Awkward Family

extending functionality through interrelated Python packages

Jim Pivarski, Princeton University

IRIS-HEP Institute for Research & Innovation in Software for High Energy Physics

PRINCETON UNIVERSITY

Introduction

In the 5+ years since their inception, Uproot and Awkward Array have become a foundation layer for Pythonic HEP analysis, both as direct user interfaces and as dependencies for physicist-facing frameworks. Although this means that the software is achieving its mission, it also puts the need for stability in conflict with new, experimental developments. Boundaries must be drawn between the code that needs to stay robust and the code that implements new ideas.

To the extent that is possible, new developments are now being persued in ***new packages***, rather than extensions of the existing packages.

Maintaining interoperability

Coherence of a single package can be maintained with git and continuous testing, but between packages, wide ranges of versions must work together.

Strict public/private boundaries are essential, as is common ground testing, such as CoffeaTeam/integration-test and Scientific Python Nightly Wheels (SPEC-4).

Why?

Small, well-scoped packages:

- can be installed à la carte;
- promote (require) clear interfaces;
- allow for more experimentation: if an idea doesn't work out, it doesn't have to be maintained;
- more visibility to new developers.

Where to ask questions?

This is still an open issue. Help with HEP software is decentralized across Gitter, GitHub, GitLab, Mattermost, Slack, RootTalk, Launchpad… Ask me about my "hep-help" project to try to link them together.

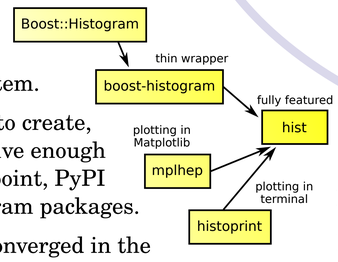
Why not?

More confusing to users:

- which ones are still being maintained?
- how can they be used together?
- where to ask questions?

Too many packages?

A single package can act as an interface to a whole ecosystem.



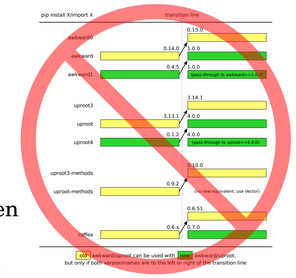
Histogram packages are easy to create, but hard to make comprehensive enough to become a standard: at one point, PyPI hosted 20 HEP-related histogram packages.

Several histogram packages converged in the Scikit-HEP ecosystem: boost-histogram can fill quickly but has no plotting capabilities; mplhep only plots. Instead of consolidating these features into one package, we defined common protocols for interoperability and wrapped all functionality into a package that can be imported and installed as "hist".

How not to roll out major interface changes:

Do not change package names. At least, do not reuse old names after a package with a different interface has used that name.

To give users a chance to gradually adapt to a major interface change, we named Awkward 0.x as "awkward0" and 1.x as "awkward1" during a transition period, so they could both be imported in the same session. However, third-party libraries were dependencies of name/version combinations that became invalid when Awkward 1.x became "awkward". Awkward 1.x → 2.x used semantic versioning in the normal way and was much smoother.

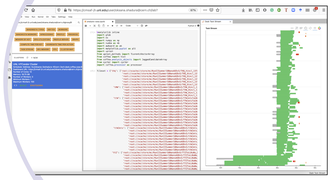


Kaitai Struct

Awkward Array backend

Kaitai Struct describes data formats in YAML and generates deserializers in many languages. Manasvi Goyal added a backend that generates Awkward Arrays in C++ and presents them in Python, with more automation than any other backend. See her poster on Monday.

dask-awkward



dask-awkward adds a new high-level collection to Dask so Awkward Array computations can be delayed and distributed.

Coffea, a physics toolset developed at Fermilab, has fully integrated dask-awkward and many new analyses in CMS are now distributed using Dask. See Lindsey Gray's 11am plenary on Thursday for more.

VECTOR

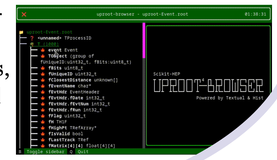
Vector transforms coordinates and computes useful functions for 2D, 3D, and 4D (Lorentz) vectors objects, NumPy arrays of vectors, and Awkward Arrays of vectors. As a dependent library, Vector tests Awkward's "behavior" mechanism, which specifies how functionality is added to deeply nested data.

hep▸convert

hepconvert is a library and command-line program for high-level conversions between file formats of interest to HEP, accumulating statistics, and copying files with changes, such as adding and dropping columns. See Zoë Bilodeau's poster on Thursday.

UPROOT5-BROWSER

uproot-browser is a text-user interface that lets you navigate ROOT files, view and save plots, and has a command-line interface for scripting.



[ragged]

ragged restricts Awkward Array data types to only lists of lists of numbers, but in a way that conforms to DataAPI specifications, which might allow them to be used in xarray.

DataAPIs

xarray

formulate

formulate parses and converts between ROOT TTree::Draw syntax, NumExpr, and array slices. It will be used as a plug-in for Uproot to compute and cut with these languages.

Lark parser

"awkward-pandas"

pandas RAPIDS

awkward-pandas wraps Awkward Arrays as Pandas Series and provides Awkward functionality through an accessor. But since Awkward Arrays are zero-copy compatible with Apache Arrow and Pandas is moving toward Arrow, this library might not be needed in the future. We're also exploring interoperability with Polars and RAPIDS CuDF, which are already backed by Arrow.

Awkward Array.jl

AwkwardArray.jl is an implementation of Awkward Array in Julia, with zero-copy interoperability between the two languages. It lowers the barrier to mixing Python and Julia, allowing Uproot to be replaced with (faster) UnROOT.jl. See Ianna Osborne's 2:30pm Track 1 talk on Wednesday.

julia UnROOT.jl

Awkward Array

Awkward Array is a library for manipulating lists, nested records, missing data and mixed types with NumPy-like idioms. It has built-in support for Numba acceleration, JAX autodiff, Apache Arrow and ROOT RDataFrame interoperability, and GPUs.

uproot

Uproot is a pure Python implementation of ROOT I/O, reading and writing TTrees and RNTuples as arrays: NumPy, Awkward, and Pandas. Uproot gets histograms through hist, remote data through fsspec, and uses dask-awkward and awkward-pandas.

When the interface must change

Define a deprecation policy and post it in a public place. Awkward Array's policy is on its roadmap/wiki:

- public interface can only change on minor version boundaries (2nd digit);
- warnings must be raised in the code two minor versions in advance;
- minor versions must be at least (but close to) two months apart.

Deprecation warnings are only possible if users can toggle between behaviors with different names!

This project is supported by the National Science Foundation (NSF) cooperative agreements OAC-1836650 and PHY-2323298 (IRIS-HEP), grant OAC-1450377 (DIANA/HEP), PHY-2121686 (US-CMS LHC Ops), and OAC-2103945 (Awkward Array). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.