Quantifying Model-Observation Similarity

Annual Meeting of the German Astronomical Society 2025

September 16th, 2025

Marco Bischoff, Susanne Pfalzner

Outline

Trans-Neptunian Objects and TNO classification

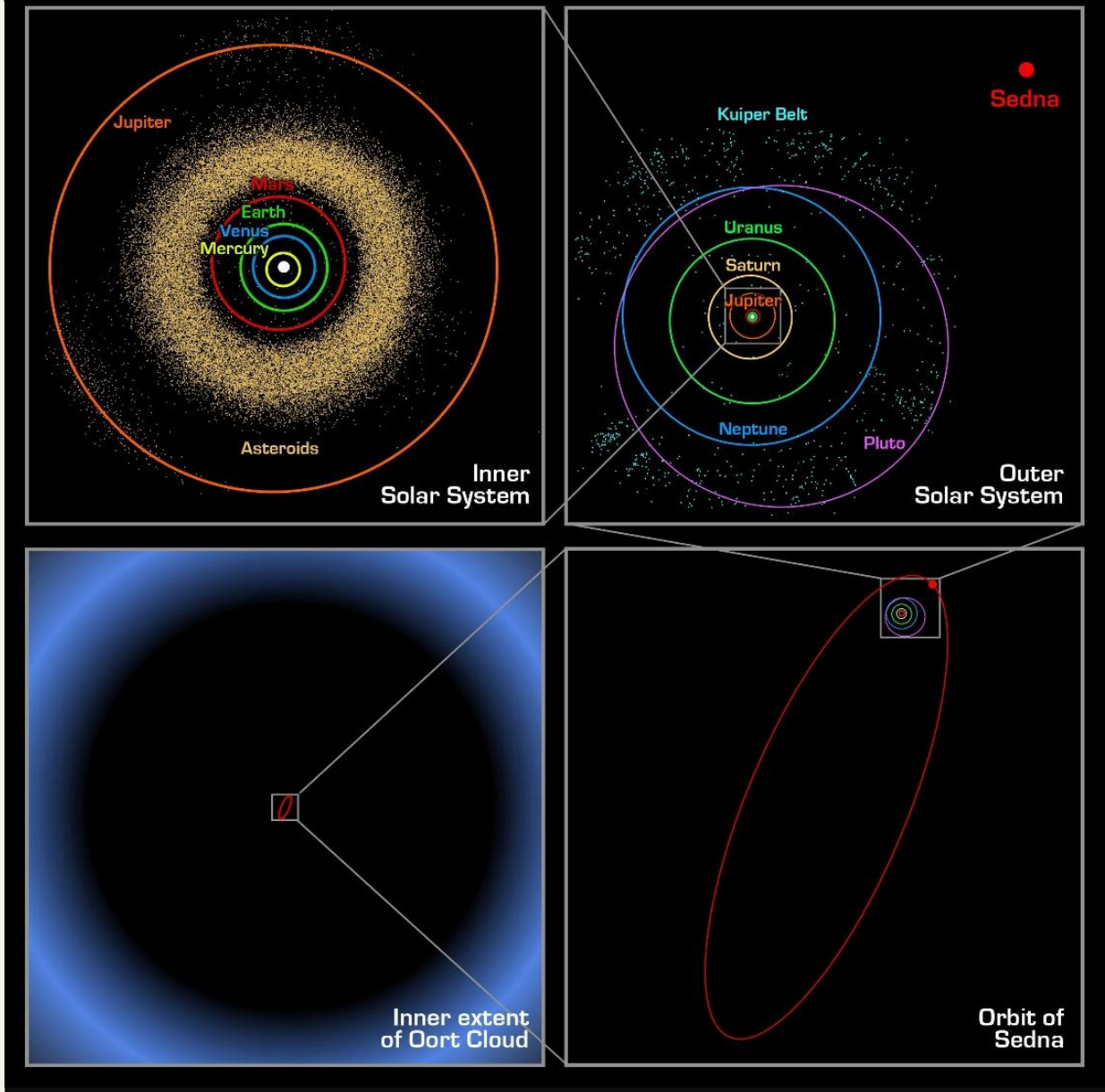
The stellar flyby model

Data pipeline for model-observation comparison

Development approach

What are Trans-Neptunian Objects?

