

Pythonic HEP ecosystem update

Jim Pivarski

Princeton University – IRIS-HEP

May 23, 2022



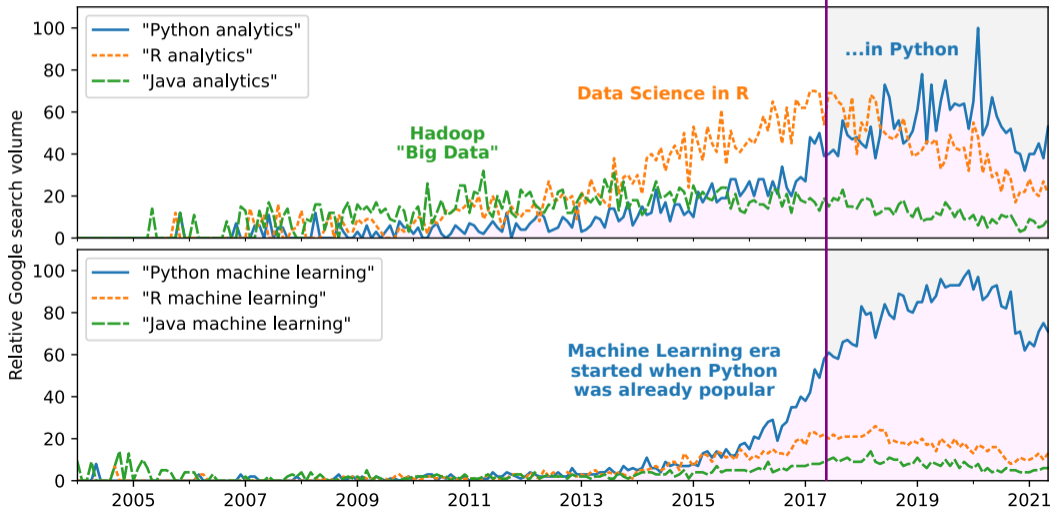
In terms of languages, amid the many languages in use there is consensus on the special roles of some. Performance is key in HEP, and certainly when we talk about reconstruction and performance-critical code, we will continue to look to C++, because the community has invested a lot in this language. It is not the only high performance language, but it is and (in the estimation of the workshop) will remain the dominant one. However, there was a very strong message from the community that for analysis code, python is highly desirable especially because of the fast development cycle and in general people can achieve the same results with fewer lines of code. **Python should be treated as a first class language.** Performance-critical code should be offloaded to libraries, probably written in C++, but the analysis code can be python. Python is also the language of choice in the wider scientific data analytics community. Sustaining PyROOT in particular is a key concern of the community. There is an argument that we lose C++ experience in the community by encouraging python but the overall view was that freeing up analysts time by enabling faster analysis turnaround is the best way to free up time for important work outside of analysis. An important point for language choices is that we need to ensure that senior people also need to be included, requiring adequate education and documentation. End users should be exposed to simple interfaces, which should reduce the complexity of analysis code.

Back then, Python was just beginning to take over data analytics



Source: Google Analytics, worldwide search traffic.

Analysis Ecosystem I

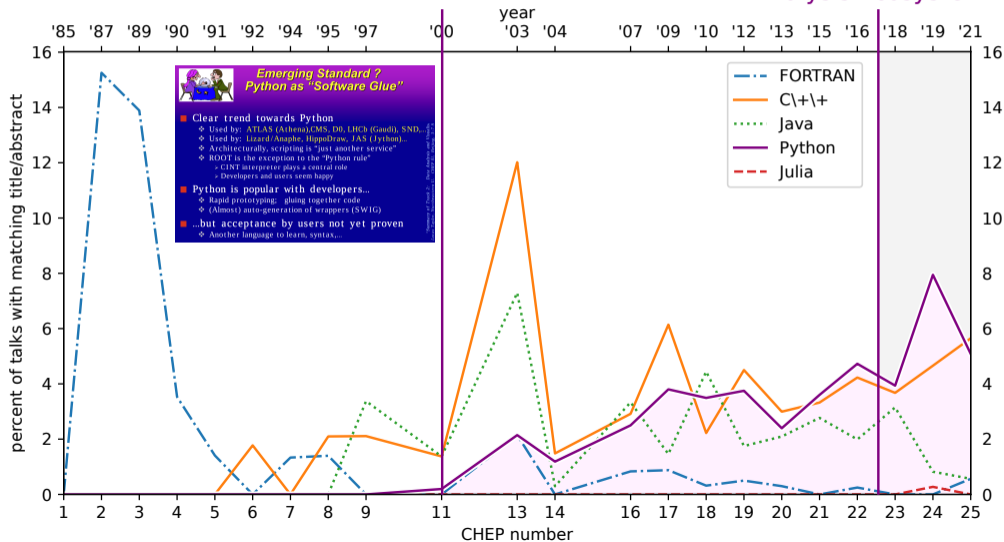


Though it had been slowly growing in HEP for years



Source: Title and abstract matches in CHEP proceedings.

Analysis Ecosystem I

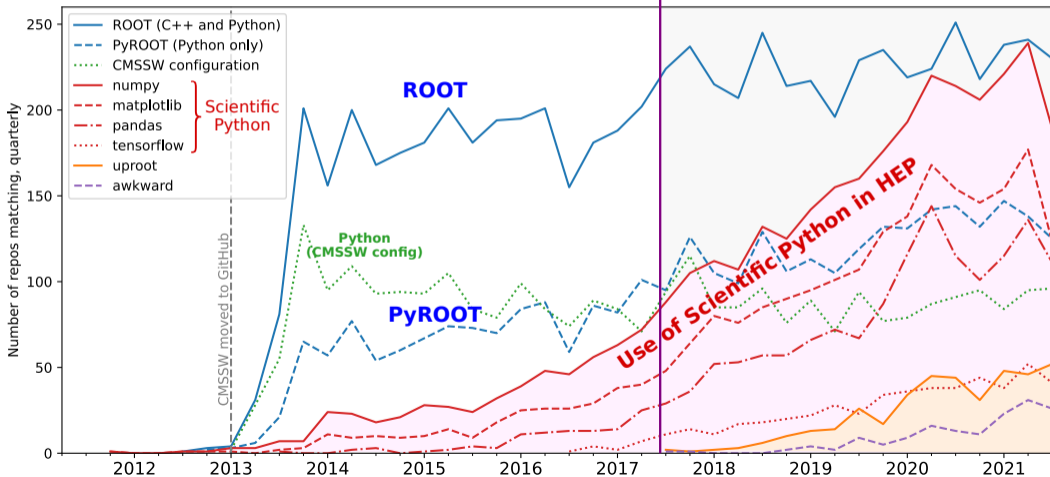


But the use of Scientific Python (NumPy, etc.) was new to HEP



Source: `import XYZ` matches in GitHub repos for users who fork CMSSW.

Analysis Ecosystem I

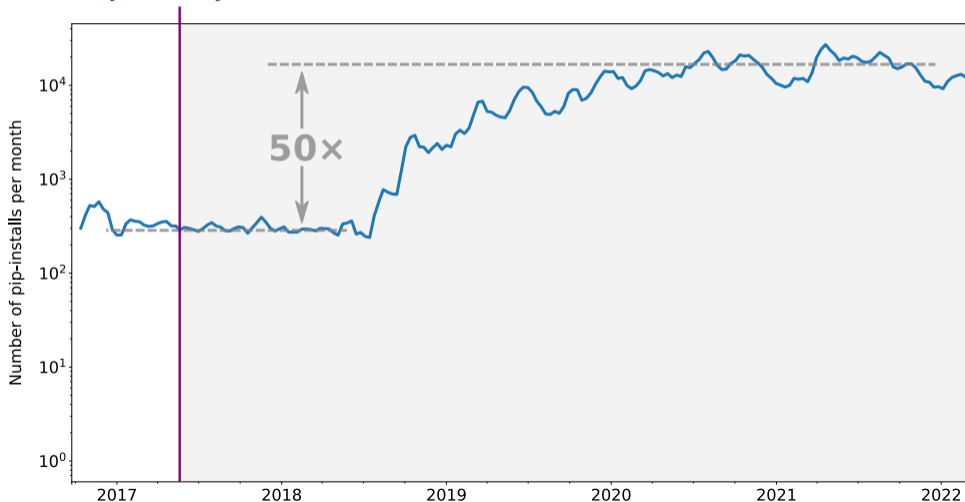


Scikit-HEP had not yet taken off



Source: "pip install XYZ" download rate for MacOS/Windows (no batch jobs).

Analysis Ecosystem I

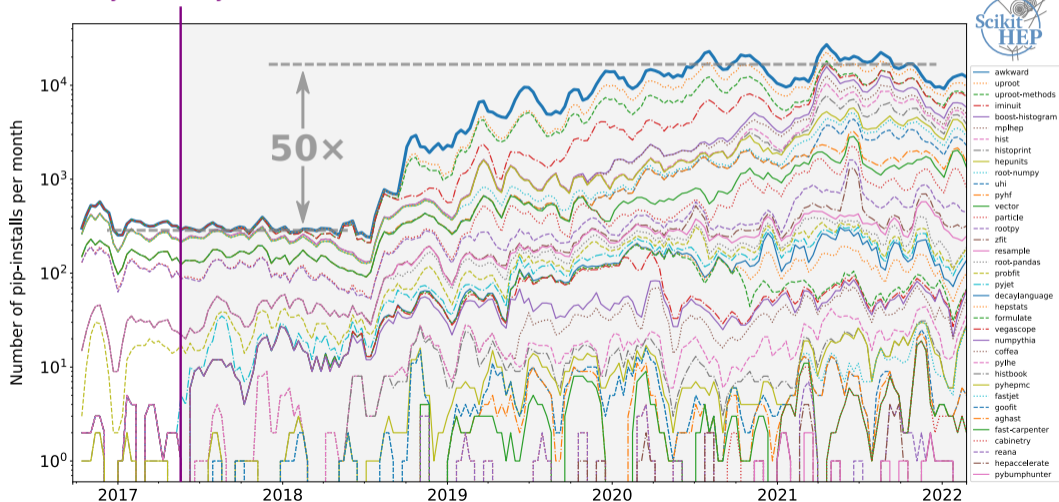


Scikit-HEP had not yet taken off



Source: "pip install XYZ" download rate for MacOS/Windows (no batch jobs).

