Quansight Labs

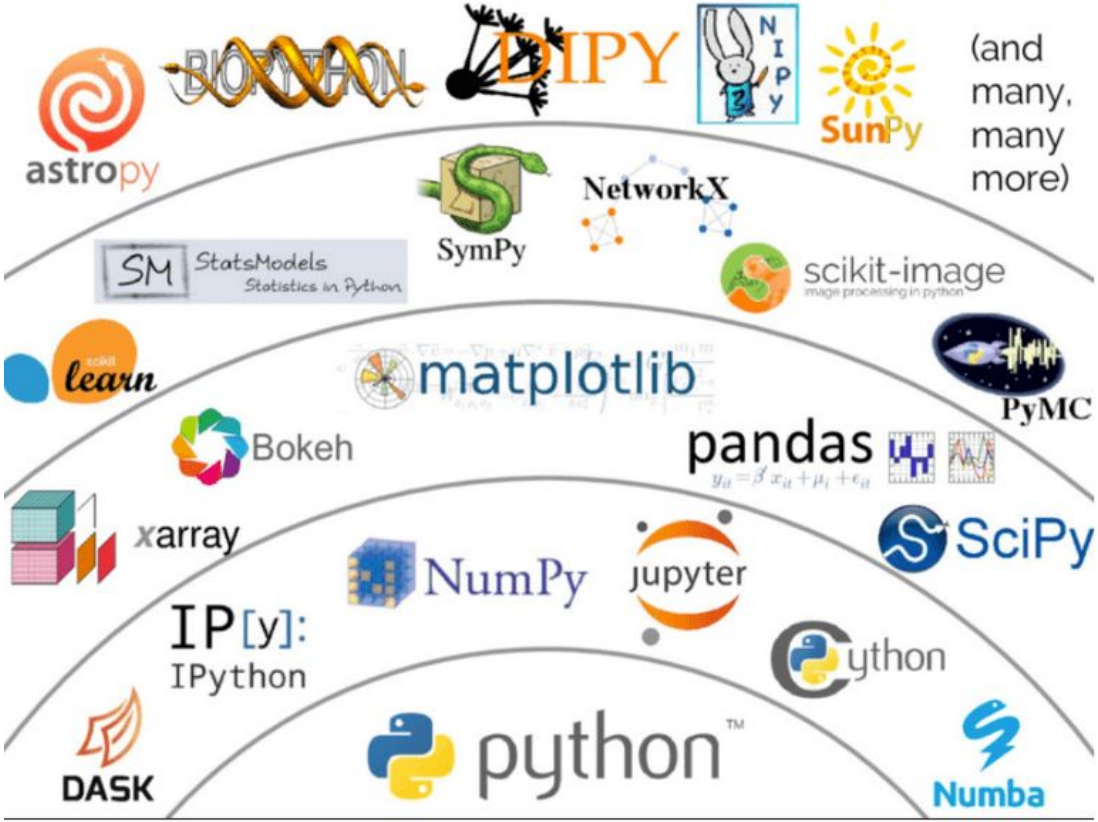
The road ahead for scientific computing with Python