- Minor planets with $a > 30$ au
- Some are very distant, highly eccentric and highly inclined
- About 6200 known
- Large observational bias



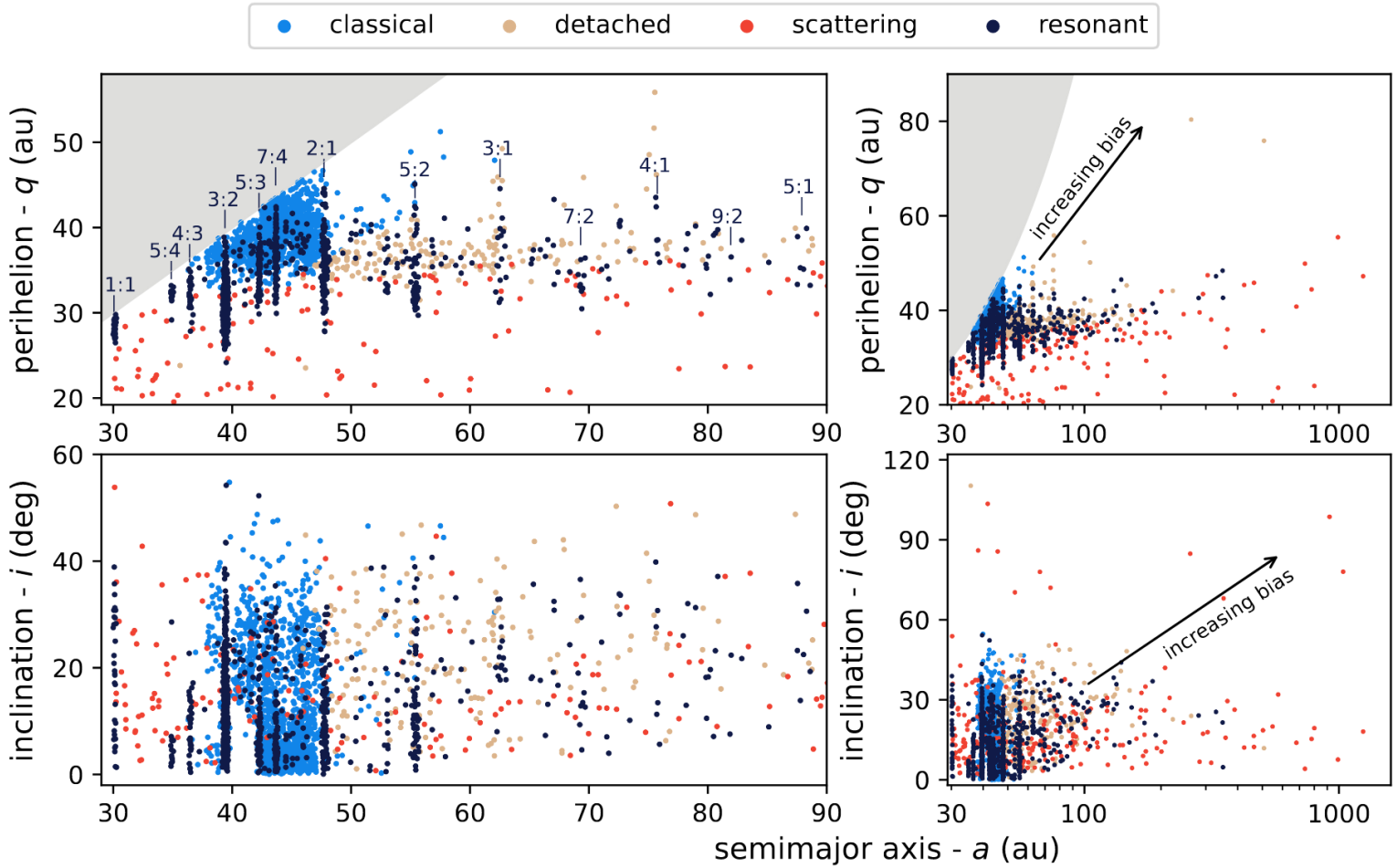
NASA/CALTECH: SEDNA ORBIT COMPARISONS

How are TNOs classified?