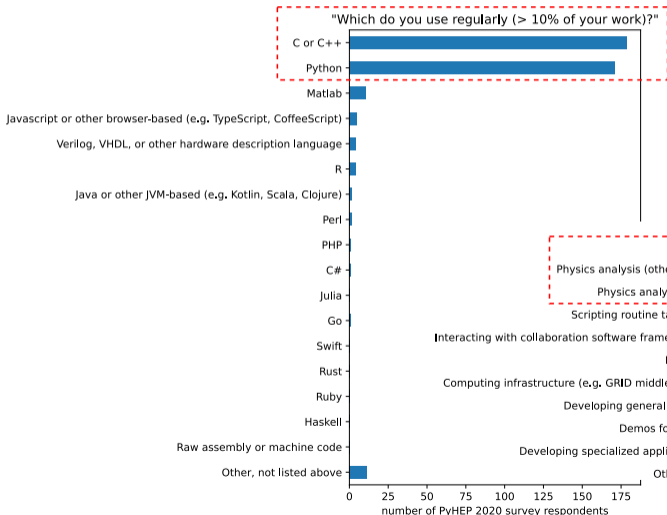
Analysis Ecosystem I



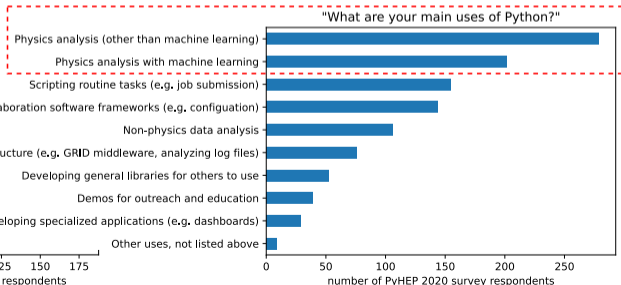
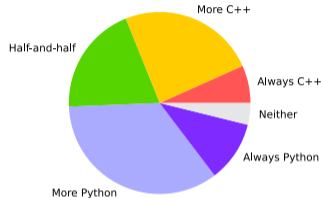
We are now a two-language community



Source: PyHEP 2020 workshop survey.



"How often do you use Python relative to C or C++?"



The Pythonic HEP ecosystem now





The Scientific Python world lacked HEP-style histograms; it's one of the things we have to make ourselves.



The Scientific Python world lacked HEP-style histograms; it's one of the things we have to make ourselves.

Also, it seems easy: just bin and count, right?



The Scientific Python world lacked HEP-style histograms; it's one of the things we have to make ourselves.

Also, it seems easy: just bin and count, right?

Physicists have created at least 20 histogram libraries in Python, most single-author.

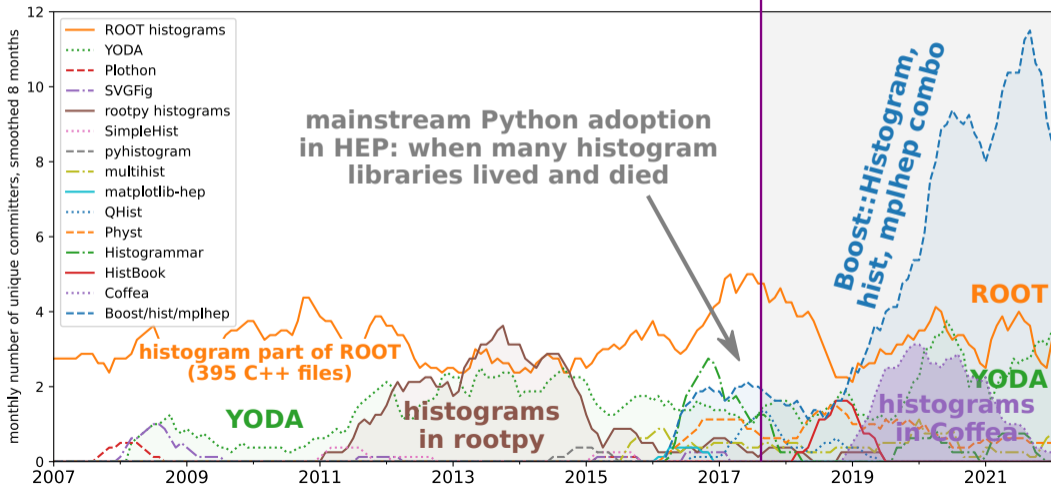
- | | | | |
|----------------------|--------------------------|---------------------------|------------------------------|
| ▶ PyROOT (2004–now) | ▶ DANSE (2009–2011) | ▶ matplotlib-hep (2016) | ▶ Coffea.hist (2019–2022) |
| ▶ PAIDA (2004–2007) | ▶ rootpy (2011–2019) | ▶ QHist (2017–2019) | ▶ boost-histogram (2019–now) |
| ▶ Plotho (2007–2008) | ▶ SimpleHist (2011–2015) | ▶ Physt (2016–now) | ▶ mplhep (2019–now) |
| ▶ SVGFig (2008–2009) | ▶ pyhistogram (2015) | ▶ Histogrammar (2016–now) | ▶ histoprint (2020–now) |
| ▶ YODA (2008–now) | ▶ multihist (2015–now) | ▶ HistBook (2018–2019) | ▶ hist (2020–now) |

Histogram proliferation and convergence



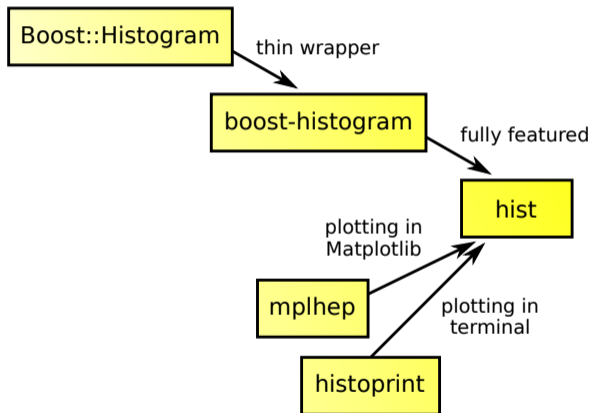
Number of unique developers contributing to each library per month (in git).

Analysis Ecosystem I



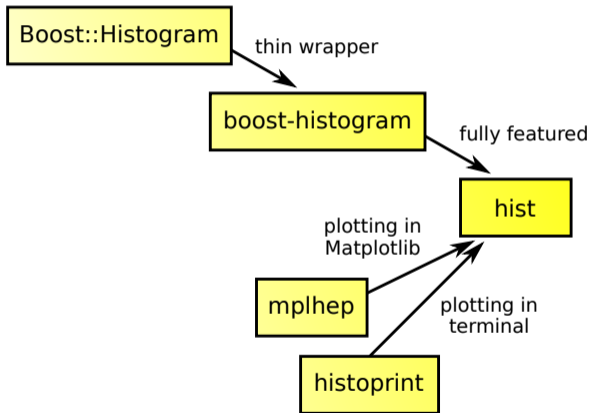


Why combine Boost::Histogram, hist, mplhep?



Originally, each of these was developed independently by a single author.

Why combine Boost::Histogram, hist, mplhep?

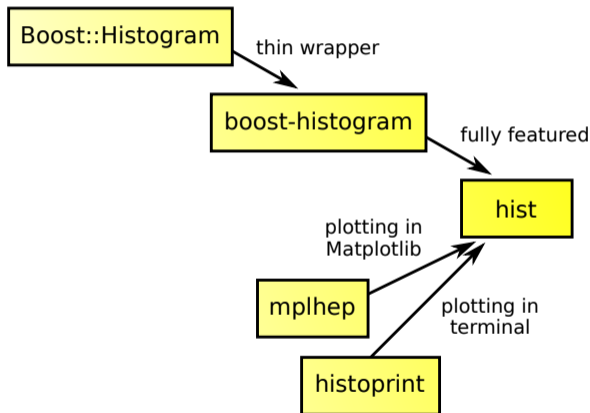


Originally, each of these was developed independently by a single author.

They each provide a piece of functionality users can get through

```
import hist
```

Why combine Boost::Histogram, hist, mplhep?



Originally, each of these was developed independently by a single author.

They each provide a piece of functionality users can get through

```
import hist
```

Now, 47 developers have contributed to these packages, and 20 contributed to more than one.



uhi 0.3.1 documentation

UHI: Unified Histogram Interface

CONTENTS:

Indexing

Indexing+

Plotting



Help for plotters

The module `uhi.numpy_plottable` has a utility to simplify the common use case of accepting a `PlottableProtocol` or other common formats, primarily a NumPy `histogram/histogram2d/histogramdd` tuple. The `ensure_plottable_histogram` function will take a histogram or NumPy tuple, or an object that implements `.to_numpy()` or `.numpy()` and convert it to a `NumPyPlottableHistogram`, which is a minimal implementation of the Protocol. By calling this function on your input, you can then write your plotting function knowing that you always have a `PlottableProtocol` object, greatly simplifying your code.

