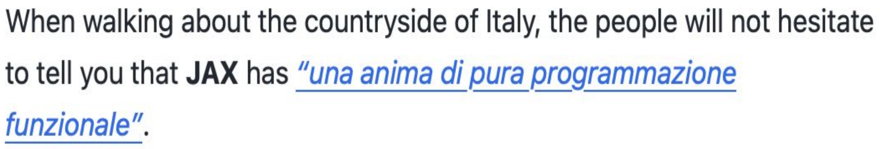
IRIS-HEP Topical Meeting - 26th February 2024

Enabling auto-differentiation for the Scikit-HEP ecosystem (and more)

Saransh Chopra (University of Delhi / Princeton University)
Supervisor: Dr. Jim Pivarski (Princeton University)

JAX - a quick introduction

- *JAX is Autograd and XLA, brought together for high-performance numerical computing.* JAX has 3 layers of API - XLA, LAX, and JAX
- The high level API (`jax.numpy`) is basically JIT-compileable and differentiable numpy on CPUs, GPUs, and TPUs; JAX also has several functions to carry out autodiff (`jvp`, `vjp`, `grad`, `jacfwd`, `jacrev`, …)
- One can make custom data containers compatible with JAX API by registering a way to flatten and unflatten them (registering `pytree` nodes)



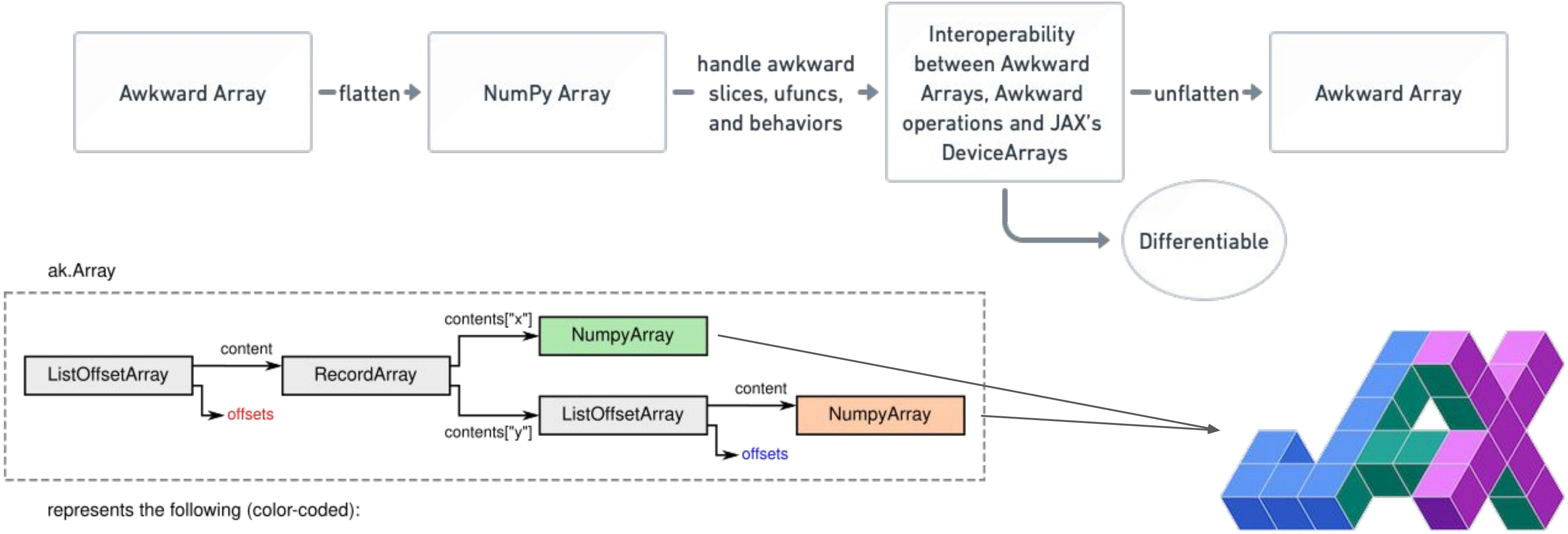
When walking about the countryside of Italy, the people will not hesitate to tell you that **JAX** has *"una anima di pura programmazione funzionale"*.