- Different classification schemes
- We use Gladman et al. (2008)

Hard to explain

- Detached TNOs
- Extreme TNOs
- Highly inclined TNOs

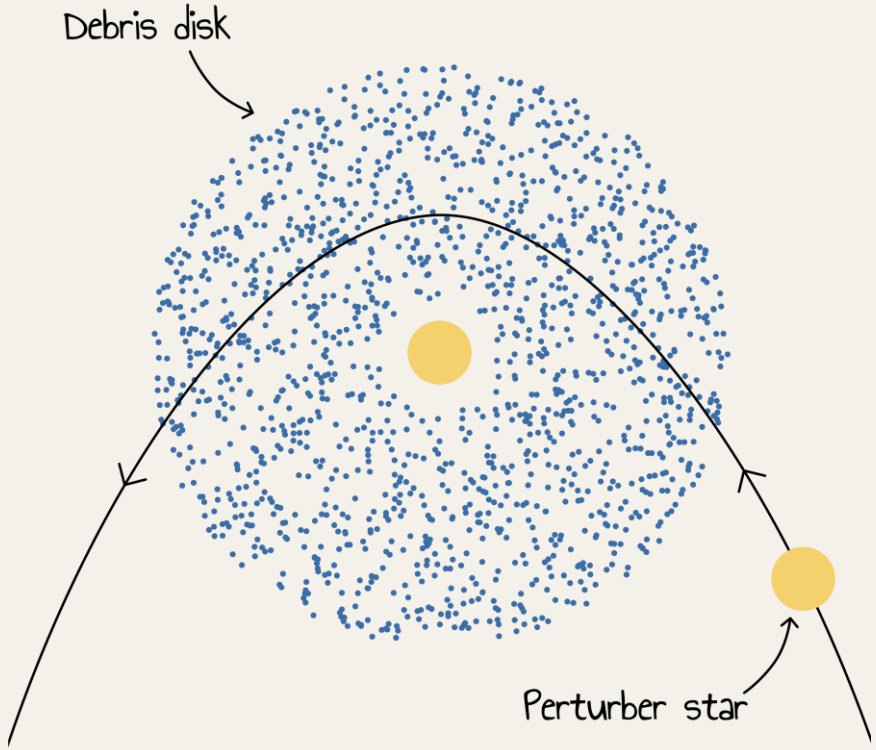


Volk, K., & Malhotra, R. (2024). Machine Learning Assisted Dynamical Classification of Trans-Neptunian Objects. DOI: 10.48550/arXiv.2405.05185

Gladman, B., Marsden, B. G., & Vanlaerhoven, C. (2008). Nomenclature in the Outer Solar System. Bibcode: 2008ssbn.book...43G

Models for TNO Dynamics

- Nice Model
  - Scattering of TNOs by chaotic migration of outer planets
- Planet Nine
  - An undiscovered planet orbiting far away
- Resonance Dropouts
  - ETNOs originated from Neptune's resonances
- Rogue Planet
  - A planet was ejected from the early Solar System
- **Stellar Flyby**
  - Another star disturbed the early Solar System



The Stellar Flyby Model

This video cannot be displayed in the PDF. Use the link instead.

https://www.youtube.com/watch?v=fOel5aWCRJs

Visualization by Sonja Habbinga

Pfalzner, S., Govind, A., & Portegies Zwart, S. (2024). Trajectory of the stellar flyby that shaped the outer Solar System. DOI: 10.1038/s41550-024-02349-x

Our Flyby Simulations

PARAMETER STUDIES

- Code: **DESTINY**
- Duration: **12,000 yr**
- Number of simulations
  - Broad study: 5472
  - Narrow study: 228
- No planets

Download at https://destiny.fz-juelich.de/

FLYBY SIMULATIONS

- Code: **REBOUND**
- Duration: **12,000 yr**
- More precise than parameter studies
- IAS15 Integrator
- No planets