The full protocol version 1.2 follows:

(Also available as `uhi.typing.plottable.PlottableProtocol`, for use in tests, etc.

```
"""
Using the protocol:

Producers: use isinstance(myhist, PlottableHistogram) in your tests; part of
the protocol is checkable at runtime, though ideally you should use MyPy; if
your histogram class supports PlottableHistogram, this will pass.

Consumers: Make your functions accept the PlottableHistogram static type, and
MyPy will force you to only use items in the Protocol.
"""
```

☰ Contents

Using the protocol:

Implementing the protocol:

Help for plotters

The full protocol version 1.2 follows:



uproot-browser - uproot-Event.root 01:38:27

- uproot-Event.root
 - ? <unnamed> TProcessID
 - * T (1000)
 - hstat TH1F (100)
 - htime TH1F (10)

Scikit-HEP
UPROOT⁴-BROWSER
Powered by Textual & Hist

B Toggle sidebar Q Quit



uproot-browser - uproot-Event.root 01:38:27

- uproot-Event.root
 - ? <unnamed> TProcessID
 - * T (1000)
 - hstat TH1F (100)
 - htime TH1F (10)

Scikit-HEP
UPROOT⁴-BROWSER
Powered by Textual & Hist

B Toggle sidebar Q Quit

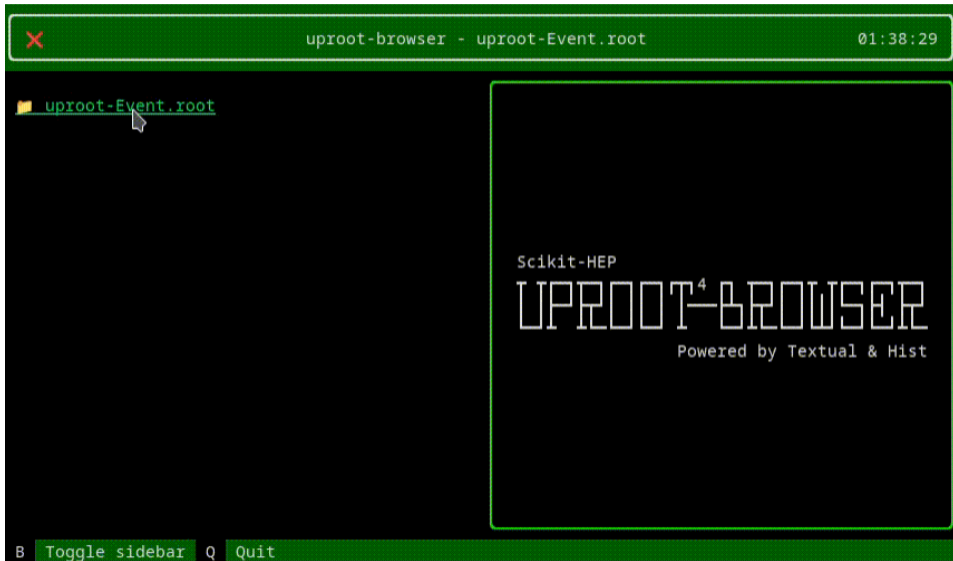


uproot-browser - uproot-Event.root 01:38:29

- uproot-Event.root
 - ? <unnamed> TProcessID
 - * T (1000)
 - hstat TH1F (100)
 - htime TH1F (10)

Scikit-HEP
UPROOT⁴-BROWSER
Powered by Textual & Hist

B Toggle sidebar Q Quit





uproot-browser - uproot-Event.root 01:38:30

- uproot-Event.root
 - ? <unnamed> TProcessID
 - * T (1000)
 - hstat TH1F (100)
 - htime TH1F (10)

Scikit-HEP
UPROOT⁴-BROWSER
Powered by Textual & Hist

B Toggle sidebar Q Quit



The screenshot shows the uproot-browser application window titled "uproot-browser - uproot-Event.root" with a timer at "01:38:30". The interface is dark-themed and features a sidebar on the left displaying a tree view of the event structure:

- uproot-Event.root
 - ? <unnamed> TProcessID
 - * T (1000)
 - hstat TH1F (100)
 - htime TH1F (10)

The main area displays the Scikit-HEP logo and the text "UPROOT⁴-BROWSER" in a monospace font, with "Powered by Textual & Hist" below it. At the bottom, a green bar contains the keyboard shortcuts "B Toggle sidebar" and "Q Quit".

uproot-browser: another interface package that pulls many together

The screenshot shows the uproot-browser application window. The title bar is green and contains the text "uproot-browser - uproot-Event.root" and a timer "01:38:31". The main content area is split into two panes. The left pane shows a tree view of the loaded data structure. The root is "uproot-Event.root", which contains a "TProcessID" object and a "T(1000)" object. The "T(1000)" object is expanded to show its members: "event" (Event), "TObject" (group of fUniqueID:uint32_t, fBits:uint8_t), "fBits" (uint8_t), "fUniqueID" (uint32_t), "fClosestDistance" (unknown[]), "fEventName" (char*), "fEvtHdr" (EventHeader), "fEvtHdr.fDate" (int32_t), "fEvtHdr.fEvtNum" (int32_t), "fEvtHdr.fRun" (int32_t), "fFlag" (uint32_t), "fH" (TH1F), "fHighPt" (TRefArray*), "fIsValid" (bool), "fLastTrack" (TRef), and "fMatrix[4][4]" (float[4][4]). The right pane displays a logo for "Scikit-HEP UPROOT⁴-BROWSER" with the text "Powered by Textual & Hist" below it. At the bottom of the window, there is a green status bar with the text "B Toggle sidebar Q Quit".



```
uproot-browser - uproot-Event.root 01:38:31
```

```
fUniqueID uint32_t
fClosestDistance unknown[]
fEventName char*
fEvtHdr EventHeader
fEvtHdr.fDate int32_t
fEvtHdr.fEvtNum int32_t
fEvtHdr.fRun int32_t
fFlag uint32_t
fH TH1F
fHighPt TRefArray*
fIsValid bool
fLastTrack TRef
fMatrix[4][4] float[4][4]
fMeasures[10] int32_t[10]
fMuons TRefArray*
fNseg int32_t
fNtrack int32_t
fNvertex uint32_t
fTemperature float
fTracks TClonesArray*
fTracks.fBits uint8_t[]
```

```
Scikit-HEP
UPROOT4-BROWSER
Powered by Textual & Hist
```

```
B Toggle sidebar Q Quit
```

uproot-browser: another interface package that pulls many together



```
uproot-browser - uproot-Event.root 01:38:32
```

```

  * fFlag uint32_t
  * fH TH1F
  * fHighPt TRefArray*
  * fIsValid bool
  * fLastTrack TRef
  * fMatrix[4][4] float[4][4]
  * fMeasures[10] int32_t[10]
  * fMuons TRefArray*
  * fNseg int32_t
  * fNtrack int32_t
  * fNvertex uint32_t
  * fTemperature float
  * fTracks TClonesArray*
  * fTracks.fBits uint8_t[]
  * fTracks.fBx Float16_t[]
  * fTracks.fBy Float16_t[]
  * fTracks.fCharge Double32_t[]
  * fTracks.fMass2 Float16_t[]
  * fTracks.fMeanCharge float[]
  * fTracks.fNpoint int32_t[]
  * fTracks.fNsp uint32_t[]

```

```

Scikit-HEP
UPROOT4-BROWSER
Powered by Textual & Hist

```

```

B Toggle sidebar Q Quit

```

uproot-browser: another interface package that pulls many together



```
uproot-browser - uproot-Event.root 01:38:32
```

```
fMuons TRefArray*
fNseg int32_t
fNtrack int32_t
fNvertex uint32_t
fTemperature float
fTracks TClonesArray*
fTracks.fBits uint8_t[]
fTracks.fBx Float16_t[]
fTracks.fBy Float16_t[]
fTracks.fCharge Double32_t[]
fTracks.fMass2 Float16_t[]
fTracks.fMeanCharge float[]
fTracks.fNpoint int32_t[]
fTracks.fNsp uint32_t[]
fTracks.fPointValue unknown[][]
fTracks.fPx float[]
fTracks.fPy float[]
fTracks.fPz float[]
fTracks.fRandom float[]
fTracks.fTArray[3] float[][3]
fTracks.fTriggerBits.fAllBits
```

```
Scikit-HEP
UPROOT4-BROWSER
Powered by Textual & Hist
```

B Toggle sidebar Q Quit

uproot-browser: another interface package that pulls many together



uproot-browser - uproot-Event.root 01:38:34

- fTracks.fBy Float16_t[]
- fTracks.fCharge Double32_t[]
- fTracks.fMass2 Float16_t[]
- fTracks.fMeanCharge float[]
- fTracks.fNpoint int32_t[]
- fTracks.fNsp uint32_t[]
- fTracks.fPointValue unknown[][]
- fTracks.fPx float[]
- fTracks.fPy float[]
- fTracks.fPz float[]
- fTracks.fRandom float[]
- fTracks.fTArray[3] float[][3]
- fTracks.fTriggerBits.fAllBits uint8_t[][]
- fTracks.fTriggerBits.fBits uint8_t[]
- fTracks.fTriggerBits.fNbits uint32_t[]
- fTracks.fTriggerBits.fNbytes uint32_t[]
- fTracks.fTriggerBits.fUniqueID

Scikit-HEP
UPROOT⁴-BROWSER
Powered by Textual & Hist

B Toggle sidebar Q Quit

uroot-browser: another interface package that pulls many together



uroot-browser - uroot-Event.root 01:38:34

```
fTracks.fBy Float16_t[]
fTracks.fCharge Double32_t[]
fTracks.fMass2 Float16_t[]
fTracks.fMeanCharge float[]
fTracks.fNpoint int32_t[]
fTracks.fNsp uint32_t[]
fTracks.fPointValue unknown[][]
fTracks.fPx float[]
fTracks.fPy float[]
fTracks.fPz float[]
fTracks.fRandom float[]
fTracks.fTArray[3] float[][3]
fTracks.fTriggerBits.fAllBits
uint8_t[][]
fTracks.fTriggerBits.fBits
uint8_t[]
fTracks.fTriggerBits.fNbits
uint32_t[]
fTracks.fTriggerBits.fNbytes
uint32_t[]
fTracks.fTriggerBits.fUniqueID
```

