Snakemake
Build reproducible data analysis workflows

Teach + Learn Session MannLabs 2025

What is snakemake?

*"The Snakemake workflow management system is a tool to create **reproducible and scalable** data analyses.*
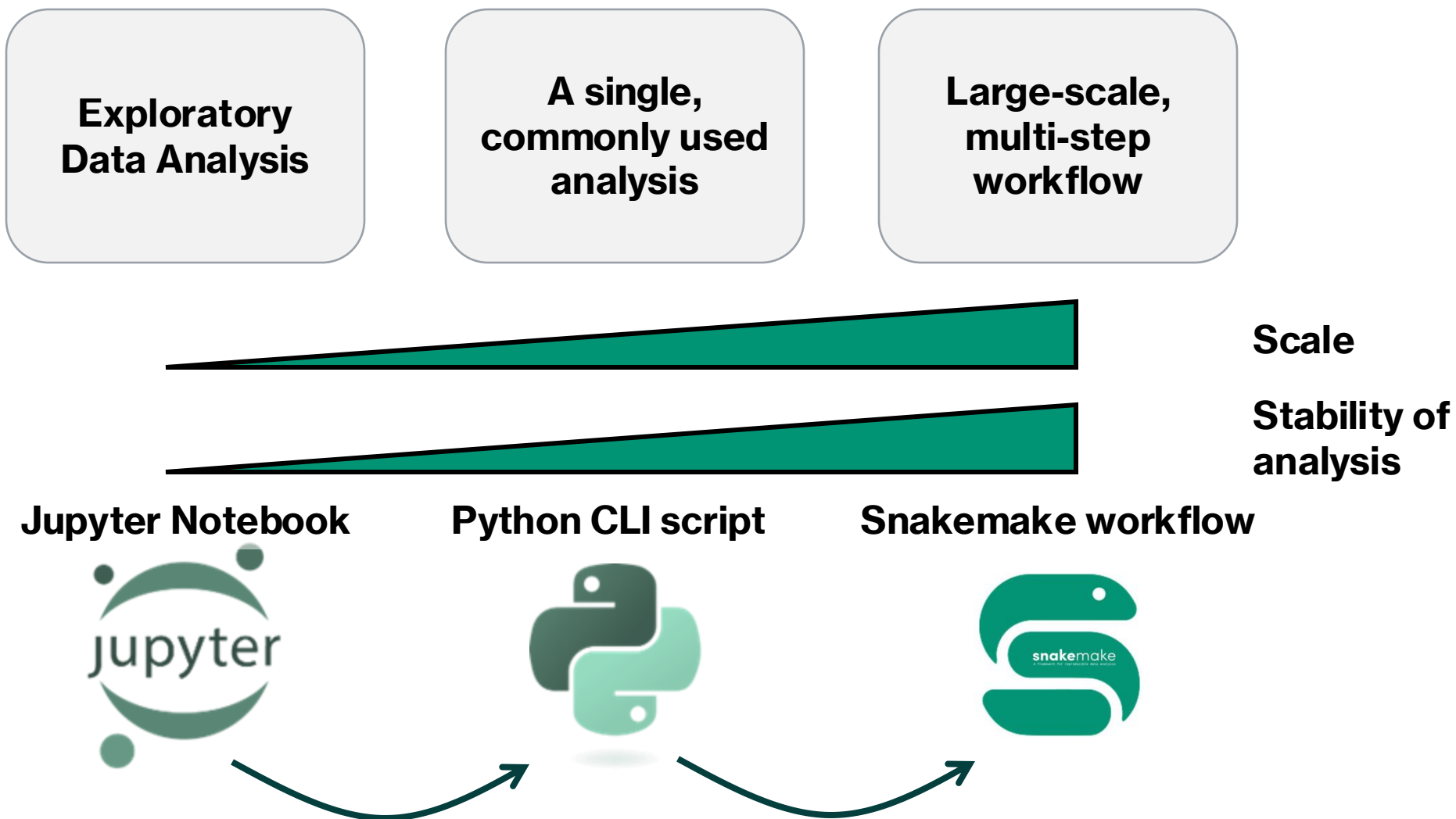
Describes + formalizes data analysis workflows in a python-based language

Improves reproducibility by describing the software environment + enforcing runnability

Manages the workflow in various environments (local, HPC, cloud), enabling scalability

Reports results in interactive reports

When should I use snakemake?

| Exploratory Data Analysis | A single, commonly used analysis | Large-scale, multi-step workflow |

Scale

Stability of analysis

Jupyter Notebook   **Python CLI script**   **Snakemake workflow**

A snakemake project

```
├── README.md
├── LICENSE.md
├── .gitignore
├── workflow
│   ├── rules
│   │   ├── module1.smk
│   │   └── module2.smk
│   ├── envs
│   │   ├── tool1.yaml
│   │   └── tool2.yaml
│   ├── scripts
│   │   ├── script1.py
│   │   └── script2.R
│   ├── notebooks
│   │   ├── notebook1.py.ipynb
│   │   └── notebook2.r.ipynb
│   ├── report
│   │   ├── plot1.rst
│   │   └── plot2.rst
│   └── Snakefile
├── config
│   ├── config.yaml
│   └── some-sheet.tsv
├── results
└── resources
```