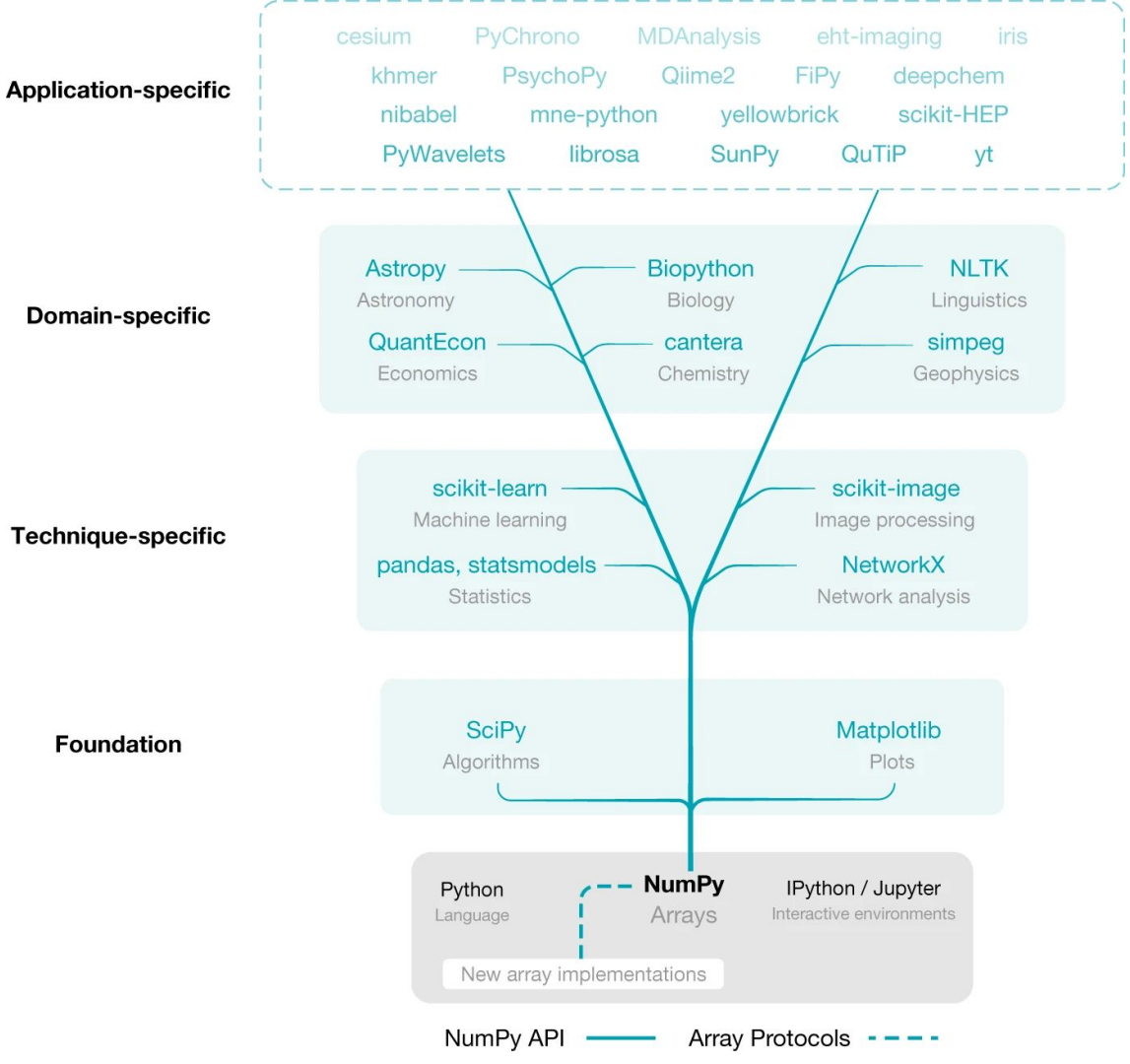
Ralf Gommers

7 Dec 2021

Quansight Labs

The open source PyData ecosystem





Quansight Labs

Quansight Labs

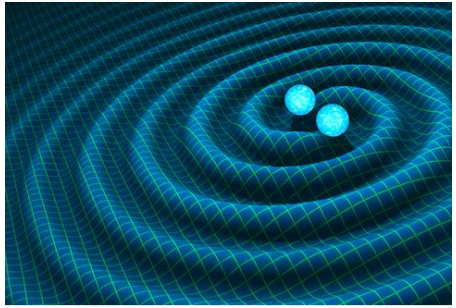
The state of PyData today

1. #1 language for scientific computing, data science, ML & AI
2. Estimated user base: 25-40 million users
3. Incredibly broad ecosystem & high-profile scientific successes