JAX, or NumPy under steroids

Why JAX?

- JAX has extensive support for NumPy and offers intuitive way to extend its API to custom data containers
- Neos and other autodiff efforts in and around IRIS-HEP already support and use JAX for pure python libraries
- Awkward has a well maintained JAX backend

Awkward and autodiff



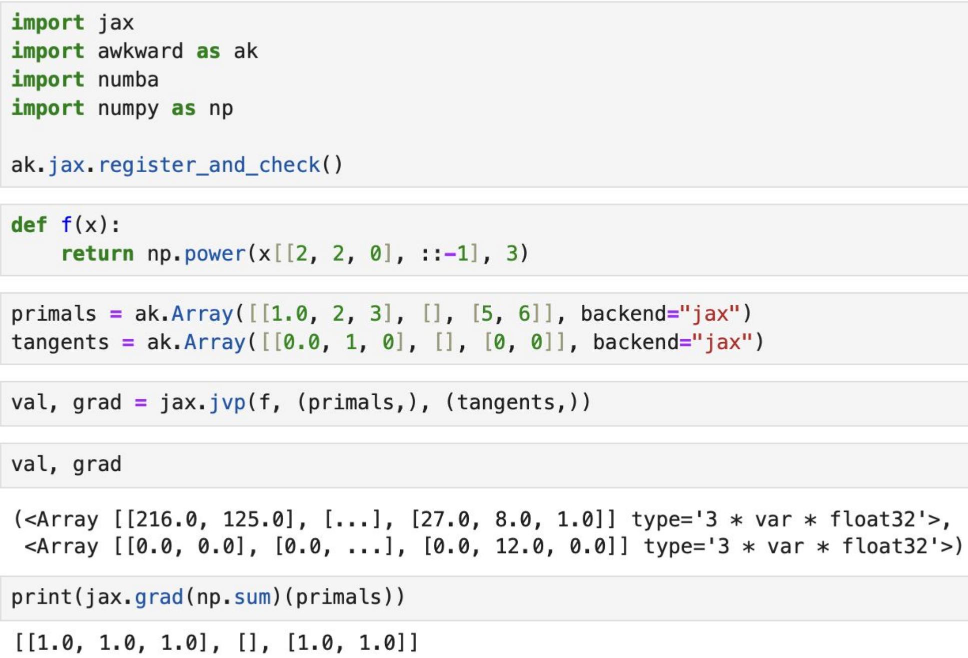
Awkward Array —flatten→ NumPy Array — handle awkward slices, ufuncs, and behaviors → Interoperability between Awkward Arrays, Awkward operations and JAX's DeviceArrays —unflatten→ Awkward Array

Interoperability between Awkward Arrays, Awkward operations and JAX's DeviceArrays → Differentiable

represents the following (color-coded):

```
[
    [{"x": 1.1, "y": [1]}, {"x": 2.2, "y": [1, 2]}, {"x": 3.3, "y": [1, 2, 3]}],
    [],
    [{"x": 4.4, "y": [3, 2]}, {"x": 5.5, "y": [3]}]
]
```

Awkward and autodiff



```
import jax
import awkward as ak
import numba
import numpy as np

ak.jax.register_and_check()
```

```
def f(x):
    return np.power(x[[2, 2, 0], ::-1], 3)
```

```
primals = ak.Array([[1.0, 2, 3], [], [5, 6]], backend="jax")
tangents = ak.Array([[0.0, 1, 0], [], [0, 0]], backend="jax")
```

```
val, grad = jax.jvp(f, (primals,), (tangents,))
```

```
val, grad
```

```
(<Array [[216.0, 125.0], [...], [27.0, 8.0, 1.0]] type='3 * var * float32'>,
 <Array [[0.0, 0.0], [0.0, ...], [0.0, 12.0, 0.0]] type='3 * var * float32'>)
```

```
print(jax.grad(np.sum)(primals))
```

