Genomics workflow management with Snakemake

Arjun Biddanda
July 11th, 2024

Key points to using snakemake and workflow management

1. “Failing fast” in academic research advising

2. Improving your reproducibility to safeguard your science*

3. Become a better “provider” of data to others

4. *Bonus:* improved power-user status in the Unix shell

The most common scenario in research advising

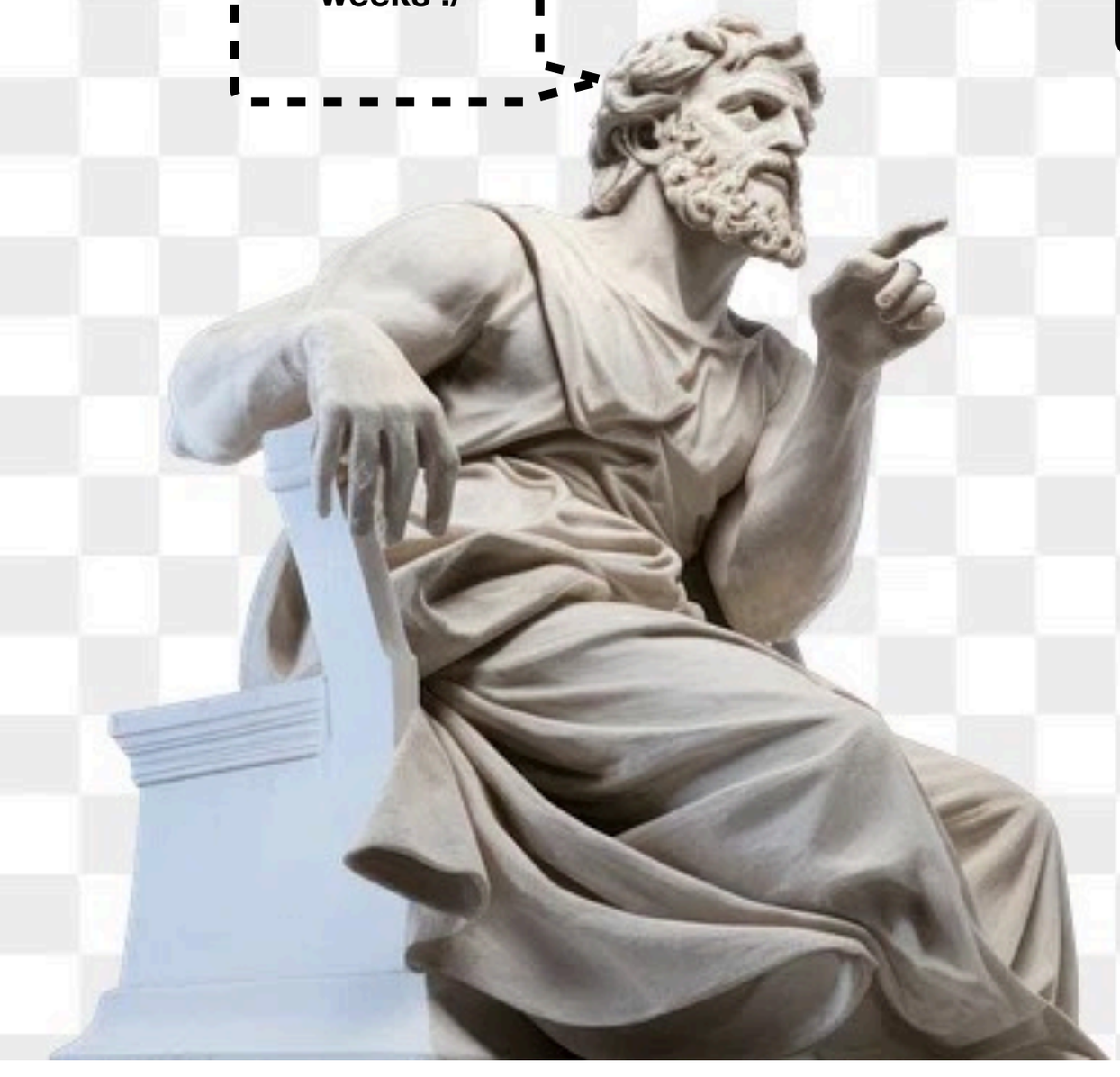
Optimistic Trainee

Pessimistic/Realistic Advisor

The most common scenario in research advising

Optimistic Trainee

There goes my next two weeks :/



Have you thought about controlling for *Z1*, *Z2*, *Z3*, … ?

Pessimistic/Realistic Advisor

Overall workflow design for projects

Where your parameter combinations are configured

The overall set of programs you need to run rules

Where the actual “workflow” or pipeline resides

The “main” file that incorporates all of your rules (sub-workflows can be in the “rules” directory)

Rule-specific environments (e.g. only one rule requires `msprime` for simulation)

Rule definitions for our simulation workflow

Helper-scripts to run aspects of the workflow

```
aabiddanda:λ> tree
.
├── README.md
├── config
│   └── config.yaml
├── environment.yaml
├── presentation
└── workflow
    ├── Snakefile
    ├── envs
    │   └── msprime.yaml
    ├── rules
    │   └── sims.smk
    └── scripts
        ├── sim_msprime.py
        └── summary_stats.py
```

Anatomy of a snakemake rule

```
rule create_summary_stats:
    """
    Create some population-genetic summary statistics
    and store in a temporary TSV file.
    """
    input:
        tsz="results/sim_data/{n}.{length}.{seed}.trees.tsz",
    output:
        tsv=temp("results/sim_tsv/{n}.{length}.{seed}.sumstats.tsv"),
    script:
        "../scripts/summary_stats.py"
```

At its simplest, a Snakemake rule needs to:

1. Know what to expect as input

2. Know what to expect as its output

3. Know what operation/algorithm to run to go from input -> output

Exploring the key features of snakemake

1. Naive process-based parallelization
2. Running on a slurm-based cluster with profiles
3. Creating rule-based “benchmarks” to understand timing and predicting compute usage
4. Managing resources & temporary files

Further resources

- https://vincebuffalo.com/blog/2020/03/04/understanding-snakemake.html

- https://snakemake.readthedocs.io/en/stable/tutorial/tutorial.html#tutorial

- https://github.com/jdblischak/smk-simple-slurm