Porting GLASS to the Python Array API

ARCHER2 Celebration of Science 2026

Patrick J. Roddy
UCL Advanced Research Computing (ARC)

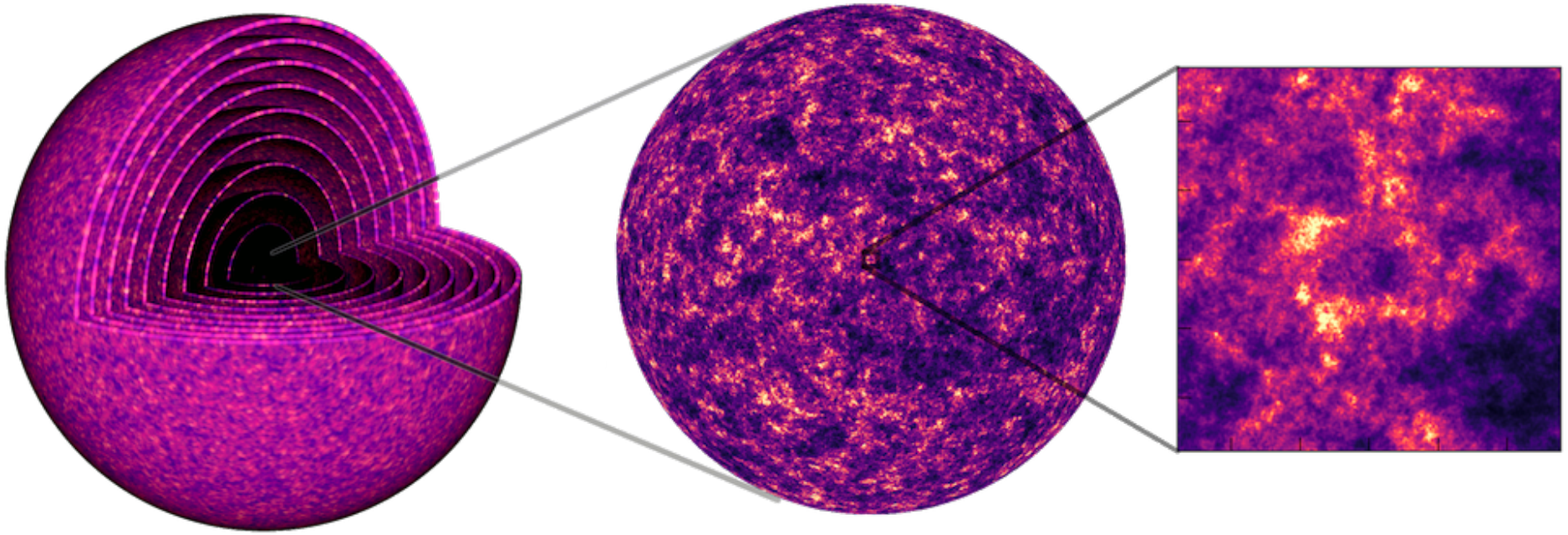
2026-03-19

GLASS: Generator for Large Scale Structure

Porting GLASS to the Python Array API

Overview

GLASS is a code used in cosmology to generate simulations of the full observable universe. In its current form, the primary user base of the code are collaborations for large galaxy surveys, which are a cornerstone of modern cosmology.



The Team

Nicolas Tessore
(Principal Investigator + Technical Staff)

Connor Aird
(Technical Staff)

Saransh Chopra
(Technical Staff)

Me (Co-Investigator + Technical Staff) ⤵

The Team

- Alessio Spurio Mancini (Co-Investigator)
- Arthur Loureiro (Co-Investigator)
- Benjamin Joachimi (Co-Investigator)
- Jason McEwen (Co-Investigator)
- Niall Jeffrey (Co-Investigator)
- Rebecca Martin (Research Administration)

The Aim

1. Transform GLASS into a GPU-enabled code.
2. Improve the performance of simulations with GPU acceleration.
3. Implement and adapt parallel processing elements that utilise GPU capabilities.
4. Optimise GLASS for modern GPU architectures.
5. Enable differentiable simulations using JAX (in part or in full).
6. Embed GLASS within N-body simulations running on GPU infrastructure for post-processing.