https://doi.org/10.26165/JUELICH-DATA/9Z9K47

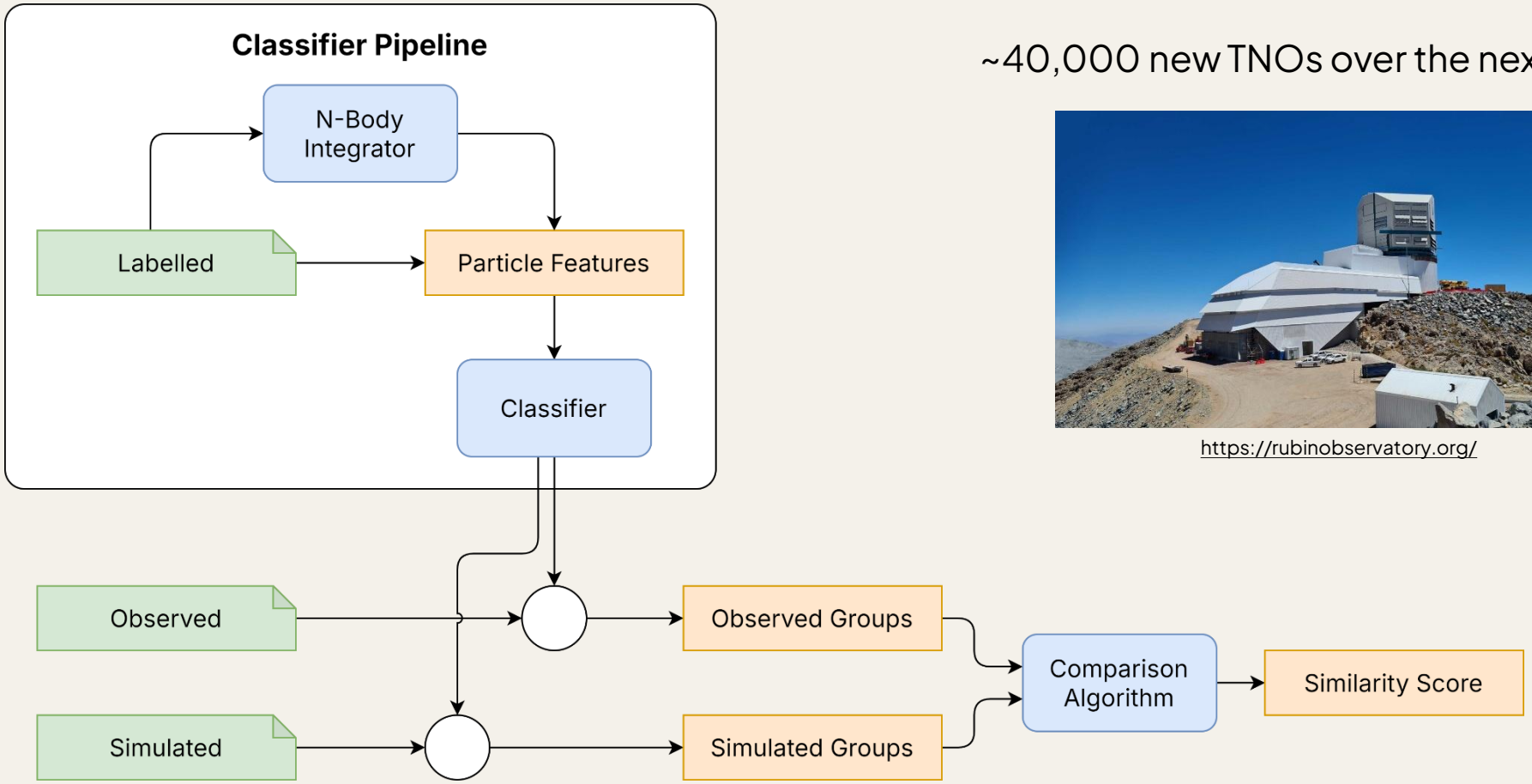
LONG-TERM SIMULATIONS

- Code: **GENGA**
- Duration: **1 Gyr**
  (soon 4.6 Gyr)
- Uses the flyby simulation as initial condition
- Includes outer planets

https://doi.org/10.26165/JUELICH-DATA/LGBORQ

→ Compare end state of simulation to observed TNOs
→ Rank simulations based on similarity to observations

Data Pipeline for Model-Observation Comparison



~40,000 new TNOs over the next 10 years

https://rubinobservatory.org/

Volk, K., & Malhotra, R. (2024). Machine Learning Assisted Dynamical Classification of Trans-Neptunian Objects. DOI: 10.48550/arXiv.2405.05185

Development Approach

Concepts and Tools

Working With Units

Pint

- You can use Quantity from `astropy.units` or `pint` instead of `float`
- No need to know which unit was specified originally

```python
@dataclass
class OrbitalBody:
    eccentricity: float
    # In astronomical units
    semi_major_axis: float
    # In degrees
    inclination: float

def example(body: OrbitalBody):
    inclination = body.inclination * math.pi / 180
    print(f"Inclination (rad): {inclination}")
```

```python
from astropy.units import Quantity

@dataclass
class OrbitalBody:
    eccentricity: Quantity
    semi_major_axis: Quantity
    inclination: Quantity


def example(body: OrbitalBody):
    inclination = body.inclination.to("rad")
    print(f"Inclination (rad): {inclination}")
```