Libraries.io

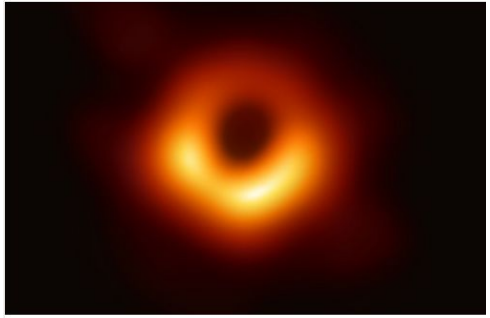
Dependent packages	30.2K
Dependent repositories	84.9K

DETECTION OF GRAVITATIONAL WAVES



In 1916, Albert Einstein predicted gravitational waves; 100 years later their existence was confirmed by LIGO scientists using NumPy.

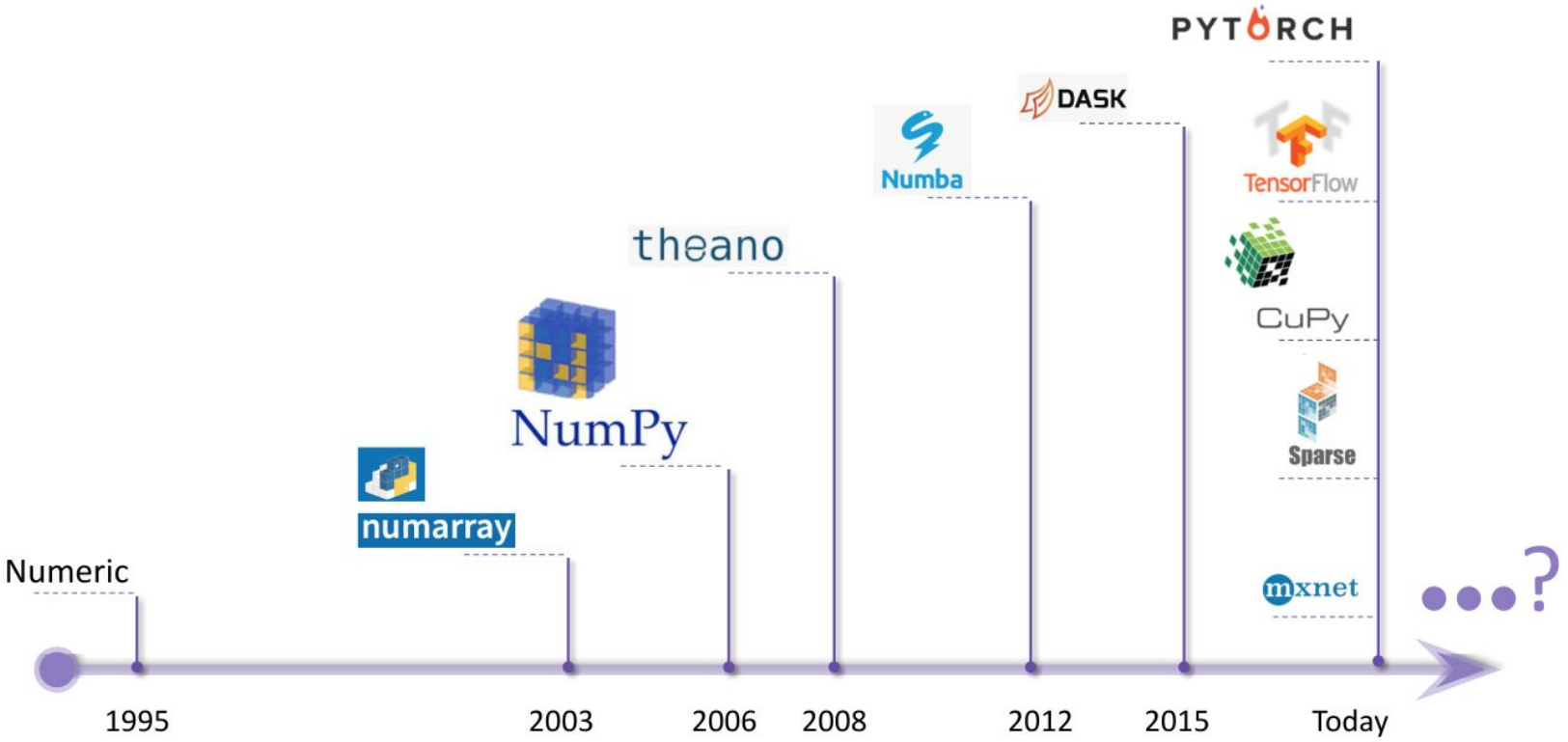
FIRST IMAGE OF A BLACK HOLE



How NumPy, together with libraries like SciPy and Matplotlib that depend on NumPy, enabled the Event Horizon Telescope to produce the first ever image of a black hole

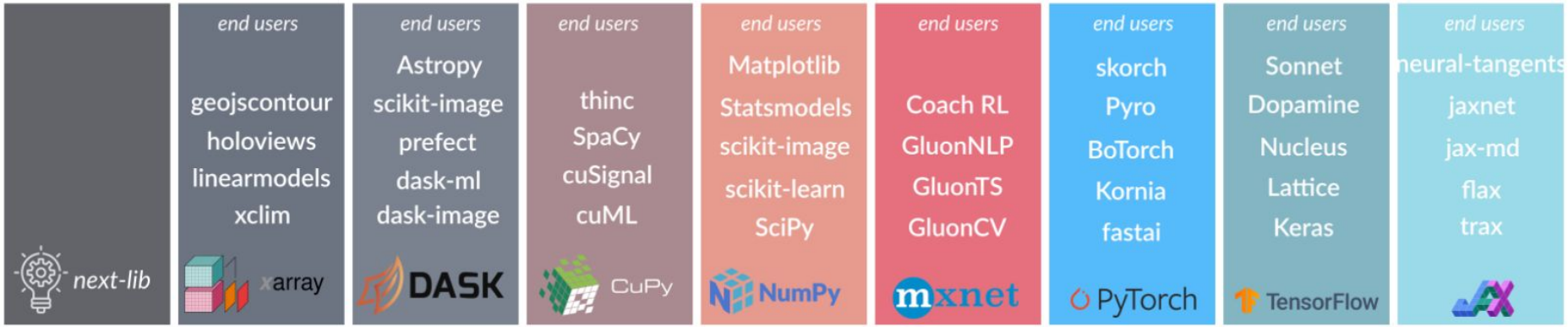
Quansight Labs

Array-based computing in Python



Quansight Labs

Today's Python data ecosystem



Many other array libraries - fragmentation!

Why?

Technical: performance, GPU & distributed computing, and autograd.

Social/commercial: large tech companies want control, and to move fast.

Key technical challenges for the PyData ecosystem

Quansight Labs

My top 5 technical challenges

1. Fragmentation in array/tensor libraries & support for heterogeneous computing

2. (lack of) parallel execution capabilities in the NumPy-based PyData stack

3. Packaging: PyPI (the Python Package Index) imposes serious constraints

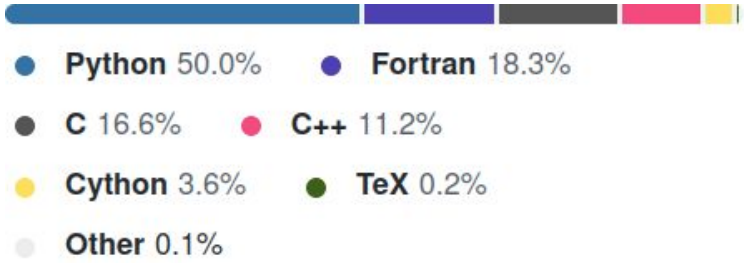
4. Performance for algorithms that can't be expressed in vectorized form.