B Toggle sidebar Q Quit

fTracks.fPx -- Entries: 599392

Bin Range	Count
-4.6 to -4.5	3.0
-4.5 to -4.4	3.0
-4.4 to -4.3	3.0
-4.3 to -4.2	3.0
-4.2 to -4.1	3.0
-4.1 to -4.0	3.0
-4.0 to -3.9	3.0
-3.9 to -3.8	3.0
-3.8 to -3.7	3.0
-3.7 to -3.6	3.0
-3.6 to -3.5	3.0
-3.5 to -3.4	3.0
-3.4 to -3.3	3.0
-3.3 to -3.2	3.0
-3.2 to -3.1	3.0
-3.1 to -3.0	3.0
-3.0 to -2.9	3.0
-2.9 to -2.8	3.0
-2.8 to -2.7	3.0
-2.7 to -2.6	3.0
-2.6 to -2.5	3.0
-2.5 to -2.4	3.0
-2.4 to -2.3	3.0
-2.3 to -2.2	3.0
-2.2 to -2.1	3.0
-2.1 to -2.0	3.0
-2.0 to -1.9	3.0
-1.9 to -1.8	3.0
-1.8 to -1.7	3.0
-1.7 to -1.6	3.0
-1.6 to -1.5	3.0
-1.5 to -1.4	3.0
-1.4 to -1.3	3.0
-1.3 to -1.2	3.0
-1.2 to -1.1	3.0
-1.1 to -1.0	3.0
-1.0 to -0.9	3.0
-0.9 to -0.8	3.0
-0.8 to -0.7	3.0
-0.7 to -0.6	3.0
-0.6 to -0.5	3.0
-0.5 to -0.4	3.0
-0.4 to -0.3	3.0
-0.3 to -0.2	3.0
-0.2 to -0.1	3.0
-0.1 to 0.0	3.0
0.0 to 0.1	3.0
0.1 to 0.2	3.0
0.2 to 0.3	3.0
0.3 to 0.4	3.0
0.4 to 0.5	3.0
0.5 to 0.6	3.0
0.6 to 0.7	3.0
0.7 to 0.8	3.0
0.8 to 0.9	3.0
0.9 to 1.0	3.0
1.0 to 1.1	3.0
1.1 to 1.2	3.0
1.2 to 1.3	3.0
1.3 to 1.4	3.0
1.4 to 1.5	3.0
1.5 to 1.6	3.0
1.6 to 1.7	3.0
1.7 to 1.8	3.0
1.8 to 1.9	3.0
1.9 to 2.0	3.0
2.0 to 2.1	3.0
2.1 to 2.2	3.0
2.2 to 2.3	3.0
2.3 to 2.4	3.0
2.4 to 2.5	3.0
2.5 to 2.6	3.0
2.6 to 2.7	3.0
2.7 to 2.8	3.0
2.8 to 2.9	3.0
2.9 to 3.0	3.0
3.0 to 3.1	3.0
3.1 to 3.2	3.0
3.2 to 3.3	3.0
3.3 to 3.4	3.0
3.4 to 3.5	3.0
3.5 to 3.6	3.0
3.6 to 3.7	3.0
3.7 to 3.8	3.0
3.8 to 3.9	3.0
3.9 to 4.0	3.0
4.0 to 4.1	3.0
4.1 to 4.2	3.0
4.2 to 4.3	3.0
4.3 to 4.4	3.0
4.4 to 4.5	3.0

uroot-browser: another interface package that pulls many together



uroot-browser - uroot-Event.root 01:38:35

```
fTracks.fBy Float16_t[]
fTracks.fCharge Double32_t[]
fTracks.fMass2 Float16_t[]
fTracks.fMeanCharge float[]
fTracks.fNpoint int32_t[]
fTracks.fNsp uint32_t[]
fTracks.fPointValue unknown[][]
fTracks.fPx float[]
fTracks.fPy float[]
fTracks.fPz float[]
fTracks.fRandom float[]
fTracks.fTArray[3] float[][3]
fTracks.fTriggerBits.fAllBits
uint8_t[][]
fTracks.fTriggerBits.fBits
uint8_t[]
fTracks.fTriggerBits.fNbits
uint32_t[]
fTracks.fTriggerBits.fNbytes
uint32_t[]
fTracks.fTriggerBits.fUniqueID
```

B Toggle sidebar Q Quit

fTracks.fPx -- Entries: 599392

Bin Range	Count
-4.6 to -4.5	3.0
-4.5 to -4.4	3.0
-4.4 to -4.3	3.0
-4.3 to -4.2	3.0
-4.2 to -4.1	3.0
-4.1 to -4.0	3.0
-4.0 to -3.9	3.0
-3.9 to -3.8	3.0
-3.8 to -3.7	3.0
-3.7 to -3.6	3.0
-3.6 to -3.5	3.0
-3.5 to -3.4	3.0
-3.4 to -3.3	3.0
-3.3 to -3.2	3.0
-3.2 to -3.1	3.0
-3.1 to -3.0	3.0
-3.0 to -2.9	3.0
-2.9 to -2.8	3.0
-2.8 to -2.7	3.0
-2.7 to -2.6	3.0
-2.6 to -2.5	3.0
-2.5 to -2.4	3.0
-2.4 to -2.3	3.0
-2.3 to -2.2	3.0
-2.2 to -2.1	3.0
-2.1 to -2.0	3.0
-2.0 to -1.9	3.0
-1.9 to -1.8	3.0
-1.8 to -1.7	3.0
-1.7 to -1.6	3.0
-1.6 to -1.5	3.0
-1.5 to -1.4	3.0
-1.4 to -1.3	3.0
-1.3 to -1.2	3.0
-1.2 to -1.1	3.0
-1.1 to -1.0	3.0
-1.0 to -0.9	3.0
-0.9 to -0.8	3.0
-0.8 to -0.7	3.0
-0.7 to -0.6	3.0
-0.6 to -0.5	3.0
-0.5 to -0.4	3.0
-0.4 to -0.3	3.0
-0.3 to -0.2	3.0
-0.2 to -0.1	3.0
-0.1 to 0.0	3.0
0.0 to 0.1	3.0
0.1 to 0.2	3.0
0.2 to 0.3	3.0
0.3 to 0.4	3.0
0.4 to 0.5	3.0
0.5 to 0.6	3.0
0.6 to 0.7	3.0
0.7 to 0.8	3.0
0.8 to 0.9	3.0
0.9 to 1.0	3.0
1.0 to 1.1	3.0
1.1 to 1.2	3.0
1.2 to 1.3	3.0
1.3 to 1.4	3.0
1.4 to 1.5	3.0
1.5 to 1.6	3.0
1.6 to 1.7	3.0
1.7 to 1.8	3.0
1.8 to 1.9	3.0
1.9 to 2.0	3.0
2.0 to 2.1	3.0
2.1 to 2.2	3.0
2.2 to 2.3	3.0
2.3 to 2.4	3.0
2.4 to 2.5	3.0
2.5 to 2.6	3.0
2.6 to 2.7	3.0
2.7 to 2.8	3.0
2.8 to 2.9	3.0
2.9 to 3.0	3.0
3.0 to 3.1	3.0
3.1 to 3.2	3.0
3.2 to 3.3	3.0
3.3 to 3.4	3.0
3.4 to 3.5	3.0
3.5 to 3.6	3.0
3.6 to 3.7	3.0
3.7 to 3.8	3.0
3.8 to 3.9	3.0
3.9 to 4.0	3.0
4.0 to 4.1	3.0
4.1 to 4.2	3.0
4.2 to 4.3	3.0
4.3 to 4.4	3.0
4.4 to 4.5	3.0

uroot-browser: another interface package that pulls many together



uroot-browser - uroot-Event.root 01:38:35

```
fTracks.fBy Float16_t[]
fTracks.fCharge Double32_t[]
fTracks.fMass2 Float16_t[]
fTracks.fMeanCharge float[]
fTracks.fNpoint int32_t[]
fTracks.fNsp uint32_t[]
fTracks.fPointValue unknown[][]
fTracks.fPx float[]
fTracks.fPy float[]
fTracks.fPz float[]
fTracks.fRandom float[]
fTracks.fTArray[3] float[][3]
fTracks.fTriggerBits.fAllBits
uint8_t[][]
fTracks.fTriggerBits.fBits
uint8_t[]
fTracks.fTriggerBits.fNbits
uint32_t[]
fTracks.fTriggerBits.fNbytes
uint32_t[]
fTracks.fTriggerBits.fUniqueID
```

B Toggle sidebar Q Quit

fTracks.fPy -- Entries: 599392

Bin Range	Count
-5.2 to -4.9	~100
-4.9 to -4.6	~100
-4.6 to -4.3	~100
-4.3 to -4.0	~100
-4.0 to -3.7	~100
-3.7 to -3.4	~100
-3.4 to -3.1	~100
-3.1 to -2.8	~100
-2.8 to -2.5	~100
-2.5 to -2.2	~100
-2.2 to -1.9	~100
-1.9 to -1.6	~100
-1.6 to -1.3	~100
-1.3 to -1.0	~100
-1.0 to -0.7	~100
-0.7 to -0.4	~100
-0.4 to -0.1	~100
-0.1 to 0.2	~100
0.2 to 0.5	~100
0.5 to 0.8	~100
0.8 to 1.1	~100
1.1 to 1.4	~100
1.4 to 1.7	~100
1.7 to 2.0	~100
2.0 to 2.3	~100
2.3 to 2.6	~100
2.6 to 2.9	~100
2.9 to 3.2	~100
3.2 to 3.5	~100
3.5 to 3.8	~100
3.8 to 4.1	~100
4.1 to 4.4	~100
4.4 to 4.7	~100
4.7 to 5.0	~100

uroot-browser: another interface package that pulls many together



uroot-browser - uroot-Event.root 01:38:36

```
fTracks.fBy Float16_t[]
fTracks.fCharge Double32_t[]
fTracks.fMass2 Float16_t[]
fTracks.fMeanCharge float[]
fTracks.fNpoint int32_t[]
fTracks.fNsp uint32_t[]
fTracks.fPointValue unknown[][]
fTracks.fPx float[]
fTracks.fPy float[]
fTracks.fPz float[]
fTracks.fRandom float[]
fTracks.fTArray[3] float[][3]
fTracks.fTriggerBits.fAllBits
uint8_t[][]
fTracks.fTriggerBits.fBits
uint8_t[]
fTracks.fTriggerBits.fNbits
uint32_t[]
fTracks.fTriggerBits.fNbytes
uint32_t[]
fTracks.fTriggerBits.fUniqueID
```