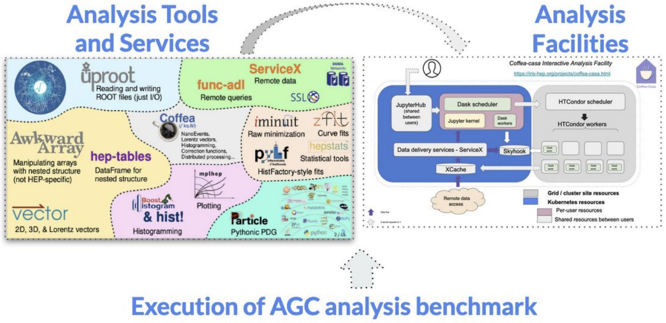
```
[[1.0, 1.0, 1.0], [], [1.0, 1.0]]
```

Analysis Grand Challenge and autodiff

- *The Analysis Grand Challenge (AGC) is about performing the last steps in an analysis pipeline at scale to test workflows envisioned for the HL-LHC.*

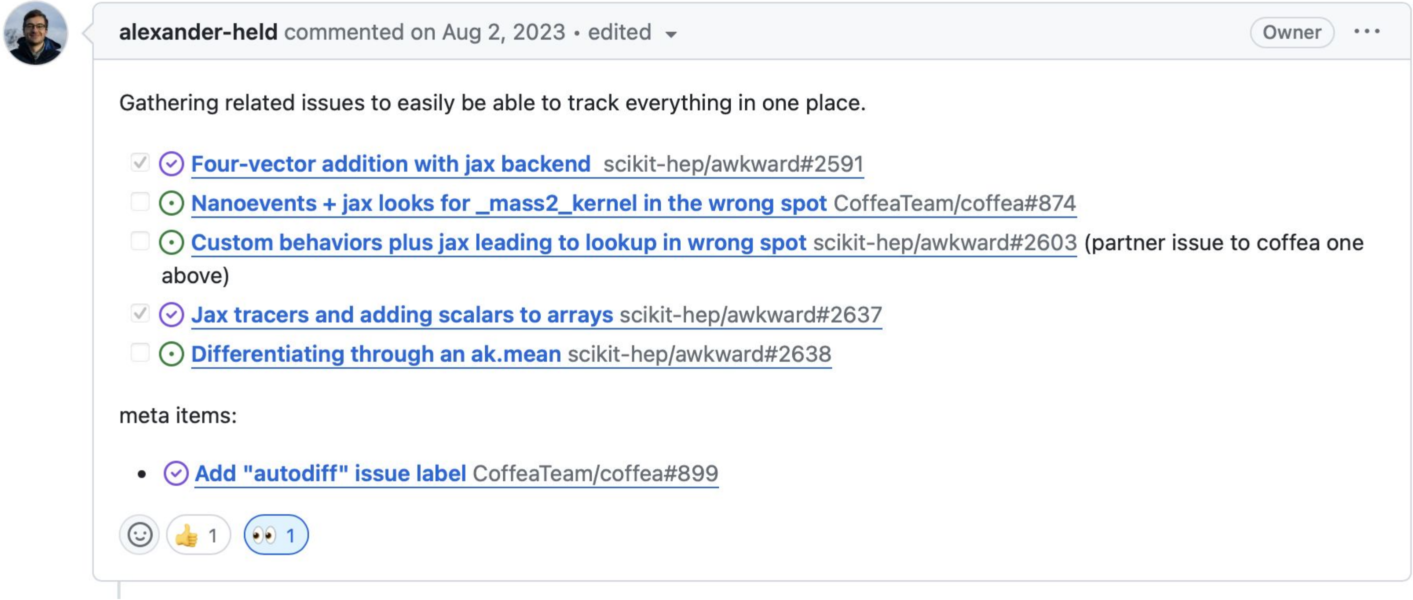
- *The AGC serves as an integration exercise for IRIS-HEP, allowing the testing of new services, libraries and workflows on dedicated analysis facilities in the context of realistic physics analyses.*

- There have been numerous efforts to introduce autodiff to AGC in the past, but the development stalled last year because of several blocker: [alexander-held/agc-autodiff](https://github.com/alexander-held/agc-autodiff).