5. Technical debt: old Fortran 77 & C code hard to maintain; long double support probably needs removing; distutils is end-of-life

Quansight Labs

(5) technical debt

SciPy: old Fortran 77 & C code hard to maintain



These are just 3 out of many examples

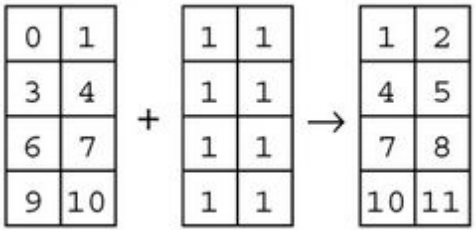
long double: basically obsolete (MSVC on Windows, macOS on arm64 and Linux/aarch64 all alias it to double) and takes a ton of time to maintain. Needs to be replaced by proper quad-precision float128.

distutils will be removed in Python 3.12 (see PEP 632), packages with a lot of compiled code should migrate to Meson or CMake.

Quansight Labs

(4) performance of numerical code

Vectorization



Use compiled code

```cpp
#include <pybind11/pybind11.h>

int add(int i, int j) {
    return i + j;
}

PYBIND11_MODULE(example, m) {
    m.doc() = "pybind11 example plugin"; // optional module docstring

    m.def("add", &add, "A function which adds two numbers");
}
```

Python compilers

Cython

Pythran

Numba

Python interpreters

CPython

pypy

GraalVM

Plus Cinder, Pyston, and more -- very experimental, and limited gains for numerical code

Quansight Labs

(3) packaging constraints

- PyPI does not offer fundamentals for scientific computing like BLAS/LAPACK or OpenMP. It was not designed for non-Python dependencies.

- PyPI + pip != a package manager
  As a result, PyPI's author-led model cannot ensure that a set of packages was built with a consistent set of rules (e.g., a single compiler toolchain)

- PyPI serves 3 distinct purposes:
  - Flow of source code from authors to redistributors (OK)
  - End-user binary installs via wheels (OK-ish)
  - End-user installs from source (very problematic)

Quansight Labs

(2) lack of parallelism in PyData stack

1. NumPy: single-threaded, except for calls to BLAS/LAPACK
2. SciPy:
    a. single-threaded by default, except for calls to BLAS/LAPACK
    b. `workers=` API to let user enable multiple threads
3. Scikit-learn:
    a. Most functionality single-threaded, with `n_jobs=` API to let user enable multiple threads
    b. Starts to use OpenMP more, for automatic parallelization
    c. Complex control (see `threadpoolctl` package) of NumPy/SciPy's BLAS and LAPACK libraries to prevent oversubscription in the presence of multiprocessing on top of multi-threading.

With 32/64-core CPUs becoming more common, this isn't a tenable situation.
A common threading layer is needed.

Quansight Labs

(1) fragmentation & heterogeneous computing

We need common APIs to address CPUs, GPUs, TPUs, FPGAs & other emerging hardware. Separate libraries for each type of hardware is not composable.

To address this, we created a standardization effort - see https://data-apis.org

DataAPIs

Consortium for Python Data API Standards

Sponsors:

intel, Quansight, Microsoft, D E Shaw & Co, TensorFlow, LG Electronics

Participation: maintainers of all of the most popular Python array and dataframe libraries

Coordination: Quansight Labs

Consortium for Python Data API Standards

Goals for and scope of the array API

Goal 1: enable writing code & packages that support multiple array libraries

Goal 2: make it easy for end users to switch between array libraries

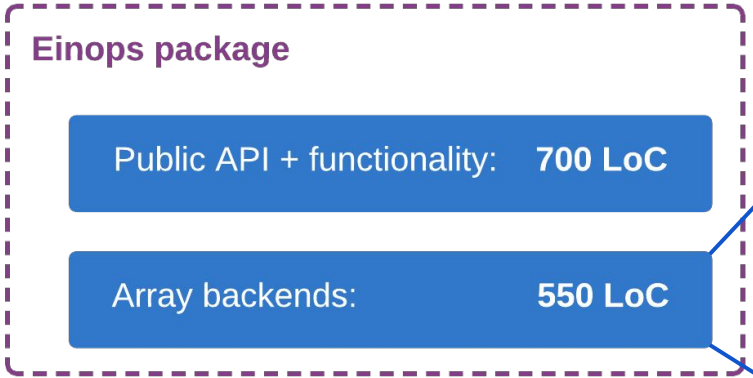
In Scope

1. Syntax and semantics of functions and objects in the API
2. Casting rules, broadcasting, indexing, Python operator support
3. Data interchange & device support

