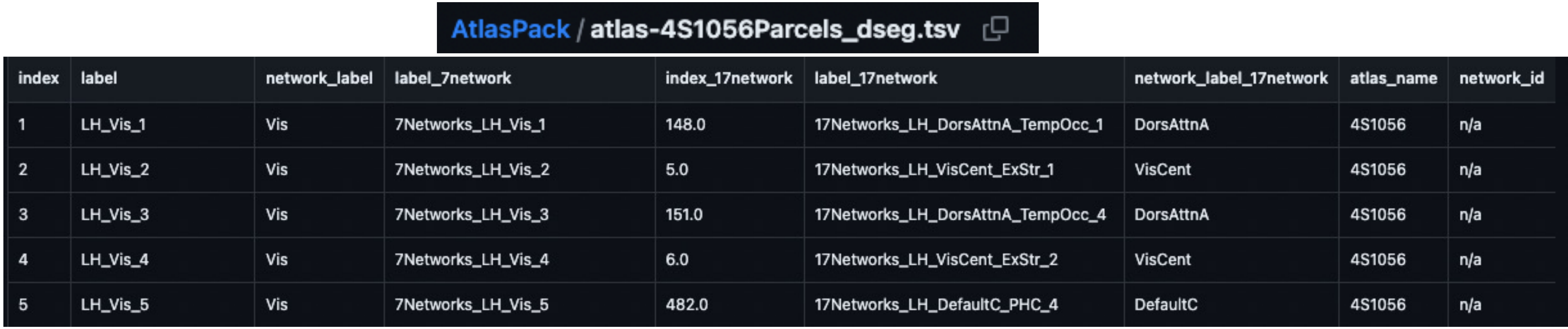
From book "R for Data Science": https://r4ds.had.co.nz/tidy-data.html
Hadley Wickham, Mine Çetinkaya-Rundel, and Garrett Grolemund

Step 2. Coding itself
Data wrangling

- Why use “Tidy data”? It’s very easy to use!
  - Easy to select / extract the data to use
  - Easy to check number of observations

Ref: book “R for Data Science”: https://r4ds.had.co.nz/tidy-data.html

This is also an example of tidy data:



AtlasPack / atlas-4S1056Parcels_dseg.tsv

index	label	network_label	label_7network	index_17network	label_17network	network_label_17network	atlas_name	network_id
1	LH_Vis_1	Vis	7Networks_LH_Vis_1	148.0	17Networks_LH_DorsAttnA_TempOcc_1	DorsAttnA	4S1056	n/a
2	LH_Vis_2	Vis	7Networks_LH_Vis_2	5.0	17Networks_LH_VisCent_ExStr_1	VisCent	4S1056	n/a
3	LH_Vis_3	Vis	7Networks_LH_Vis_3	151.0	17Networks_LH_DorsAttnA_TempOcc_4	DorsAttnA	4S1056	n/a
4	LH_Vis_4	Vis	7Networks_LH_Vis_4	6.0	17Networks_LH_VisCent_ExStr_2	VisCent	4S1056	n/a
5	LH_Vis_5	Vis	7Networks_LH_Vis_5	482.0	17Networks_LH_DefaultC_PHC_4	DefaultC	4S1056	n/a

https://github.com/PennLINC/AtlasPack/blob/main/atlas-4S1056Parcels_dseg.tsv

For subject/session data: just add columns of subjects, sessions, etc

Step 2. Coding itself
Data wrangling

Personally prefer R for data wrangling

- a lot of great R packages, like `dplyr`, `tidyr`, etc

Step 2. Coding itself

Even better: Follow the BIDS convention

For example:

- use tsvs instead of csvs
- write out jsons describing each of the columns in the tsv
- name the files using BIDS entities and suffixes

credit to Taylor Salo

Step 2. Coding itself
Reduce manual work

- What kinds of manual work?
  - Manually copying and pasting the statistical results into a table
  - Adding components in figures which can be replaced by code
- Why reduce manual work?
  - Manual work (e.g., manually counting or changing numbers) increases risk of mistakes and reduces the confidence of what you reported in the manuscript.
  - In addition, it will also increase the burden for the reproducibility buddy - they need to more carefully check it to make sure what you reported are consistent with the original output, especially if they are in a different format.

Step 2. Coding itself
Reduce manual work

- Generate final tables and figures directly from the code as much as possible.
- Avoid additional manual work (e.g., round up, copy-paste, combination of figure panels) to generate the final table or figure you'll use in the paper. Instead:
    - Consider using R to export pdf versions of tables.
    - Consider writing code to directly generate the figure you want, instead of manually adjusting the fonts in Illustrator or Powerpoint later.
    - Consider writing code to directly combine figure panels and save the figure - although this might not be always easy – see below *
    - Consider directly generating the final format of the values, e.g., let the code round up statistics, calculate the final number, or count for you

** Although in some cases, manual combination would be easier. See Audrey Luo's tutorial on "InDesign for Efficient and Beautiful Figure-making": https://pennlinc.github.io/docs/Tutorials/InDesign_Tutorial/*