B Toggle sidebar Q Quit

fTracks.fPy -- Entries: 599392

Bin Range	Count
-5.2 to -4.9	~100
-4.9 to -4.6	~100
-4.6 to -4.3	~100
-4.3 to -4.0	~100
-4.0 to -3.7	~100
-3.7 to -3.4	~100
-3.4 to -3.1	~100
-3.1 to -2.8	~100
-2.8 to -2.5	~100
-2.5 to -2.2	~100
-2.2 to -1.9	~100
-1.9 to -1.6	~100
-1.6 to -1.3	~100
-1.3 to -1.0	~100
-1.0 to -0.7	~100
-0.7 to -0.4	~100
-0.4 to -0.1	~100
-0.1 to 0.2	~100
0.2 to 0.5	~100
0.5 to 0.8	~100
0.8 to 1.1	~100
1.1 to 1.4	~100
1.4 to 1.7	~100
1.7 to 2.0	~100
2.0 to 2.3	~100
2.3 to 2.6	~100
2.6 to 2.9	~100
2.9 to 3.2	~100
3.2 to 3.5	~100
3.5 to 3.8	~100
3.8 to 4.1	~100
4.1 to 4.4	~100
4.4 to 4.7	~100
4.7 to 5.0	~100

uproot-browser: another interface package that pulls many together



uproot-browser - uproot-Event.root 01:38:36

```
fTracks.fBy Float16_t[]
fTracks.fCharge Double32_t[]
fTracks.fMass2 Float16_t[]
fTracks.fMeanCharge float[]
fTracks.fNpoint int32_t[]
fTracks.fNsp uint32_t[]
fTracks.fPointValue unknown[][]
fTracks.fPx float[]
fTracks.fPy float[]
fTracks.fPz float[]
fTracks.fRandom float[]
fTracks.fTArray[3] float[][3]
fTracks.fTriggerBits.fAllBits
uint8_t[][]
fTracks.fTriggerBits.fBits
uint8_t[]
fTracks.fTriggerBits.fNbits
uint32_t[]
fTracks.fTriggerBits.fNbytes
uint32_t[]
fTracks.fTriggerBits.fUniqueID
```

B Toggle sidebar Q Quit

fTracks.fPz -- Entries: 599392

Bin Range	Count
0.0 - 0.2	~10000
0.2 - 0.4	~20000
0.4 - 0.6	~40000
0.6 - 0.8	~80000
0.8 - 1.0	~150000
1.0 - 1.2	~250000
1.2 - 1.4	~350000
1.4 - 1.6	~400000
1.6 - 1.8	~380000
1.8 - 2.0	~350000
2.0 - 2.2	~300000
2.2 - 2.4	~250000
2.4 - 2.6	~200000
2.6 - 2.8	~150000
2.8 - 3.0	~100000
3.0 - 3.2	~70000
3.2 - 3.4	~50000
3.4 - 3.6	~40000
3.6 - 3.8	~30000
3.8 - 4.0	~20000
4.0 - 4.2	~15000
4.2 - 4.4	~10000
4.4 - 4.6	~8000
4.6 - 4.8	~6000
4.8 - 5.0	~5000
5.0 - 5.2	~4000
5.2 - 5.4	~3000
5.4 - 5.6	~2000
5.6 - 5.8	~1000

uroot-browser: another interface package that pulls many together



uroot-browser - uroot-Event.root 01:38:36

```

- fIsValid bool
- fLastTrack TRef
- fMatrix[4][4] float[4][4]
- fMeasures[10] int32_t[10]
- fMuons TRefArray*
- fNseg int32_t
- fNtrack int32_t
- fNvertex uint32_t
- fTemperature float
- fTracks ClonesArray*
- fTracks.fBits uint8_t[]
- fTracks.fBx Float16_t[]
- fTracks.fBy Float16_t[]
- fTracks.fCharge Double32_t[]
- fTracks.fMass2 Float16_t[]
- fTracks.fMeanCharge float[]
- fTracks.fNpoint int32_t[]
- fTracks.fNsp uint32_t[]
- fTracks.fPointValue unknown[][]
- fTracks.fPx float[]
- fTracks.fPy float[]

```

B Toggle sidebar Q Quit

fTracks.fPz -- Entries: 599392

Bin Range	Count
0.0 - 0.2	~10000
0.2 - 0.4	~20000
0.4 - 0.6	~40000
0.6 - 0.8	~80000
0.8 - 1.0	~150000
1.0 - 1.2	~250000
1.2 - 1.4	~350000
1.4 - 1.6	~410000
1.6 - 1.8	~380000
1.8 - 2.0	~300000
2.0 - 2.2	~200000
2.2 - 2.4	~120000
2.4 - 2.6	~70000
2.6 - 2.8	~40000
2.8 - 3.0	~20000
3.0 - 3.2	~10000
3.2 - 3.4	~5000
3.4 - 3.6	~2000
3.6 - 3.8	~1000
3.8 - 4.0	~500
4.0 - 4.2	~200
4.2 - 4.4	~100
4.4 - 4.6	~50
4.6 - 4.8	~20
4.8 - 5.0	~10
5.0 - 5.2	~5
5.2 - 5.4	~2
5.4 - 5.6	~1
5.6 - 5.8	~1

uroot-browser: another interface package that pulls many together



uroot-browser - uroot-Event.root 01:38:37

```

- fClosestDistance unknow[]
- fEventName char*
- fEvtHdr EventHeader
- fEvtHdr.fDate int32_t
- fEvtHdr.fEvtNum int32_t
- fEvtHdr.fRun int32_t
- fFlag uint32_t
- fH TH1F
- fHighPt TRefArray*
- fIsValid bool
- fLastTrack TRef
- fMatrix[4][4] float[4][4]
- fMeasures[10] int32_t[10]
- fMuons TRefArray*
- fNseg int32_t
- fNtrack int32_t
- fNvertex uint32_t
- fTemperature float
- fTracks TClonesArray*
- fTracks.fBits uint8_t[]
- fTracks.fBx Float16_t[]

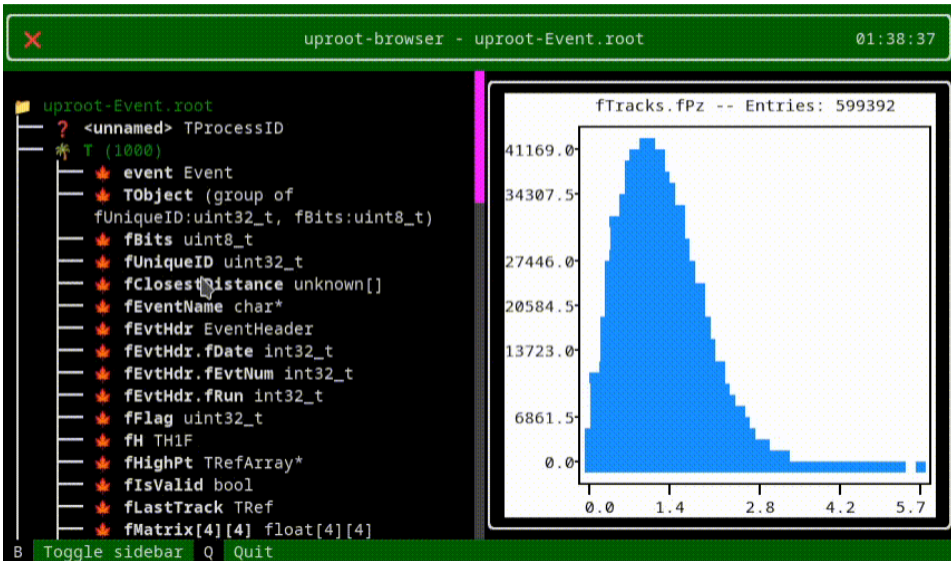
```

B Toggle sidebar Q Quit

fTracks.fPz -- Entries: 599392

Bin Range (fPz)	Count (Frequency)
0.0 - 0.2	~10000
0.2 - 0.4	~20000
0.4 - 0.6	~35000
0.6 - 0.8	~40000
0.8 - 1.0	~41000
1.0 - 1.2	~40000
1.2 - 1.4	~38000
1.4 - 1.6	~35000
1.6 - 1.8	~30000
1.8 - 2.0	~25000
2.0 - 2.2	~20000
2.2 - 2.4	~15000
2.4 - 2.6	~10000
2.6 - 2.8	~7000
2.8 - 3.0	~5000
3.0 - 3.2	~3000
3.2 - 3.4	~2000
3.4 - 3.6	~1500
3.6 - 3.8	~1000
3.8 - 4.0	~800
4.0 - 4.2	~600
4.2 - 4.4	~500
4.4 - 4.6	~400
4.6 - 4.8	~300
4.8 - 5.0	~200
5.0 - 5.2	~150
5.2 - 5.4	~100
5.4 - 5.6	~50
5.6 - 5.7	~20

uroot-browser: another interface package that pulls many together



uproot-browser: another interface package that pulls many together



uproot-browser - uproot-Event.root 01:38:38

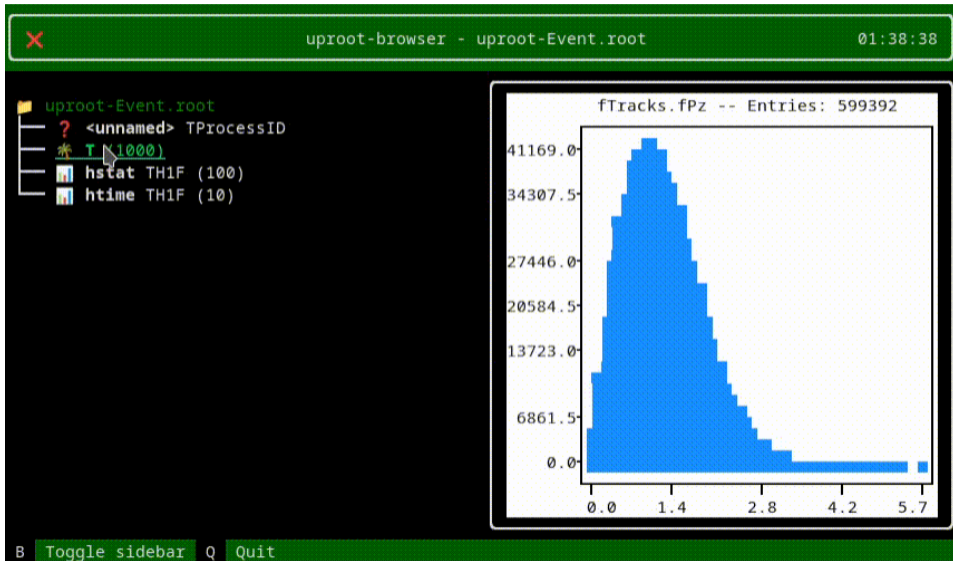
```
uproot-Event.root
├── ? <unnamed> TProcessID
├── * T (1000)
│   ├── event Event
│   ├── TObject (group of
│   │   ├── fUniqueID:uint32_t, fBits:uint8_t)
│   │   ├── fBits uint8_t
│   │   ├── fUniqueID uint32_t
│   │   ├── fClosestDistance unknown[]
│   │   ├── fEventName char*
│   │   ├── fEvtHdr EventHeader
│   │   ├── fEvtHdr.fDate int32_t
│   │   ├── fEvtHdr.fEvtNum int32_t
│   │   ├── fEvtHdr.fRun int32_t
│   │   ├── fFlag uint32_t
│   │   ├── fH TH1F
│   │   ├── fHighPt TRefArray*
│   │   ├── fIsValid bool
│   │   ├── fLastTrack TRef
│   │   └── fMatrix[4][4] float[4][4]
└──
```