Standardizing Columns

Library: `typed-polars`

- Less ambiguity
- Type casting
- Unit conversion
- Data validation
- Code completion
- Metadata handling

Column Definition

```python
class ExampleFields(FieldCatalog):
    inclination = TypedField(
        key="inclination",
        type=pl.Float64,
        short_name="i",
        title="Inclination",
        unit="degree",
        ge=0,
        le=180,
    )

    semi_major_axis = TypedField(
        key="semi_major_axis",
        type=pl.Float64,
        short_name="a",
        title="Semi-major axis",
        unit="au",
        ge=0,
    )
```

Data Frame Definition

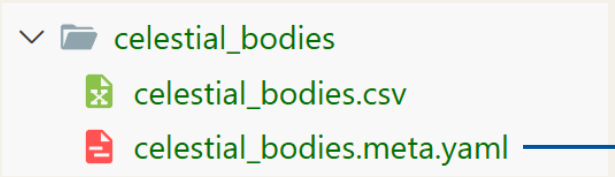
```python
class BodyFrame(RequireFrame):
    required_columns = [
        ExampleFields.inclination,
        ExampleFields.semi_major_axis,
    ]
```

celestial_bodies.csv

```
1   inclination,semi_major_axis
2   67.5,34.2
3   23.1,12.5
4   45.0,22.3
5   89.9,45.1
6   12.0,5.6
7   34.5,18.9
```

Annotating Datasets With Metadata

- Publish metadata file alongside the dataset
- Metadata includes
  - Reference to column definitions
  - Which columns are used
  - Documentation
- Read by `typed-polars` to infer the schema of the data frame