Snakefile

Provides the entrypoint to the workflow for snakemake

- **rules**

  Describe + formalize the order in which tasks are performed

- **scripts**

  Contain the computing logic

- **configs**

  Add configuration options to the workflow

- **envs**

  Describe the computing environment

(Hands on) Snakemake rules
Formalizing a workflow

```
rule transform_image:
    input:
        "../results/astronaut.png",
    output:
        "../results/transformed-images/{transformation}.png",
    log:
        "../transform_image/{transformation}.log",
    conda:
        "./envs/environment.yaml"
    shell:"""
        python scripts/transform.py \
            --image {input} \
            --transformation {wildcards.transformation} \
            --output {output}
        """
```

During a run, snakemake tests whether the output of the rule exists for a specific parametrization and if not, reruns the command

Input

Input file for this step

Output

Output file generated by this step

Log

Log file of step

Conda

Conda environment used by this step

shell

Shell command (see also run directive for scripts)

(Hands on) Snakemake wildcards
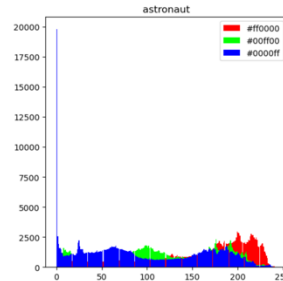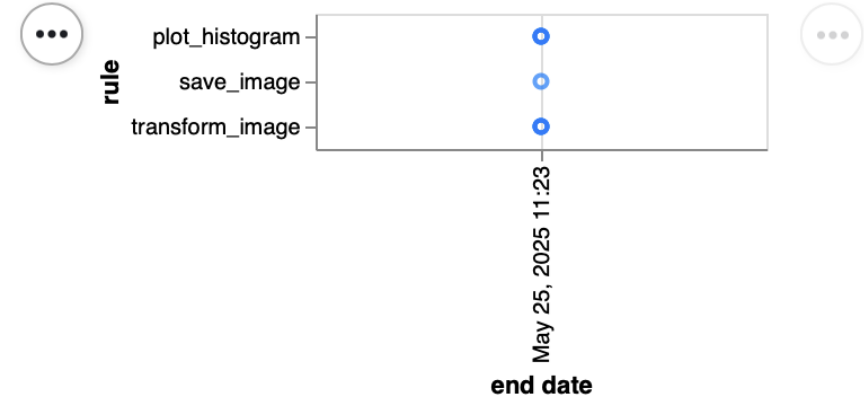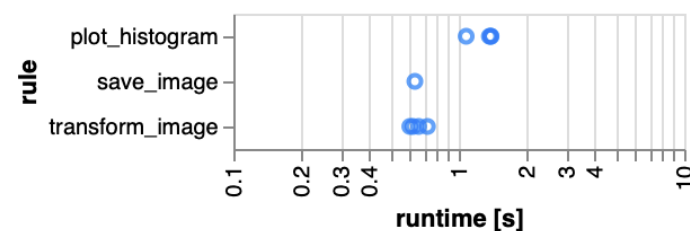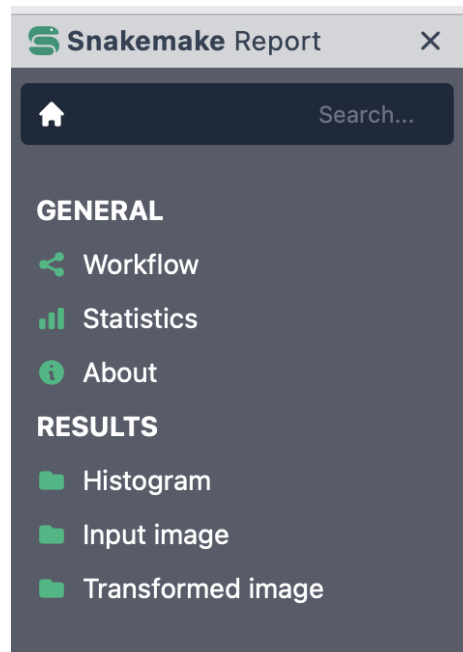## Scale your workflow easily to many files + conditions

Wildcards `{ }` are variables/placeholders that are tracked by snakemake. They allow us to generalize a rule to many samples

- Parametrize file names and other content
- Dynamically matched between rules
- Link rules

```
shell:"""
    python scripts/transform.py \
        --image {input} \
        --transformation {wildcards.transformation} \
        --output {output}
    """
```

Lucas Diedrich (Mann Labs, Max Planck Institute of Biochemistry), CC-BY-SA 4.0

(Hands on) Snakemake reports

Reports enable us to summarize run statistics and results in a convenient, self-contained format. Plots can be easily added to the report by wrapping the output definition with the `report()` directive

Lucas Diedrich (Mann Labs, Max Planck Institute of Biochemistry), CC-BY-SA 4.0

(Opinionated) Do's and dont's

Do

- Document the project in a README

- Document + Validate your configuration settings

- Track your project + log the git commit hashes

- Use `snakefmt` and `snakemake --lint`

Do not

- Publish your workflow without rerunning it in a clean environment on a minimal dataset

- Import `snakemake` into your scripts

- Use notebooks in your workflow

Lucas Diedrich (Mann Labs, Max Planck Institute of Biochemistry), CC-BY-SA 4.0