B Toggle sidebar Q Quit

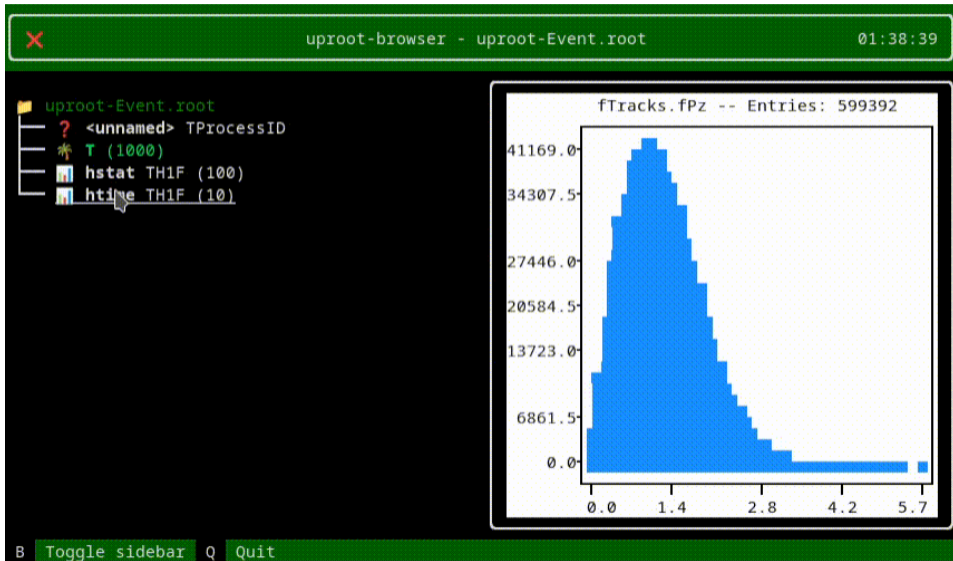
fTracks.fPz -- Entries: 599392

fTracks.fPz Bin Range	Approximate Count
0.0 - 0.2	10000
0.2 - 0.4	20000
0.4 - 0.6	35000
0.6 - 0.8	45000
0.8 - 1.0	41000
1.0 - 1.2	40000
1.2 - 1.4	38000
1.4 - 1.6	35000
1.6 - 1.8	30000
1.8 - 2.0	25000
2.0 - 2.2	20000
2.2 - 2.4	15000
2.4 - 2.6	10000
2.6 - 2.8	7000
2.8 - 3.0	5000
3.0 - 3.2	3000
3.2 - 3.4	2000
3.4 - 3.6	1500
3.6 - 3.8	1000
3.8 - 4.0	800
4.0 - 4.2	600
4.2 - 4.4	500
4.4 - 4.6	400
4.6 - 4.8	300
4.8 - 5.0	200
5.0 - 5.2	150
5.2 - 5.4	100
5.4 - 5.6	50
5.6 - 5.7	20

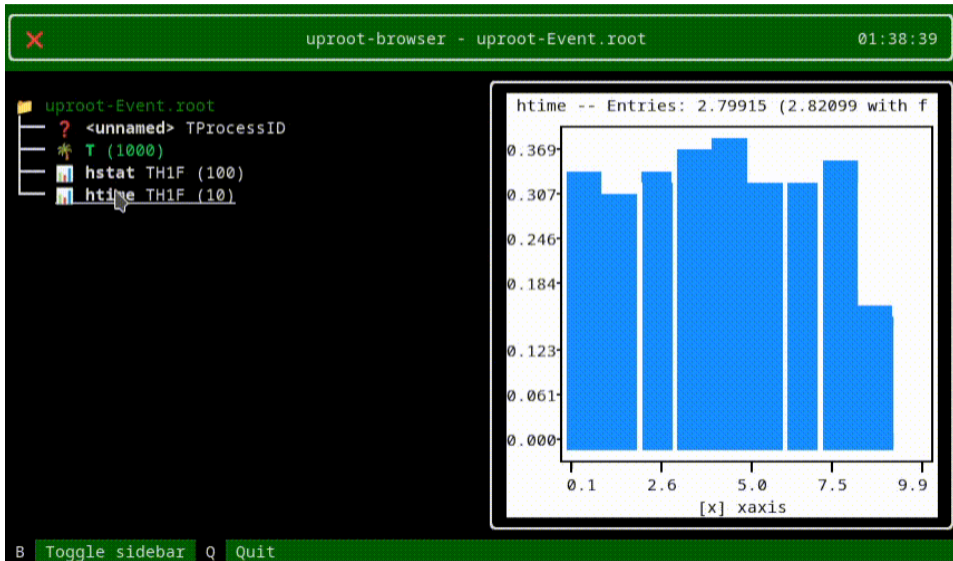
uproot-browser: another interface package that pulls many together



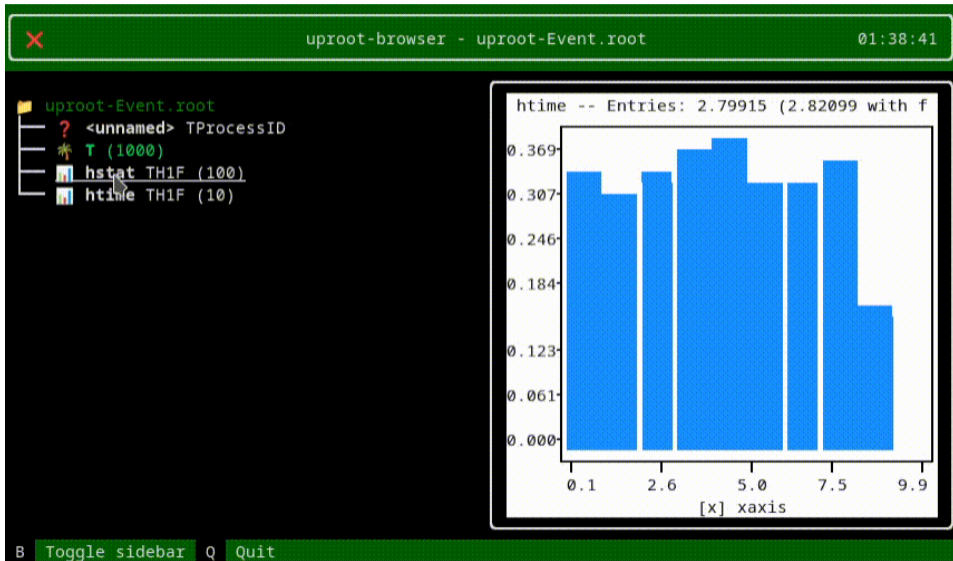
uproot-browser: another interface package that pulls many together



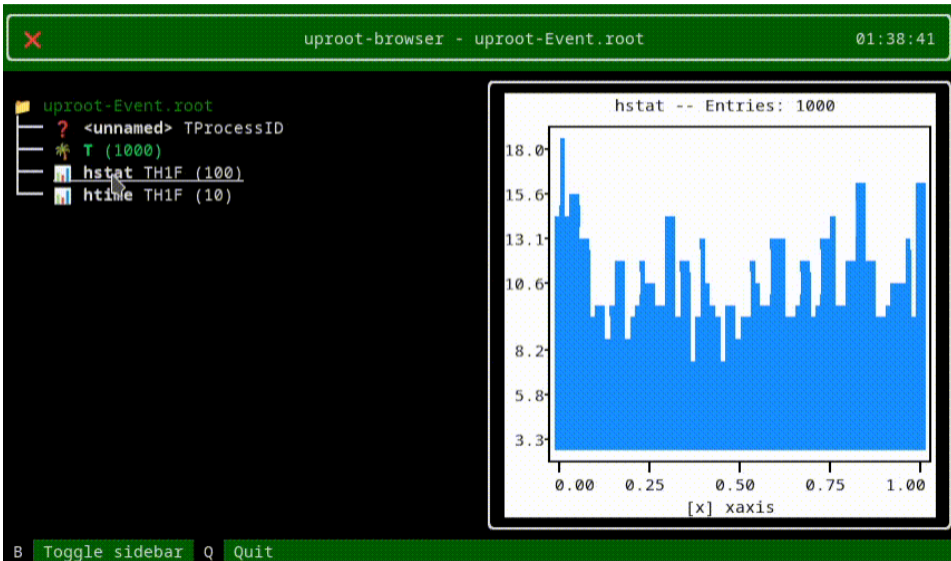
uproot-browser: another interface package that pulls many together

