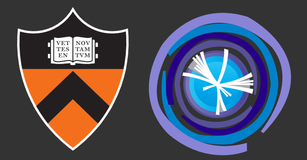
PRINCETON UNIVERSITY

Demo Day
-
Awkward-array: named axis

Peter Fackeldey

10/29/24

2 awkward-array has named axis 🎉



Awkward Array | Choose version | Getting started | User guide | API reference | Contributor guide | Release history | Search ⌘ + K

Section Navigation

- Converting arrays
- Creating new arrays
- Examining arrays
- Array properties
  - Named axes
- Numerical math
- Working with strings
- Filtering data
- Restructuring data
- Combinatorics
- Using arrays in Numba
- Using arrays in C++
- Special topics

User guide > Array properties > Named axes

Named axes

Named axes are a feature in Awkward Array that allows you to give names to the axes of an array. This can be useful for documentation, debugging, and for writing code that is more robust to changes in the structure of the data. As argumented at PyHEP.dev 2023 and by the Harvard NLP group in their "Tensor Considered Harmful" write-up, named axes can be a powerful tool to make code more readable and less error-prone.

Awkward array ensures that named axes are properly propagated to the result. All highlevel, indexing, and broadcasting operations in awkward array support named axes.

Other libraries that support named axes include:

- hist
- haliax
- Tensor Considered Harmful
- PyTorch Named Tensors
- Penzai Named Axis
- xarray Named Axis

On this page

Edit on GitHub

Show Source

PR: #3238
available in upcoming release

Named axis: exemplary benefits

- More readable code (good for supervisors aswell)
- Less errors: wrong axis, wrong broadcastings, …
- Binary ops (e.g. +) of 2 axes with different names will throw an Error (i.e. you shouldn't add 'event' and 'particle' axis)

PyHEP.dev 2023 "Book of the notes"

- Gordon's talk:
  - > If you have to write `axis=1` in an argument to numpy, columnar programming has failed (UX).
  - Started discussion about being able to assign physics meaningful aliases (e.g. `axis="jets"`)
    - c.f. Awkward behaviors
    - c.f. xarray –- beloved by the broader Scientific Python community
      - Example from user guide: https://docs.xarray.dev/en/stable/user-guide/indexing.html#

https://nlp.seas.harvard.edu/NamedTensor.html

Tensor Considered Harmful

Alexander Rush - @harvardnlp

Open in Colab

TL;DR: Despite its ubiquity in deep learning, Tensor is broken. It forces bad habits such as exposing private dimensions, broadcasting based on absolute position, and keeping type information in documentation. This post presents a proof-of-concept of an alternative approach, ***named tensors****, with named dimensions. This change eliminates the need for indexing, dim arguments, einsum- style unpacking, and documentation-based coding. The prototype* ***PyTorch library*** *accompanying this blog post is available as namedtensor.*

Named axis: basics

- Attach names to axes with ‘ak.with_named_axis’ or in ’ak.Array’ constructor:

```
x = ak.Array([[1, 2], [3, 4]], named_axis=("events", "particles"))

x = ak.Array([[1, 2], [3, 4]], named_axis={"events": 0, "particles": 1})

x = ak.with_named_axis(ak.Array([[1, 2], [3, 4]]), named_axis=("events", "particles"))

x = ak.with_named_axis(ak.Array([[1, 2], [3, 4]]), named_axis={"events": 0, "particles": 1})
```

- dict-interface allows to attach names to negative axes swell

- Names to positional axis mapping is visible in the ‘__repr__’ and ‘.show(named_axis=True)’:

```
In [28]: x.show(named_axis=True, type=True)
type: 2 * var * int64
axes: events:0, particles:1
[[1, 2],
 [3, 4]]

In [29]: x
Out[29]: <Array [[1, 2], [3, 4]] events:0,particles:1 type='2 * var * int64'>
```

Named axis: basics

- Named axis can be used in ‘axis=‘ argument of every high-level awkward operation (also to dispatched ones, e.g. ‘np.*’)

```
In [35]: ak.sum(x, axis="events")
Out[35]: <Array [4, 6] particles:0 type='2 * int64'>

In [36]: np.sum(x, axis="events")
Out[36]: <Array [4, 6] particles:0 type='2 * int64'>
```

- Named axis move dynamically with the operations you perform:

  E.g. a reduction removes an axis, and remaining named axis will be adjusted

```
In [30]: x
Out[30]: <Array [[1, 2], [3, 4]] events:0,particles:1 type='2 * var * int64'>

In [31]: ak.sum(x, axis="events")
Out[31]: <Array [4, 6] particles:0 type='2 * int64'>
```

Named axis: new indexing

- Indexing can change the named axes aswell (reducing or adding dimensions)

```
In [37]: x[0]
Out[37]: <Array [1, 2] particles:0 type='2 * int64'>

In [38]: x[1]
Out[38]: <Array [3, 4] events:0 type='2 * int64'>

In [39]: x[None, ...]
Out[39]: <Array [[[1, 2], [3, 4]]] events:1,particles:2 type='1 * 2 * var * int64'>
```

- We can index using the axis names, positional axis, or mixed (with dicts)

```
In [44]: x[{"events": 0}]
Out[44]: <Array [1, 2] particles:0 type='2 * int64'>

In [45]: x[{"events": 1}]
Out[45]: <Array [3, 4] particles:0 type='2 * int64'>

In [46]: x[{"particles": 0}]
Out[46]: <Array [1, 3] events:0 type='2 * int64'>

In [47]: x[{"particles": 1}]
Out[47]: <Array [2, 4] events:0 type='2 * int64'>
```

```
In [55]: x[{0: 0}]
Out[55]: <Array [1, 2] particles:0 type='2 * int64'>

In [56]: x[{0: 1}]
Out[56]: <Array [3, 4] particles:0 type='2 * int64'>

In [57]: x[{"particles": np.s_[0:1]}]
Out[57]: <Array [[1], [3]] events:0,particles:1 type='2 * var * int64'>

In [58]: x[{0: 0, "particles": np.s_[0:1]}]
Out[58]: <Array [1] particles:0 type='1 * int64'>
```

syntax is similar to UHI (Indexing+)

Summary

- Named axis is a new feature in the upcoming awkward release
- Documented at: https://awkward-array.org/doc/main/user-guide/how-to-array-properties-named-axis.html
- Named axis can reduce errors and make code more readable
- It's more than just 'events, particles = 0, 1' - named axes *adjust dynamically* with the array
- These policies (how named axes adjust) are chosen conservatively, mainly inspired by PyTorch's NamedTensor
- New indexing syntax with dicts (similar UHI Indexing +)