Three stages of coding

Before you start coding for a project

During coding

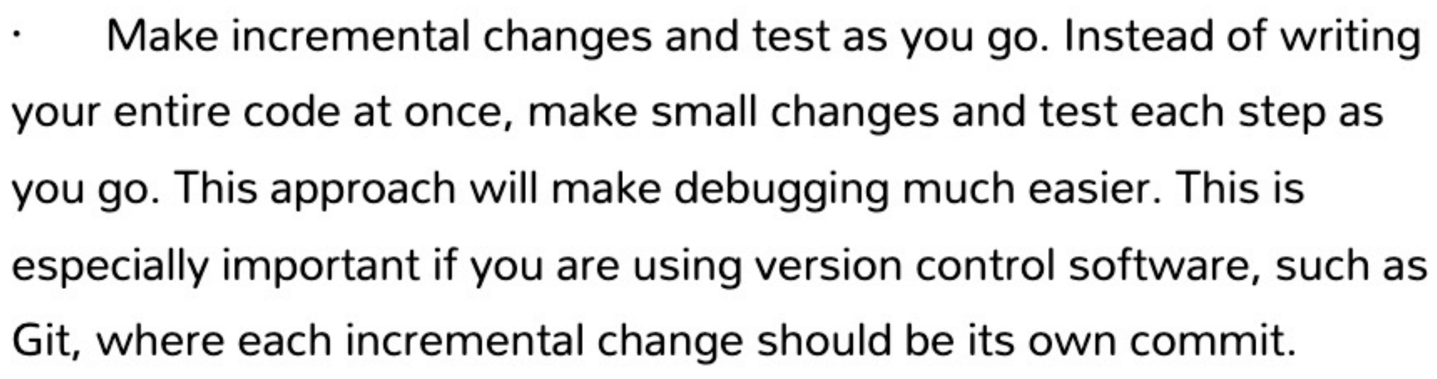
After coding a bit

Step 3. After you code a bit
Test your code

- It's important to test out if your code really works as you thought.
- Make some simple toy data, apply your code, and see if you can get what you want.
- Include sanity checks (in your code, or separately in CircleCI tests)
    - Sanity checks are very helpful to rule out mistakes.
    - Are the shape (number of rows and columns) of the data frame you generated is consistent with your expectation?
    - Use \`testthat\` from R (e.g., \`expect_that()\`), and \`assert\` from Python to assert that what you actually get is the same as what you expect.

Step 3. After you code a bit

“Make incremental changes and test as you go.”



- Make incremental changes and test as you go. Instead of writing your entire code at once, make small changes and test each step as you go. This approach will make debugging much easier. This is especially important if you are using version control software, such as Git, where each incremental change should be its own commit.

From: https://www.ohbmtrainees.com/blog-overview/2023/2/10/coding-best-practices-for-academia-bridging-the-gap-between-research-and-industry

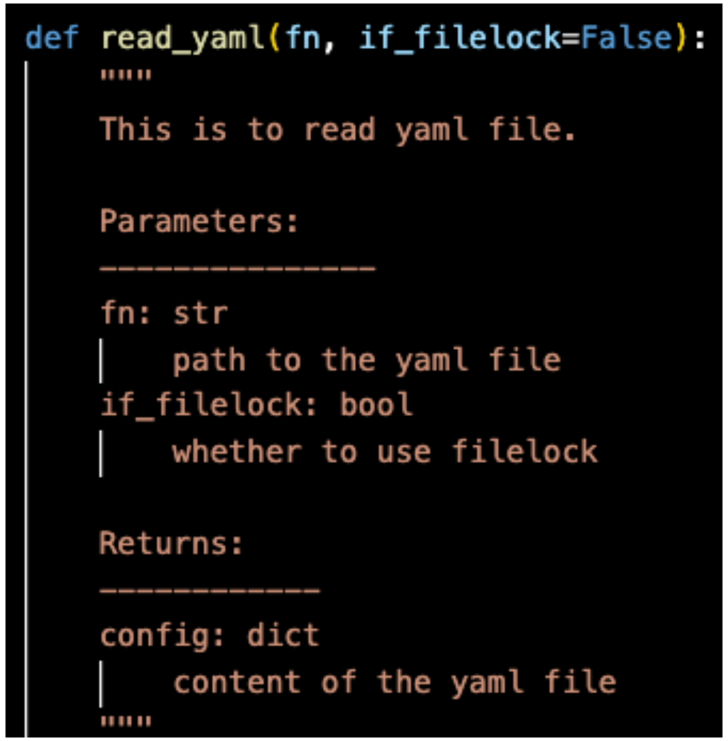
Step 3. After you code a bit
Document your code and make it readable

- Sometimes we're too focused on writing runnable code and we forget to write notes for explanation.
- You'll quickly forget the definition of arguments and the purpose of a function.
- Document your code before you move on to another task.
- You can also write docs before you write the function and use it to guide your coding.

Step 3. After you code a bit
Document your code and make it readable

- We strongly recommend doing the following:
  - Add comments throughout the code to make sure readers can understand what's happening.
  - Add docstrings to any functions and scripts you write.