Out of Scope

1. Execution semantics (e.g. task scheduling, parallelism, lazy eval)
2. Non-standard dtypes, masked arrays, I/O, subclassing array object, C API
3. Error handling & behaviour for invalid inputs to functions and methods

Use case: the einops package




```python
class AbstractBackend:
    def transpose(self, x, axes):
        return x.transpose(axes)
class TorchBackend(AbstractBackend):
    framework_name = 'torch'
    def transpose(self, x, axes):
        return x.permute(axes)
class KerasBackend(AbstractBackend):
    framework_name = 'keras'
    def transpose(self, x, axes):
        return self.K.permute_dimensions(x, axes)
```

- A popular package for array manipulation
- Supports 8 popular array/tensor libraries.
- *Almost 50% of the code can be removed through array API standardization!*

Array API - participation & adoption

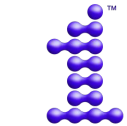
API adoption done or close to done

In numpy.array_api namespace

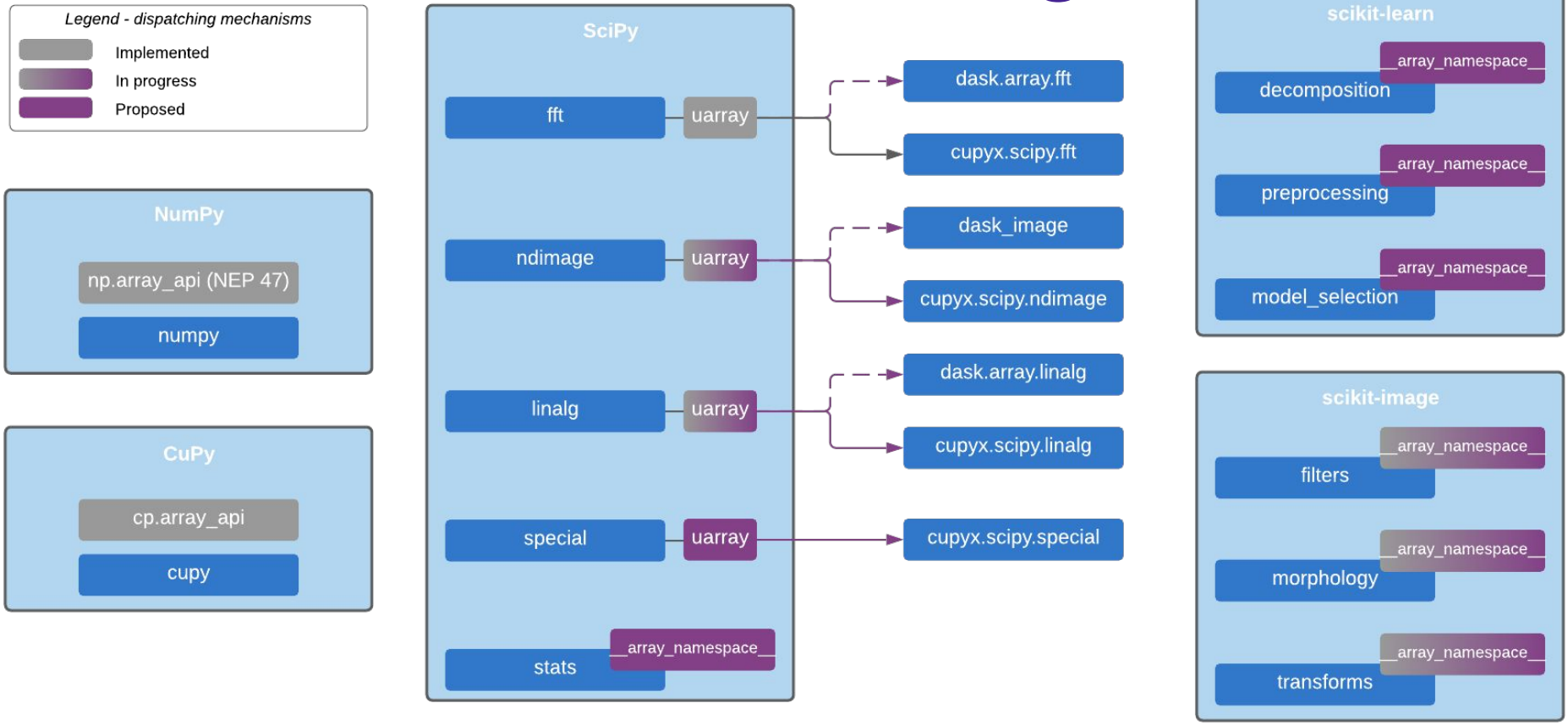
In torch (main) namespace

In cupy.array_api namespace

Design participation, adoption in progress or being discussed



Distributed & GPU arrays with SciPy, scikit-learn and scikit-image



Key social challenges for the PyData ecosystem

Quansight Labs

My top 3 social challenges

1. Sustainability of key projects
2. Big tech has discovered PyData
3. Academia still needs to find its role

Quansight Labs

Sustainability

For NumPy, SciPy, scikit-learn: each ~10-20 active maintainers, O(10-20 million) users

Funding is still very hard to obtain. Number of funded devs:

- NumPy: 1 full-time, 3 part-time
- SciPy: 2 part-time
- Scikit-learn: 2 full-time, 3 part-time
- Most other projects: volunteer-only

Burnout of maintainers is a real risk/problem.

Funding is mostly coming from independent funders: Sloan Foundation, Moore Foundation, Chan-Zuckerberg Institute.

Diversity & inclusivity remains challenging: >90% of maintainers are white men

Quansight Labs

Big tech has discovered PyData

Big tech focuses on a narrower set of capabilities for deep learning than is needed for scientific computing.

Ex.: complex dtype support only arrived in 2021)

Corporate-backed vs. community-driven culture:

- Tension in funded vs. volunteer efforts
  - “Move fast and break things” vs.
  - “Backwards compatibility for 10+ year old scientific models”

- Bandwidth problem:
  TensorFlow/PyTorch have ~200-300 full-time engineers, RAPIDS >50.

Quansight Labs

Role of academia

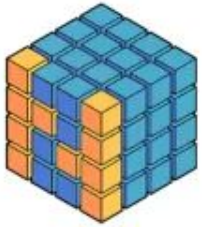
Academia relies heavily on scientific open source, however:

- Open source contributions are valued significantly less than papers today.
  Even if often *much* more impactful (ex: 13% of all papers on ArXiV used Matplotlib)
- Institutional funders are behind the times. The investment in, e.g., exoscale facilities is >100x (or >1000x ?) that in open source.
- Publishers & reviewers need to require source code for papers (slowly happening).

Recommendations:

1. Work on fixing career paths and funding
2. Focus on
   a. missing science-specific key pieces (e.g., sparse linear algebra)
   b. production-quality code
   c. skills building for grad students