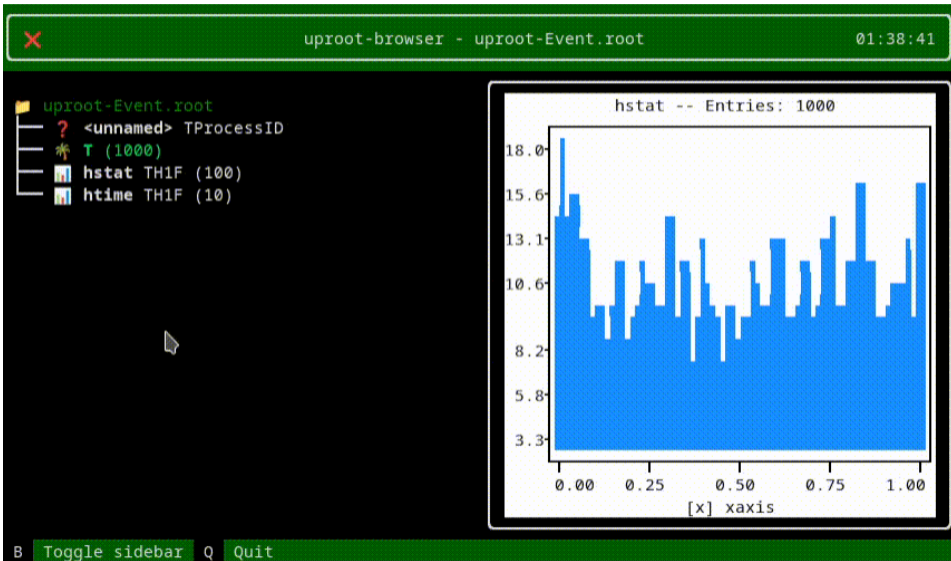


uproot-browser: another interface package that pulls many together



uroot-browser: another interface package that pulls many together







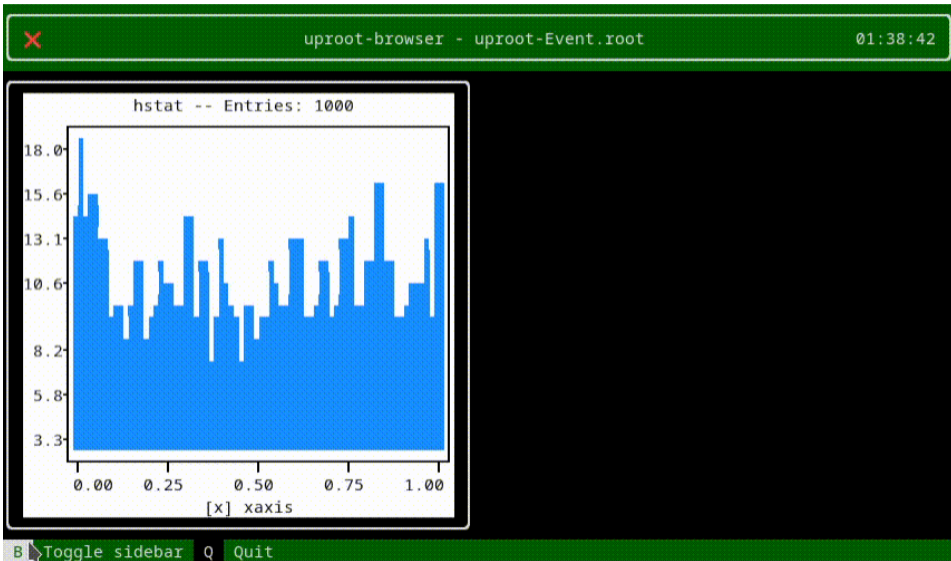
uproot-browser - uproot-Event.root 01:38:42

- uproot-Event.root
 - ? <unnamed> TProcessID
 - * T (1000)
 - hstat TH1F (100)
 - htime TH1F (10)

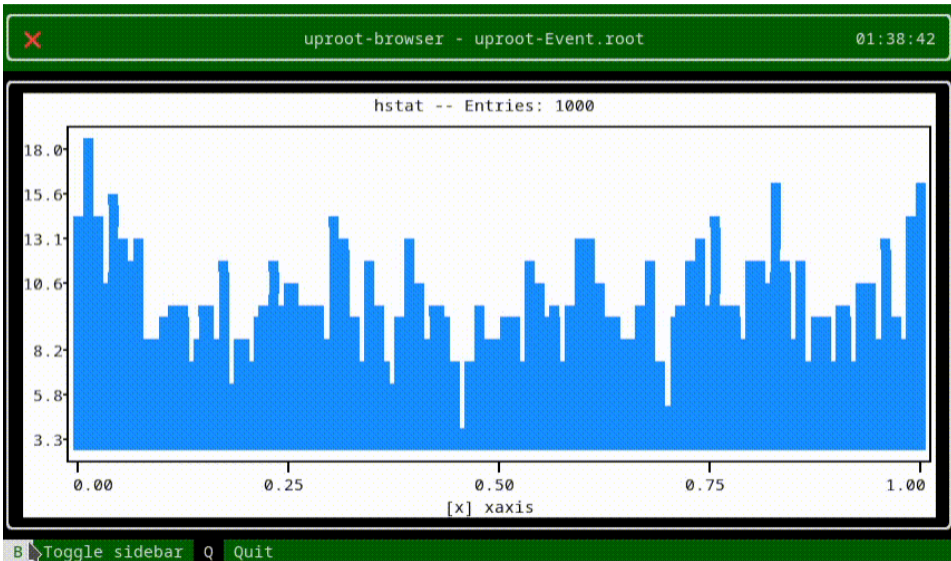
hstat -- Entries: 1000

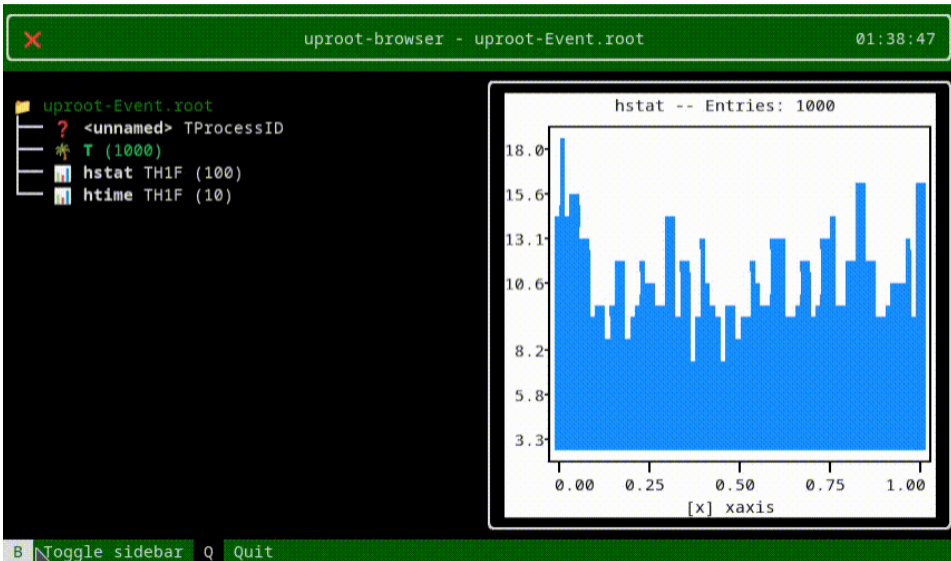
[x] axis

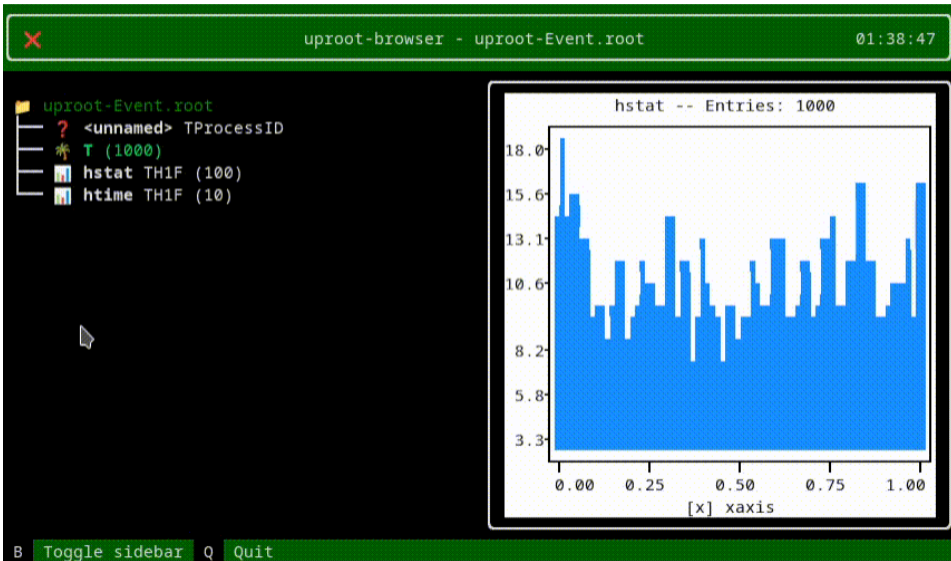
B Toggle sidebar Q Quit



uproot-browser: another interface package that pulls many together







uroot-browser: another interface package that pulls many together



Exit uroot-browser - uroot-Event.root 01:38:48

uroot-Event.root

- ? <unnamed> TProcessID
- * T (1000)
- hstat TH1F (100)
- htime TH1F (10)

hstat -- Entries: 1000

[x] axis

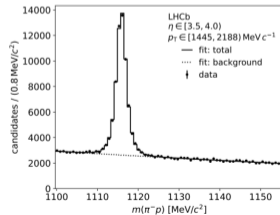
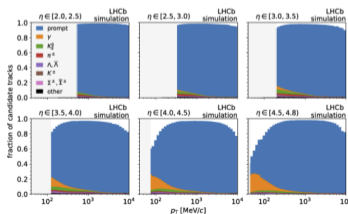
B Toggle sidebar Q Quit



LHCb Publication Using Solely Scikit-HEP Tools

Post data-processing all performed with Python HEP tools!

- Uproot: Interfacing with input ROOT files
- boost-histogram: Replace classic TH* ROOT classes;
Bonus: Multi-dimensional histograms!
- iminuit: User-friendly interface to minuit2 to process minimization
- PDF build in Python using SciPy library components



LHCb-PAPER-2021-010



My point is not that you *should*.

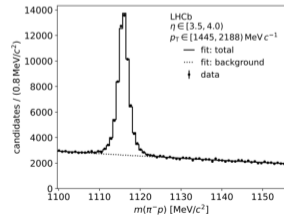
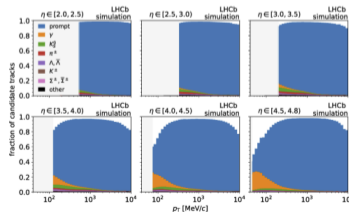
(Use any tool that gets the job done!)

My point is that you *can*. Scikit-HEP packages cover all aspects of analysis.

LHCb Publication Using Solely Scikit-HEP Tools

Post data-processing all performed with Python HEP tools!