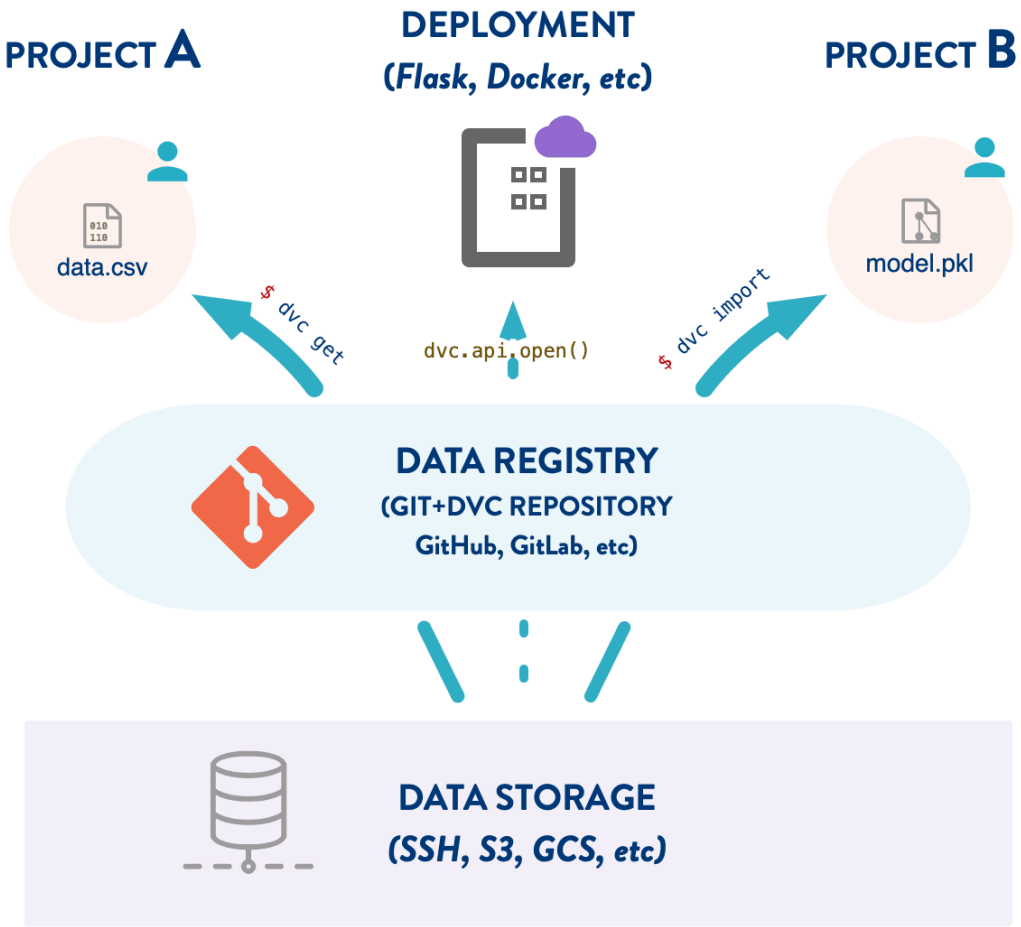


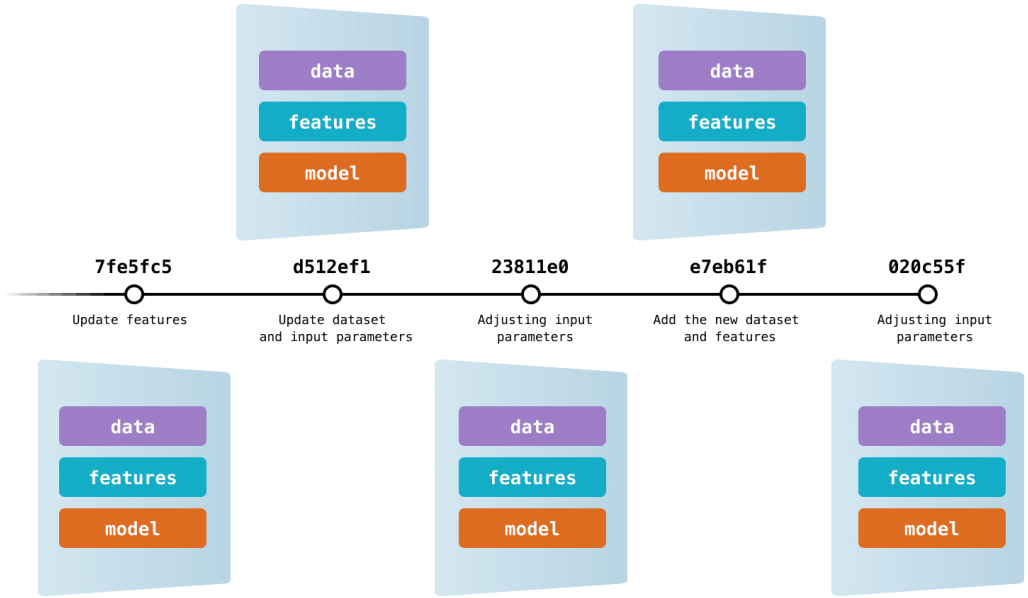
example.meta.yaml > ...

root-dataset.schema.json

```yaml
# yaml-language-server: $schema=https://astro-lab.pages.jsc.fz-
catalog: "https://www.github.com/bischoff-m/example-catalog/raw
name: "celestial_bodies"
version: "0.1.0"
publication_date: "2024-06-01"
files: "celestial_bodies.csv"
columns:
  - "inclination"
  - "semi_major_axis"
title: "Example Dataset of Celestial Bodies"
description: "A dataset containing information about various ce
author: "Max Mustermann"
contact: "max.mustermann@example.com"
license: "CC-BY-4.0"
related_publications:
  - "10.1234/example.doi"
software: "GENGA, Version: c91137d"
```