Analysis Tools and Services
Analysis Facilities
uproot
Reading and writing ROOT files (just I/O)
ServiceX
Remote data
func-adl
Remote queries
DOMA Networks
SSL
Coffea
NanoEvents, Lorentz vectors, Histogramming, Correction functions, Distributed processing...
Awkward Array
Manipulating arrays with nested structure (not HEP-specific)
hep-tables
DataFrame for nested structure
iminuit
Raw minimization
zfit
Curve fits
hepstats
Statistical tools
pyhf
HistFactory-style fits
mplhep
Plotting
Boost histogram & hist!
Histogramming
vector
2D, 3D, & Lorentz vectors
Particle
Pythonic PDG
2/15
Coffea-casa Interactive Analysis Facility
https://iris-hep.org/projects/coffea-casa.html
JupyterHub (shared between users)
Dask scheduler
Jupyter kernel
Dask workers
HTCondor scheduler
HTCondor workers
Data delivery services - ServiceX
Skyhook
XCache
Remote data access
Grid / cluster site resources
Kubernetes resources
Per-user resources
Shared resources between users
Execution of AGC analysis benchmark

Status of autodiff for AGC before my fellowship



alexander-held commented on Aug 2, 2023 • edited

Owner

Gathering related issues to easily be able to track everything in one place.

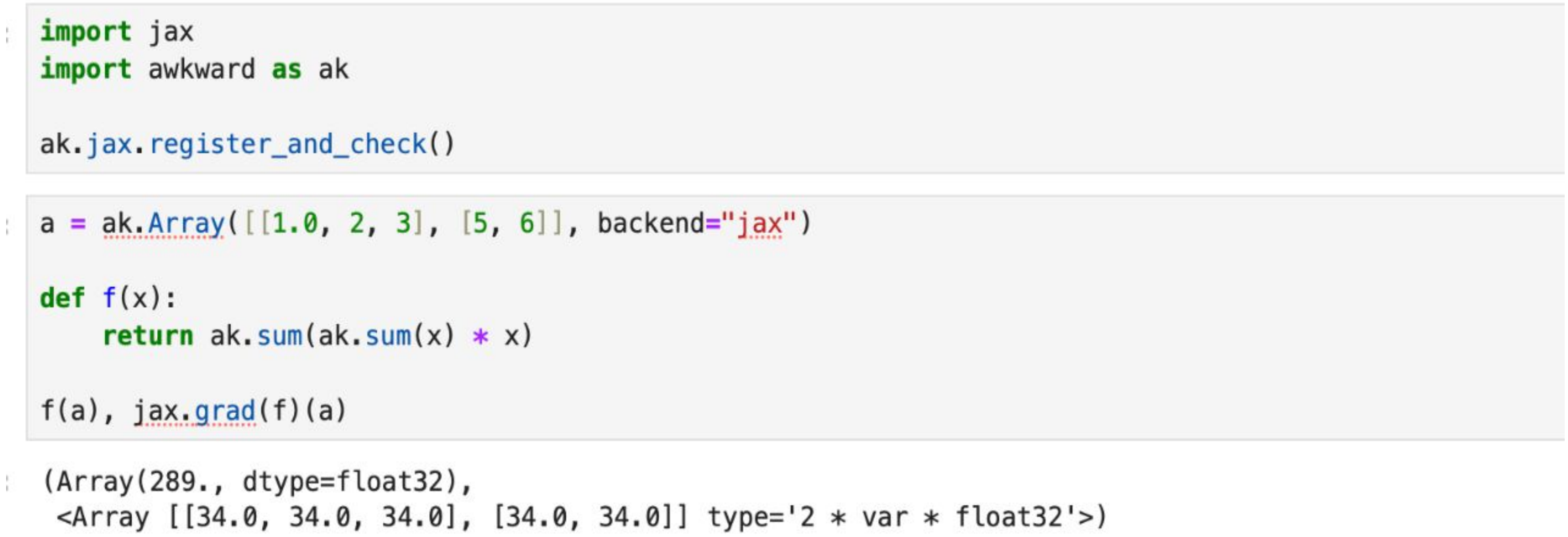
- ☑ Four-vector addition with jax backend scikit-hep/awkward#2591
- □ Nanoevents + jax looks for _mass2_kernel in the wrong spot CoffeaTeam/coffea#874
- □ Custom behaviors plus jax leading to lookup in wrong spot scikit-hep/awkward#2603 (partner issue to coffea one above)
- ☑ Jax tracers and adding scalars to arrays scikit-hep/awkward#2637
- □ Differentiating through an ak.mean scikit-hep/awkward#2638

meta items:

- Add "autodiff" issue label CoffeaTeam/coffea#899

👍 1 👀 1

Current status of autodiff for AGC

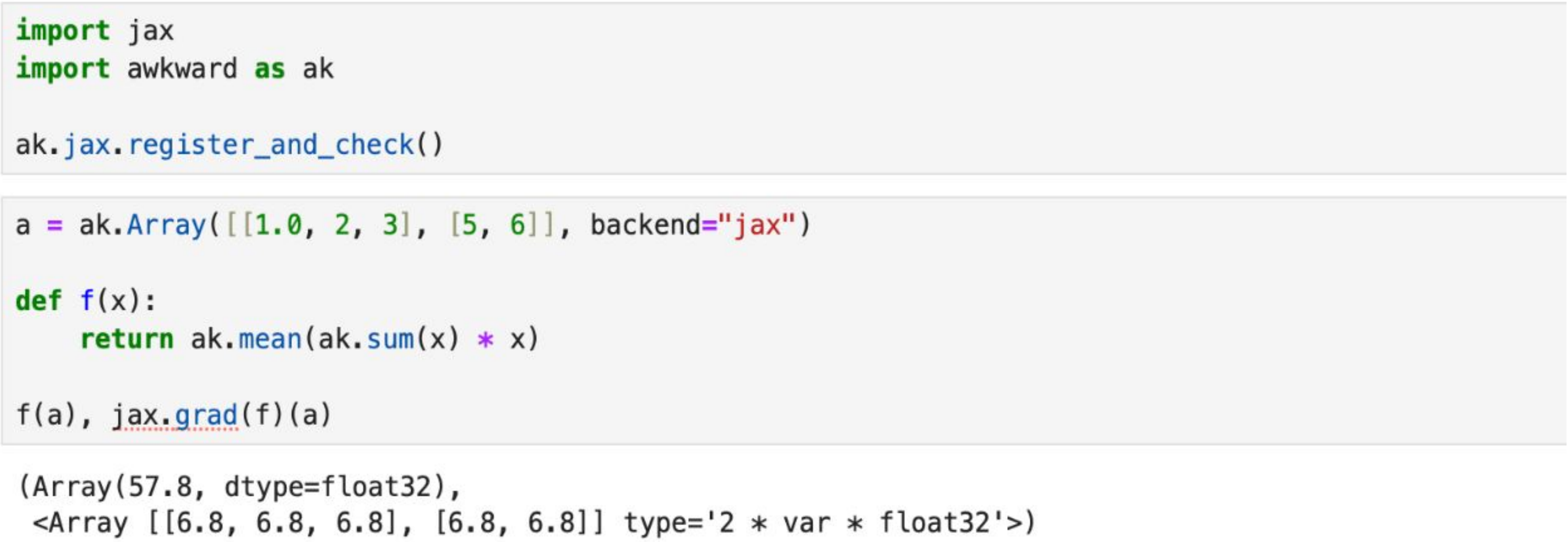


```
import jax
import awkward as ak

ak.jax.register_and_check()
```

```
a = ak.Array([[1.0, 2, 3], [5, 6]], backend="jax")

def f(x):
    return ak.sum(ak.sum(x) * x)

f(a), jax.grad(f)(a)
```

```
(Array(289., dtype=float32),
 <Array [[34.0, 34.0, 34.0], [34.0, 34.0]] type='2 * var * float32'>)
```