Python function (from BABS):

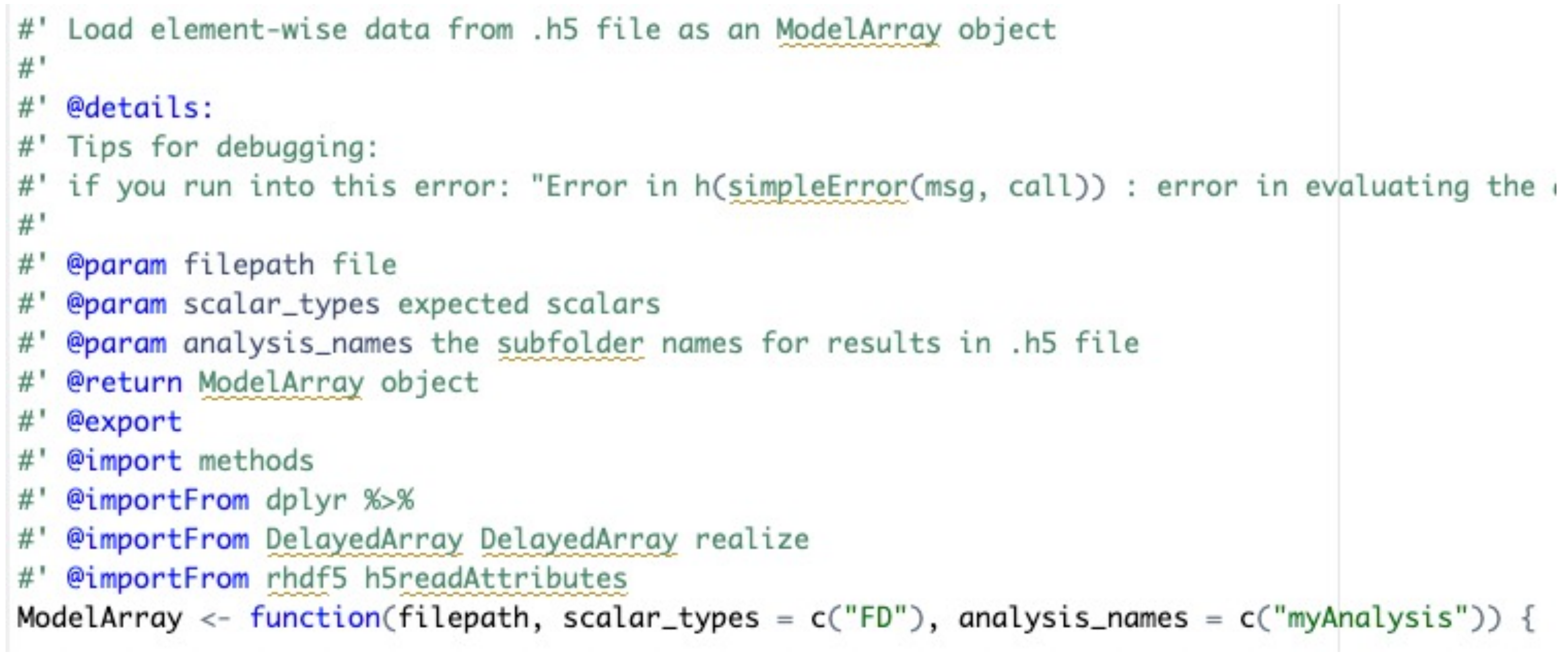


```python
def read_yaml(fn, if_filelock=False):
    """
    This is to read yaml file.

    Parameters:
    ---------------
    fn: str
        path to the yaml file
    if_filelock: bool
        whether to use filelock

    Returns:
    ------------
    config: dict
        content of the yaml file
    """
```

R function (from ModelArray):



```r
#' Load element-wise data from .h5 file as an ModelArray object
#'
#' @details:
#' Tips for debugging:
#' if you run into this error: "Error in h(simpleError(msg, call)) : error in evaluating the
#'
#' @param filepath file
#' @param scalar_types expected scalars
#' @param analysis_names the subfolder names for results in .h5 file
#' @return ModelArray object
#' @export
#' @import methods
#' @importFrom dplyr %>%
#' @importFrom DelayedArray DelayedArray realize
#' @importFrom rhdf5 h5readAttributes
ModelArray <- function(filepath, scalar_types = c("FD"), analysis_names = c("myAnalysis")) {
```

Incorporated suggestions from Taylor Salo

Step 3. After you code a bit
Document your code and make it readable

- We strongly recommend doing the following (cont’d):
  - Use informative variable names.
    - Single-letter variables (e.g., `i`) are generally not helpful for readers. Using interpretable variables will make it easier for readers (and future you) to make sense of the code.

Incorporated suggestions from Taylor Salo

There are more things we haven’t covered

- Informative way to name a script
- How to handle old and temporary data files
- …

Final words – need more help?

Need more help? Coding in an inefficient way?

- Check out answers on Neurostars, Stack Overflow, etc
- Ask in slack channels: #informatics, #r, etc

Thank you !!!