A look at NumPy's technical and social roadmap

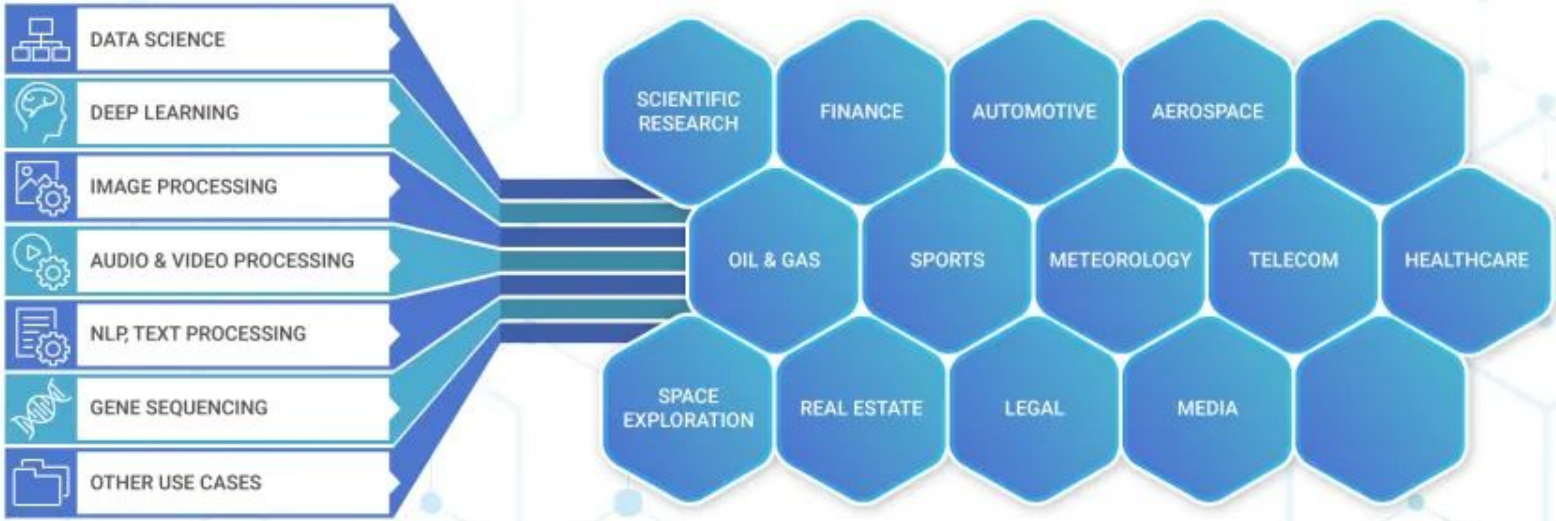


24
million
downloads (Dec, 2019)

20–40
million
users worldwide

$1,4
million
lifetime funding

~10
devs
active maintainers



Sources: anaconda.com, pypistats.org
Special thanks: Ralf Gommers, Inessa Pawson, Shaloo Shalini

Quansight Labs

Where is NumPy going - technical

Interoperability

Array API standard support

Extensibility

Easier custom dtypes

Performance

SIMD acceleration on:
x86, arm64, PPC, …?

C++ (?)

Just dipping our toes in the water here - so far it was just Python and C

Platform support

PPC, AIX, s390x,
cross-compiling to embedded ARM systems, ...

Type annotations

Main namespace annotations just completed

Quansight Labs

Where is NumPy going - community

NumPy is a community-driven project. Most people are volunteers.

Goals for the coming 1-2 years include:

- Grow more (autonomous) teams: web, docs, triage, f2py, SIMD, API, ...
- Further increase the diversity of the team & inclusivity of the project
- Better communication channels: move to Discourse, with separate user forum
- Increasing active alignment across PyData projects (from processes like when to drop Python versions, to hairy technical topics like parallelism)

Getting involved

Quansight Labs

How can you get involved?

Yes, you can! People are friendly, and you have talent and knowledge that has value!

Domain knowledge (linear algebra, special functions, statistics, etc.) is just as important as coding skills.

Quansight Labs

How can you get involved?

How do you pick a project to start contributing too?

- First, pick something that interests you!
- Look at the activity on GitHub - do pull requests get reviewed and merged in a reasonable amount of time? Is the feedback constructive and given in a friendly manner?
- Small vs. large projects

Thank you!

Find me at: ralf.gommers@gmail.com, rgommers, ralfgommers