Current status of autodiff for AGC

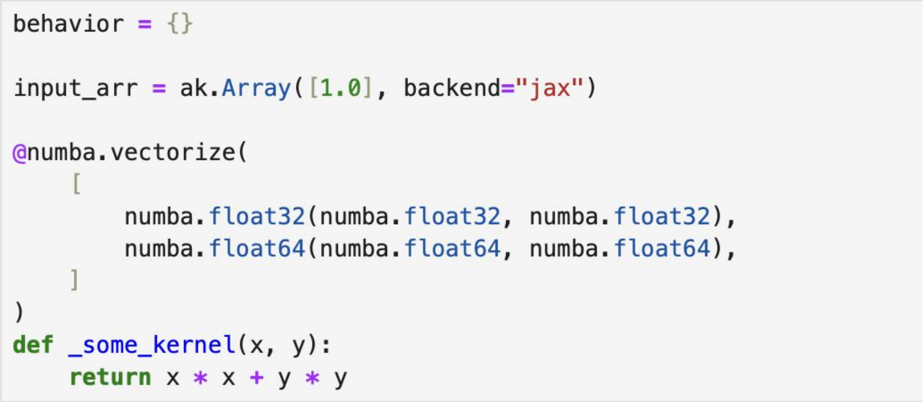


```
import jax
import awkward as ak

ak.jax.register_and_check()
```

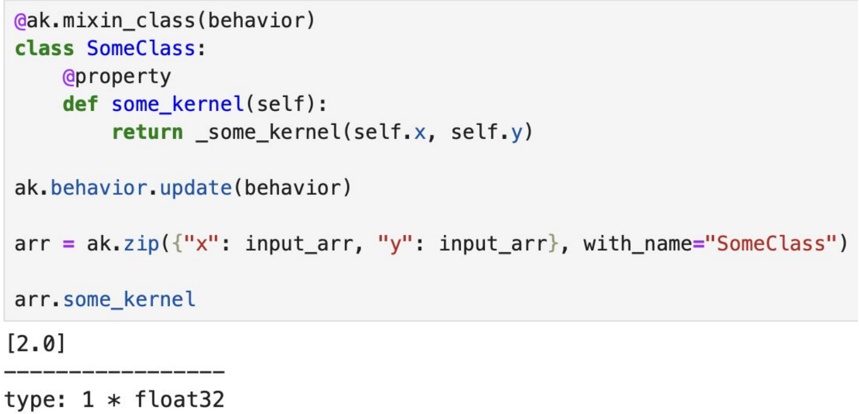
```
a = ak.Array([[1.0, 2, 3], [5, 6]], backend="jax")

def f(x):
    return ak.mean(ak.sum(x) * x)

f(a), jax.grad(f)(a)
```

```
(Array(57.8, dtype=float32),
 <Array [[6.8, 6.8, 6.8], [6.8, 6.8]] type='2 * var * float32'>)
```

Current status of autodiff for AGC



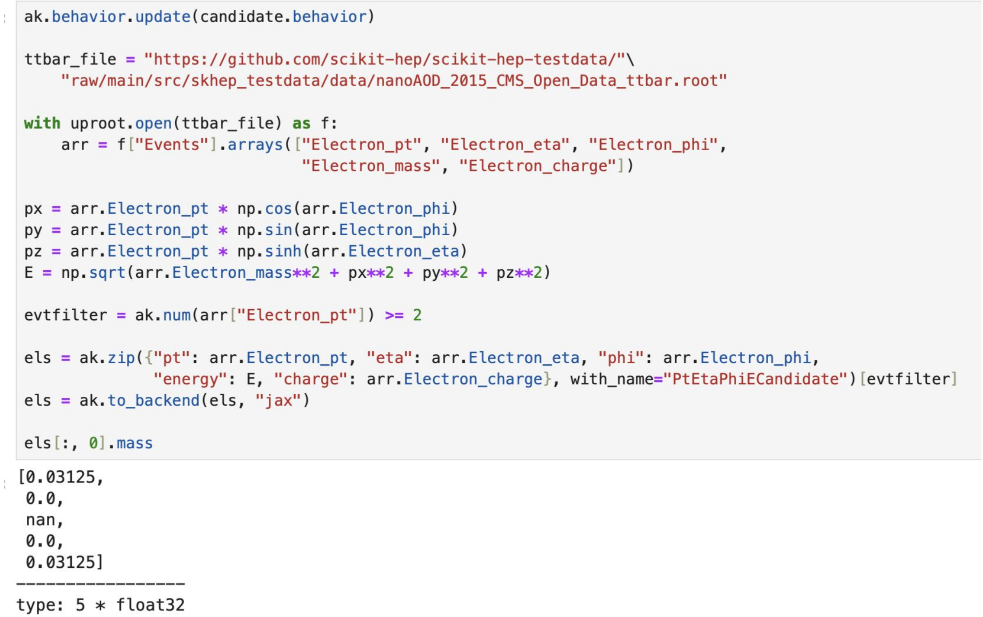
```python
behavior = {}

input_arr = ak.Array([1.0], backend="jax")

@numba.vectorize(
    [
        numba.float32(numba.float32, numba.float32),
        numba.float64(numba.float64, numba.float64),
    ]
)
def _some_kernel(x, y):
    return x * x + y * y
```



```python
@ak.mixin_class(behavior)
class SomeClass:
    @property
    def some_kernel(self):
        return _some_kernel(self.x, self.y)

ak.behavior.update(behavior)

arr = ak.zip({"x": input_arr, "y": input_arr}, with_name="SomeClass")

arr.some_kernel
```

```
[2.0]
-----------------
type: 1 * float32
```