How Can This be Achieved

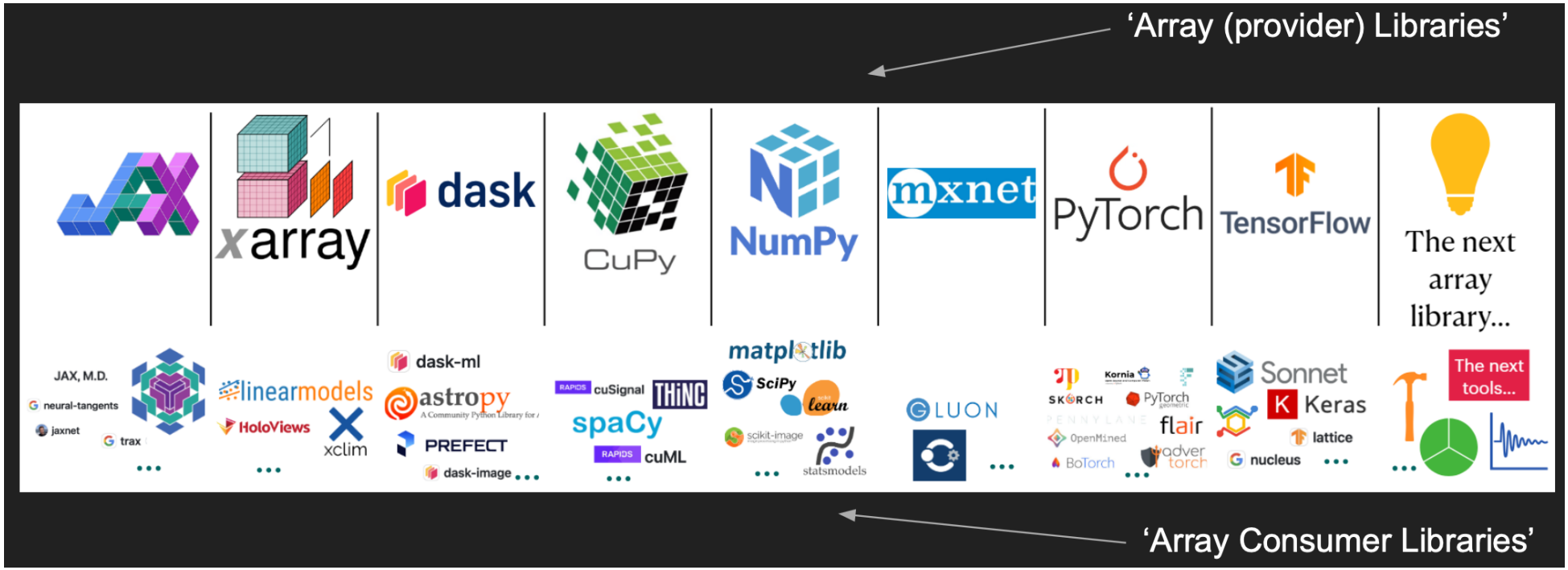
- Adopt the Python Array API Standard to write hardware-agnostic code.
- Core physics logic maps perfectly to the standard.
- The remaining gaps:
  - Handling RNG seeds consistently across CPU/GPU backends.
  - Falling back to library-specific code for FFTs and so on.
  - Minimising the cost of moving arrays between different devices.

Python Array API Standard

What Are Arrays

- N-dimensional, grid-like data structure.
- `numpy.ndarray` is the most well known.
- “Rectangular” shape data type.
- Fast, easy to manipulate.
- Ubiquitous.

The Array Ecosystem



Aaron Meurer: Python Array API Standard, SciPy 2023

Motivation: End Users

- Ability to switch array libraries without rewriting the whole code base.
- Avoid repeated transfers between array libraries or devices.
- Enable experimentation:
  - Test new hardware.
  - Use functionality specific to an array library.