- Uproot: Interfacing with input ROOT files
- boost-histogram: Replace classic TH* ROOT classes;
Bonus: Multi-dimensional histograms!
- iminuit: User-friendly interface to minuit2 to process minimization
- PDF build in Python using SciPy library components



LHCb-PAPER-2021-010



Overview

Timetable

Contribution List

Registration

Participant List

Videoconference

Code of conduct

EDI statement

The **IRIS-HEP AGC Tools 2022 Workshop** is dedicated to showcasing tools and workflows related to the so-called "[Analysis Grand Challenge](#)" (AGC) being organized by [IRIS-HEP](#) and partners. This workshop is a part of preparations for HSF [Analysis Ecosystems Workshop II](#).

The AGC focuses on running a physics analysis at scale, including the handling of systematic uncertainties, binned statistical analysis, reinterpretation, and end-to-end optimization. The AGC makes use of new and advanced analysis tools developed by the community in the Python ecosystem and relies on the development of the required cyberinfrastructure to be executed at scale. A specific goal of the AGC is to demonstrate technologies envisioned for use at the HL-LHC.

- *Foundation libraries (**uproot**, **awkward**, **hist**, **mplhep**)*
- *Queries with **func_adl** and data delivery with **ServiceX***
- *Columnar analysis with **coffea***
- *Statistical inference: **pyhf** and **cabinetry***
- *From data delivery to statistical inference: **ServiceX**, **coffea**, **cabinetry** & **pyhf***
- *Data management with **Skyhook***
- *Scale-out with coffea: **coffea-casa analysis facility***
- *Analysis user experience with Python HEP data science tools in different experiments*
- *Experiment related discussions*

The agenda will be composed of hands-on tutorials based on various tools and services developed in the Python ecosystem by and for the particle physics community, and room to discuss the current status of projects and interfaces.



- ▶ Sample analysis on Run-2 CMS Open Data
 - ▶ open data is crucial!
 - ▶ currently, MiniAOD → NanoAOD-like
 - ▶ will switch to NanoAOD when available
 - ▶ mirrors PHYSLITE
 - ▶ thanks to the CMS DPOA team
- ▶ Datasets [listed here](#)
- ▶ Everything openly developed
 - ▶ [github:iris-hep/analysis-grand-challenge](#)
- ▶ Using coffea-casa analysis facilities at [UNL](#), [UChicago](#), [Fermilab](#)

Nov 2021: demo at [first AGC workshop](#)

Apr 2022: second [second AGC workshop](#)

mid 2022: benchmarking of components

spring 2023: full scale challenge

analysis-grand-challenge@iris-hep.org ([sign up](#))

Sample $t\bar{t}$ analysis: open data delivery \rightarrow inference in a notebook



```

python3 main.py --process-name ProcessName --file-name File --dataset-name DatasetName --input-dir InputDir --output-dir OutputDir --log-dir LogDir --help
# Example usage: python3 main.py --process-name ttbar --file-name ttbar --dataset-name ttbar --input-dir /data --output-dir /output --log-dir /log

# Parameters
process_name = ProcessName
file_name = File
dataset_name = DatasetName
input_dir = InputDir
output_dir = OutputDir
log_dir = LogDir

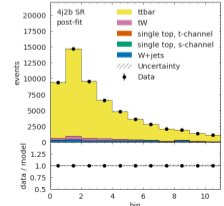
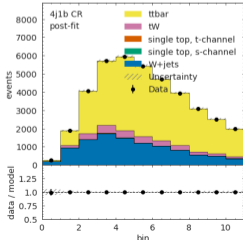
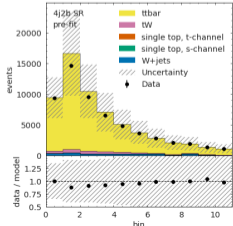
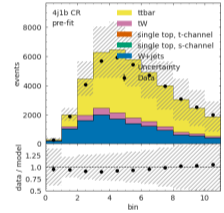
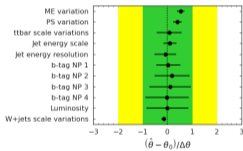
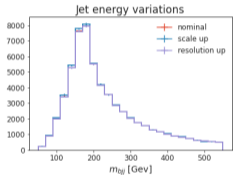
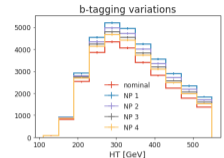
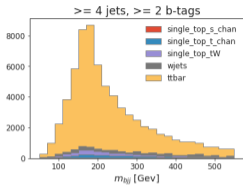
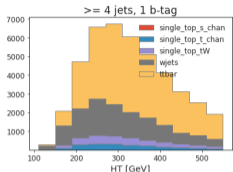
# Main function
def main():
    # Load data
    data = load_data(input_dir)

    # Process data
    processed_data = process_data(data)

    # Save results
    save_results(processed_data, output_dir)

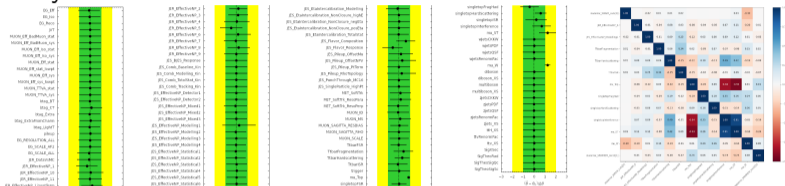
    # Log progress
    log_progress(log_dir)

if __name__ == '__main__':
    main()
    
```

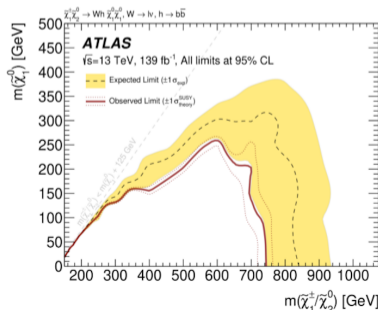
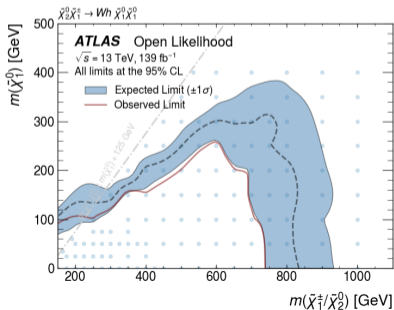




Laundry list of systematics



Reproduce the published plot (in a tutorial)



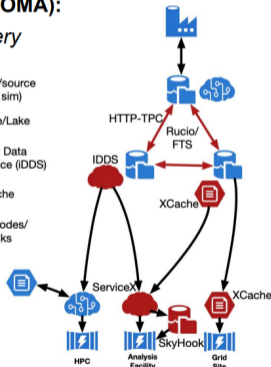
AGC tests connections between services and tools



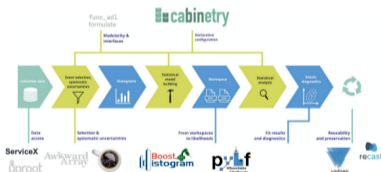
Data Organization, Management and Access (DOMA):

Data delivery

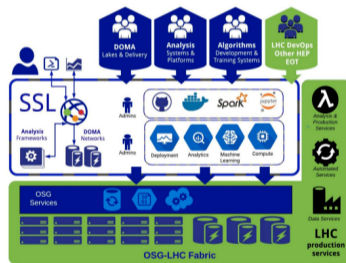
- Data Factory/source (e.g. T0 or sim)
- Data Store/Lake
- Intelligent Data Delivery Service (IDDS)
- Data Cache
- Compute Nodes/ Data Sinks



Analysis Systems (AS): tools



Scalable Systems Laboratory (SSL): deployment techniques and resources



AGC tests connections between services and tools



uproot
Awkward Array
FASTJET
VECTOR
mplhep
Boost istogram
cabinetry
Func ADL
iminuit
Coffea
pyhf

ServiceX

Coffea-Casa
XCache
func

Analysis specific frameworks and packages (available in Docker container)

Data delivery service (k8s)

Optional services (k8s)



The Scikit-HEP project



The idea, in just one sentence

The Scikit-HEP project (<http://scikit-hep.org/>) is a community-driven and community-oriented project with the aim of providing Particle Physics at large with a Python package containing core and common tools.

What it is NOT ...

- A replacement for ROOT
- A replacement for the Python ecosystem based on NumPy, scikit-learn & co.

... and what IT IS

- Bridge/glue between the ROOT-based and the Python scientific ecosystem
 - Expand typical toolkit of HEP physicists
 - Common definitions and APIs to ease “cross-talk”
- Project similar to the Astropy project – learn from good examples ;-)
- We are building a community, **engaging with (future) collaborators in various experiments**

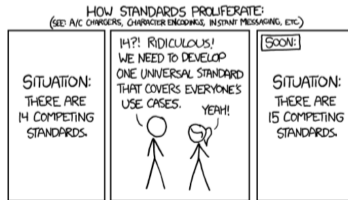
Eduardo Rodrigues



VISION 25

HOW TO BENEFIT FROM THE WORLD OUT THERE

- ▶ Industry has grown a LOT
- ▶ Google/Facebook/Amazon/Microsoft/Apple employ a lot of very clever people
- ▶ Doing your own bare-bones 'GPU' framework will not keep up
- ▶ Better to re-use/interop with eg. TensorFlow / SPARK (or lower level like Thrust) and focus on how to leverage those
- ▶ But what if you pick the "wrong" standard, and it dead-ends?
 - ▶ Major reason why in the past we 'did it ourselves'....
 - ▶ Contribute back (eg. to standards)



Gerhard Raven



2017

```
events
  .Select(e => e.Data.eventWeight)
  .FuturePlot("event_weights", "Sample EventWeights",
    100, 0.0, 1000.0)
  .Save(hdir);
```

What we want to plot

1D Histogram Specification

Save the plot in a file

today

```
In [4]: filtered = (edm
  .Select(lambda jets: jets.Where(lambda j: j.pT > 30 and abs(j.eta) < 2.5))
  )
```

Behind your back, the `uproot` transformer in ServiceX translates this into a column operation!

Gordon Watts



Representing objects as flat arrays



For simple collections of records (e.g. particles), these arrays have the same interpretation as ROOT TLeaves:

- ▶ data arrays contain all values, ignoring event boundaries,
- ▶ size array contains the size of each event's collection.

Except for runtime calculations, rather than just on disk.

For collections of collections (with fixed, known depth), we can extend this definition recursively:

Given: [[a b c] [d e f g]] [[h] [i j]]
Data array: a b c d e f g h i j
Recursive counter: 2 3 4 2 1 2

embryonic Awkward Array