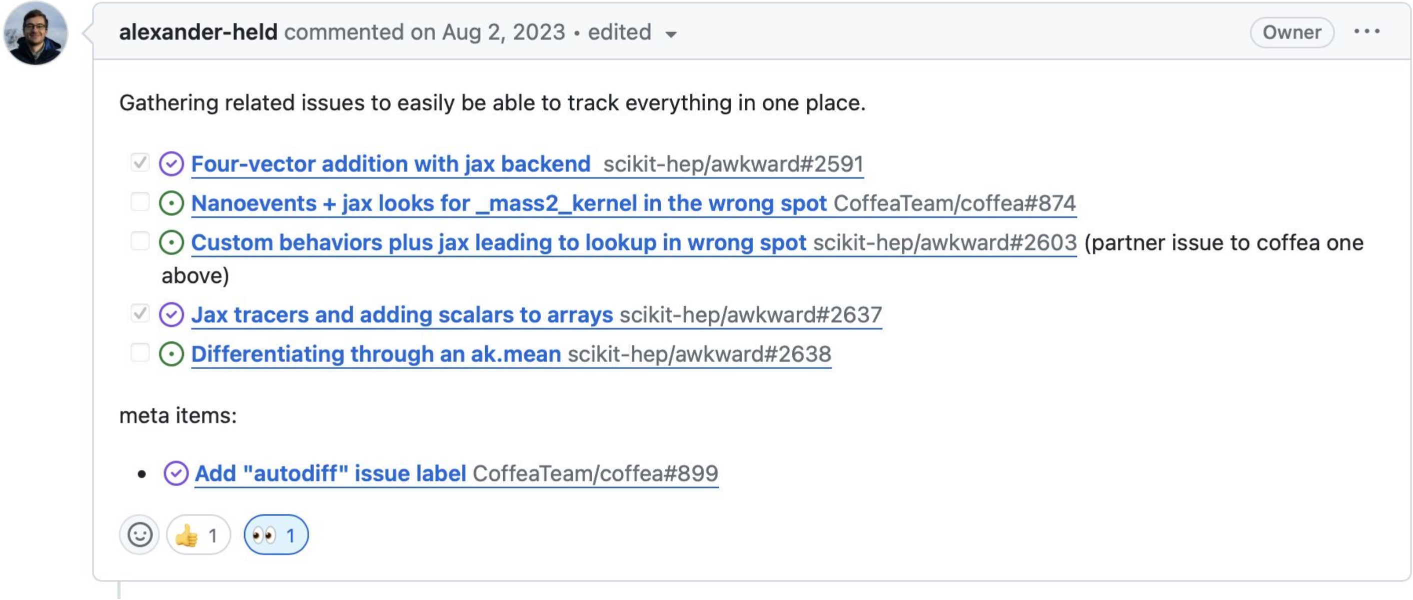
Current status of autodiff for AGC



```
ak.behavior.update(candidate.behavior)

ttbar_file = "https://github.com/scikit-hep/scikit-hep-testdata/"\
    "raw/main/src/skhep_testdata/data/nanoAOD_2015_CMS_Open_Data_ttbar.root"

with uproot.open(ttbar_file) as f:
    arr = f["Events"].arrays(["Electron_pt", "Electron_eta", "Electron_phi",
                              "Electron_mass", "Electron_charge"])

px = arr.Electron_pt * np.cos(arr.Electron_phi)
py = arr.Electron_pt * np.sin(arr.Electron_phi)
pz = arr.Electron_pt * np.sinh(arr.Electron_eta)
E = np.sqrt(arr.Electron_mass**2 + px**2 + py**2 + pz**2)

evtfilter = ak.num(arr["Electron_pt"]) >= 2

els = ak.zip({"pt": arr.Electron_pt, "eta": arr.Electron_eta, "phi": arr.Electron_phi,
              "energy": E, "charge": arr.Electron_charge}, with_name="PtEtaPhiECandidate")[evtfilter]
els = ak.to_backend(els, "jax")

els[:, 0].mass
```

```
[0.03125,
 0.0,
 nan,
 0.0,
 0.03125]
-----------------
type: 5 * float32
```