Motivation: Array *Providing* Libraries

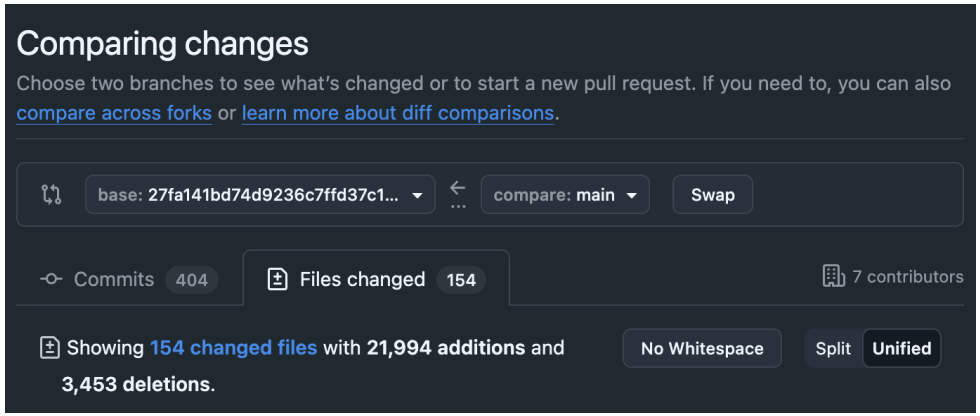
- Existing libraries:
  - Interoperability with new consuming libraries.
  - Collaborative API decisions with other array libraries
- New libraries:
  - Concrete API to implement.
  - Guaranteed compatibility with consuming libraries.

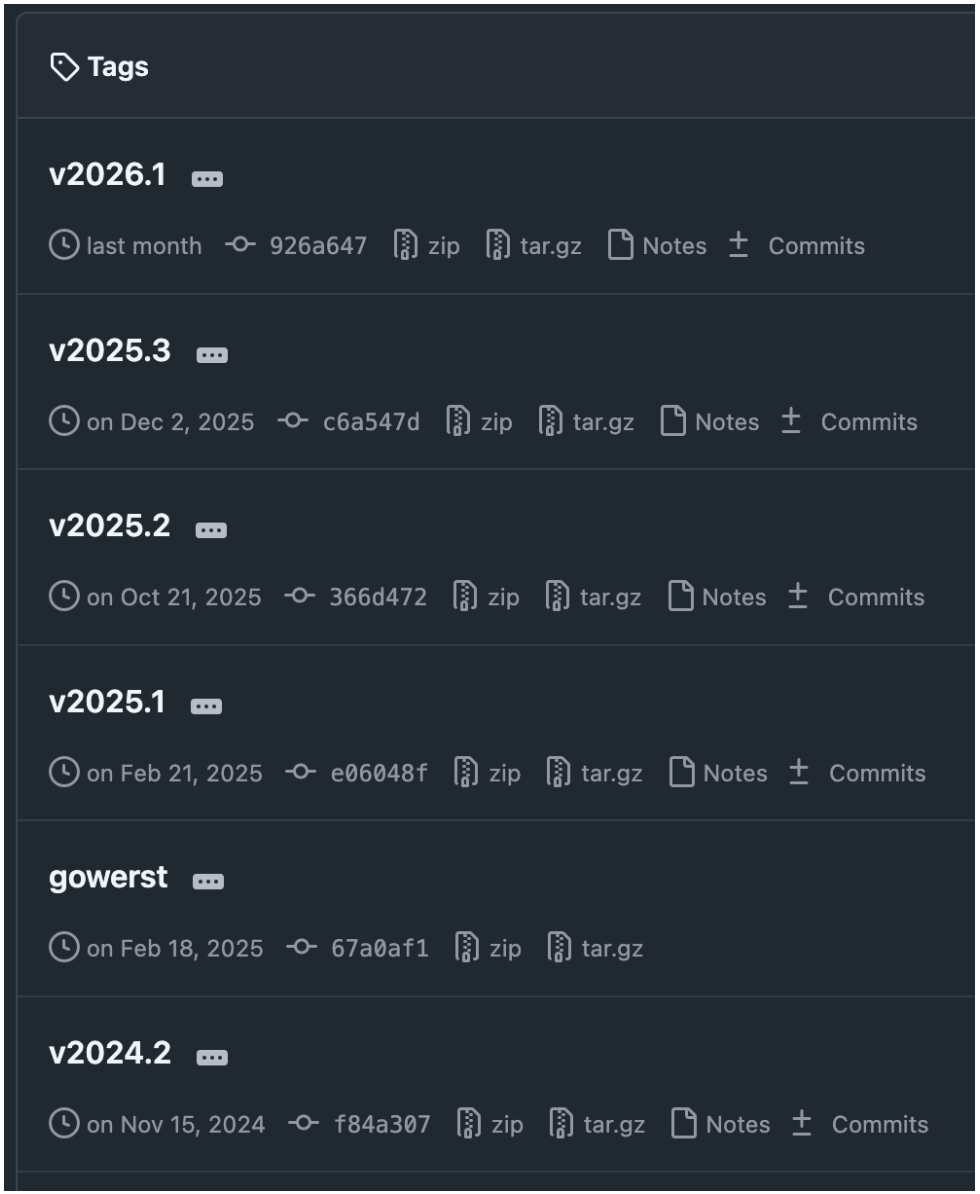
Motivation: Array *Consuming* Libraries

- Support hardware-accelerated and vendor-specific array types.
- But without additional maintenance burden.
- Library remains relevant when the community switches to a new array library.