Data Version Control

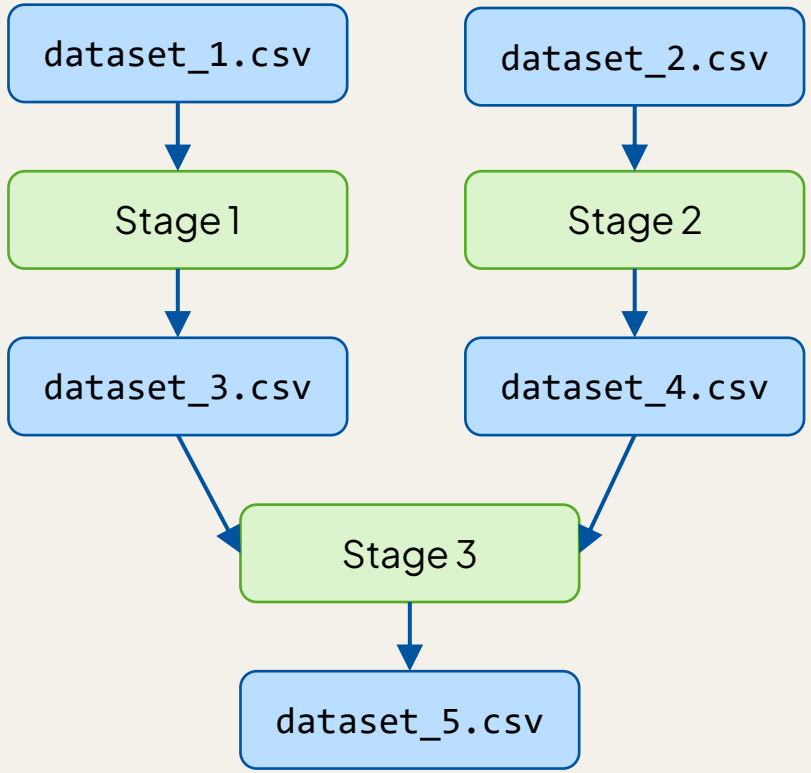




Images: https://dvc.org/

Pipeline Stages

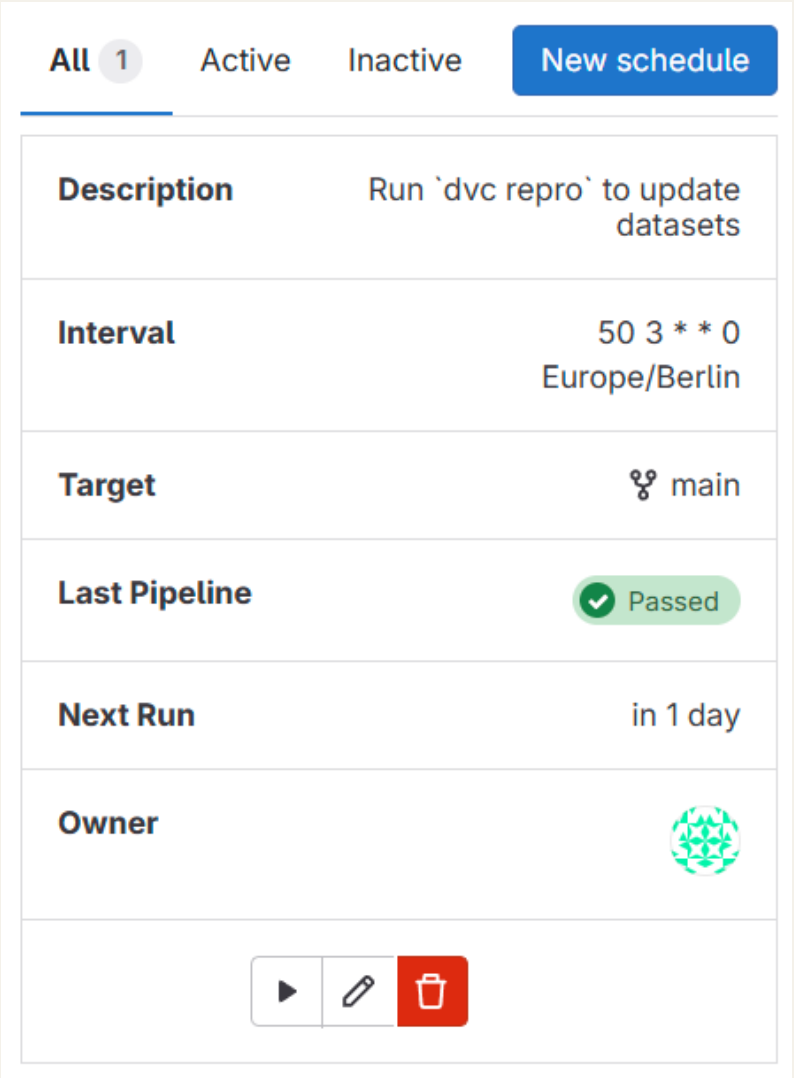
- Commands
  - `dvc pull` Downloads the datasets
  - `dvc repro` Computes the datasets from scratch
- Dependency management
- Separation of concerns
- Reproducible datasets
- No knowledge of the codebase needed



Continuous Integration

- Update pipeline once per week
- GitLab Runner executes pipeline

GitLab Pipeline



Thank you for listening

Find all links here