Status of autodiff for AGC before my fellowship



alexander-held commented on Aug 2, 2023 • edited ▾

Owner ···

Gathering related issues to easily be able to track everything in one place.

- ☑ Four-vector addition with jax backend scikit-hep/awkward#2591
- □ Nanoevents + jax looks for _mass2_kernel in the wrong spot CoffeaTeam/coffea#874
- □ Custom behaviors plus jax leading to lookup in wrong spot scikit-hep/awkward#2603 (partner issue to coffea one above)
- ☑ Jax tracers and adding scalars to arrays scikit-hep/awkward#2637
- □ Differentiating through an ak.mean scikit-hep/awkward#2638

meta items:

- Add "autodiff" issue label CoffeaTeam/coffea#899

Migrating Coffea to Scikit-HEP/vector (spin-off work)

- Coffea (Columnar Object Framework For Effective Analysis) is migrating from their vector module to Scikit-HEP/vector

- My involvement in the efforts -
  - Testing the new implementations (done, users should already be getting a warning when using coffea)
  - Trimming down the vector module and switching to Scikit-HEP/vector as a backend (ongoing, scheduled to go into the March release)
  - Entirely removing coffea’s vector module (tested, will happen in the near future)

- The work on 2 new Scikit-HEP/vector releases was initiated due to issues/discussions that popped up during this migration effort

Coffea

Developments in Scikit-HEP/vector (spin-off work)

- Several issues/design discussions popped up in vector following its adoption on Coffea
- This led to vector’s v1.2 release and a new v1.3 release with more changes will be out soon
- Both the releases change vector to adapt to a physicist’s requirements

Developments in Scikit-HEP/vector (spin-off work) - v1.2

- fix:
  - syncing backends to follow the same promotion/demotion scheme for geometric dimensions (demote to the lowest dimension)
  - returning the correct awkward record when changing dimensions
  - infix operations should not depend on the order of arguments
  - respect user defined awkward mixin subclasses
- docs:
  - better API docs and tutorials
- chore:
  - migrate to ruff
  - migrate to pytest-doctestplus

Developments in Scikit-HEP/vector (spin-off work) - v1.3

- feat:
  - allow momentum coords in `to_Vector*D` methods
  - coordinate transformation functions with momentum names
  - `like` method for projecting vector to the coordinate space of a given vector to mandate strict dimensionality checks (`vector_3d + vector_4d` will now error out but `vector_3d + vector_4d.like(vector_3d)` will work)
- fix:
  - error out on operations on vectors of different geometric dimensions

Developments in Scikit-HEP/boost-histogram (spin-off work)

- feat:
  - support full UHI for rebinning (in progress)

What’s next?

- Collaborating with AGC maintainers and physicists to restart the work on agc-autodiff
- Solving autodiff related blockers whenever they are reported
- A potential SymPy backend for vector (awkward’s jax backend works naturally for vector)
- Helping Coffea with the final migration to Scikit-HEP/vector

Thank you!