Progress

Changelog





Porting GLASS to the Python Array API

Testing on Different Array Backends

./noxfile.py run as `uv run nox -s tests-3.14`.

```python
"""Nox config."""

import os
import pathlib
import shutil

import nox
import nox_uv

# Options to modify nox behaviour
nox.options.default_venv_backend = "uv"
nox.options.reuse_existing_virtualenvs = True
nox.options.sessions = [
    "lint",
    "tests",
]

ALL_PYTHON = [
```

Custom Typing

./glass/_types.py pending array-api-typing  .

```python
from typing import TYPE_CHECKING, Any

if TYPE_CHECKING:
    from collections.abc import Sequence
    from typing import ParamSpec, TypeAlias, TypeVar

    import jaxtyping
    import numpy as np

    from array_api_strict._array_object import Array
    from array_api_strict._dtypes import DType

    import glass.jax
    from glass import _rng

    P = ParamSpec("P")
    R = TypeVar("R")
    T = TypeVar("T")
```

Benchmarking

Perform regression tests by using pytest-benchmark .

Run regression test 8m 23s

```
199 ----------------------------------------------------------------------------
200 Performance has regressed:
201     test_generate[numpy-None] (0001_c4ad4a5) - Field 'mean' has failed PercentageRegressionCheck: 83.983018854 > 5.000000000
202     test_uniform_positions[numpy] (0001_c4ad4a5) - Field 'mean' has failed PercentageRegressionCheck: 6.962950314 > 5.000000000
203 ----------------------------------------------------------------------------
204
205
206
207 ------------------------------------------------- benchmark: 54 tests -------------------------------------------------
208 Name (time in us)                                          Mean                 StdDev            Rounds
209 -----------------------------------------------------------------------------------------------------------------------
210 test_cls2cov[numpy] (0001_c4ad4a5)                      402.0254 (2.85)       14.2538 (2.40)         94
211 test_cls2cov[numpy] (NOW)                              344.6912 (2.44)        7.6379 (1.29)       6933
212 test_displacement[numpy] (0001_c4ad4a5)            103,969.9087 (737.32)    664.9928 (111.94)       49
213 test_displacement[numpy] (NOW)                     103,081.6571 (731.02)    637.0859 (107.24)       49
```

JAX does not allow array mutation, so need to be careful not to regress NumPy as a result of enabling JAX.

Additional Array API Utilities

glass/_array_api_utils.py for functions not in the specification nor in array-api-extra .

```
"""
Array API Utilities for glass.
==============================

This module provides utility functions and classes for working with multipl
backends in the glass project, including NumPy, JAX, and array-api-strict.
functions for importing backends, determining array namespaces, dispatching
number generators, and providing missing functionality for array-api-strict
xp_additions class.

Classes and functions in this module help ensure consistent behavior and co
across different array libraries, and provide wrappers for common operation
integration, interpolation, and linear algebra.

"""

from __future__ import annotations

```

Random Number Generator

glass/_rng.py

```
"""
Random Number Generation Utilities for glass.
=============================================

This module includes functions for dispatching random number generators usi
seeds. The chhoice of rng generator is determined based on the array librar
the user.

"""

from __future__ import annotations

from typing import TYPE_CHECKING

if TYPE_CHECKING:
    from types import ModuleType

    from glass._types import DTypeLike, FloatArray, IntArray, UnifiedGenera
```

JAX Support

./glass/jax.py

```
"""Wrapper for JAX RNG with a NumPy-like interface."""

from __future__ import annotations

import math
import threading
from typing import TYPE_CHECKING

import jax.dtypes
import jax.numpy as jnp
import jax.random
import jax.scipy
import jax.typing

if TYPE_CHECKING:
    from jaxtyping import PRNGKeyArray
    from typing_extensions import Self

```

Further Work

- Test on CuPy following v14 release .
- Create scripts to benchmark on ARCHER2 using different array backends.
- Contribute remaining required functions to array-api-extra .
- Implement spherical harmonic transforms that are array API compatible.
- Tech debt…

https://
paddyroddy.github.io/
talks