University of CINCINNATI
ARTS AND SCIENCES

dianahep

The Scikit-HEP Project

Eduardo Rodrigues
University of Cincinnati

CHEP 2018

23RD INTERNATIONAL CONFERENCE ON
COMPUTING IN HIGH ENERGY AND NUCLEAR PHYSICS

9-13 July 2018
National Palace of Culture
Sofia, Bulgaria

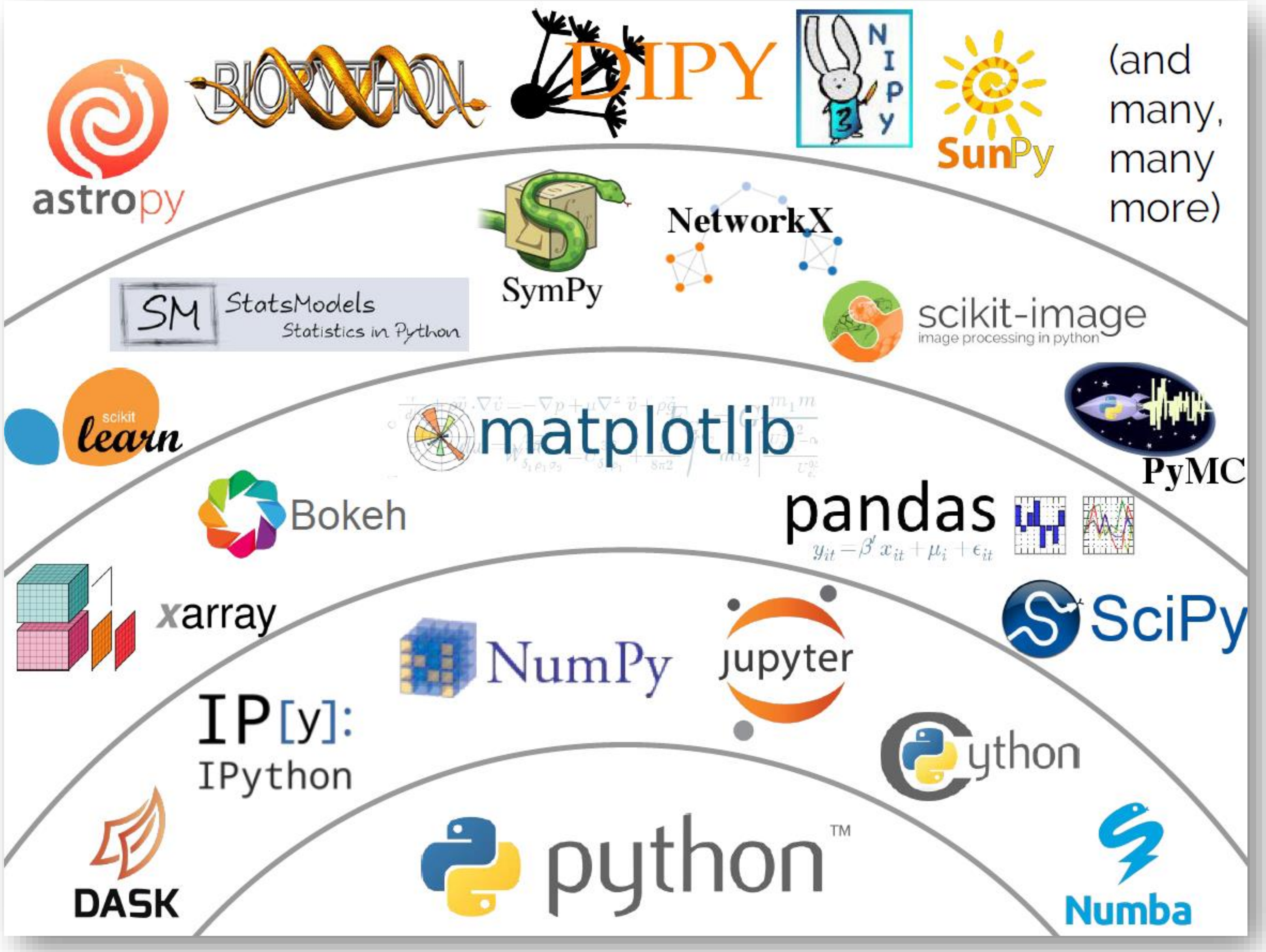
How’s the Python scientific ecosystem like, outside HEP?

Domain-specific

Python’s

Scientific

stack



Jake VanderPlas,
The Unexpected Effectiveness of Python in Science,
PyCon 2017

How does it compare vis-à-vis the HEP ecosystem ?

❑ Traditionally, we have been in a rather disjoint ecosystem

HEP, largely ROOT-based

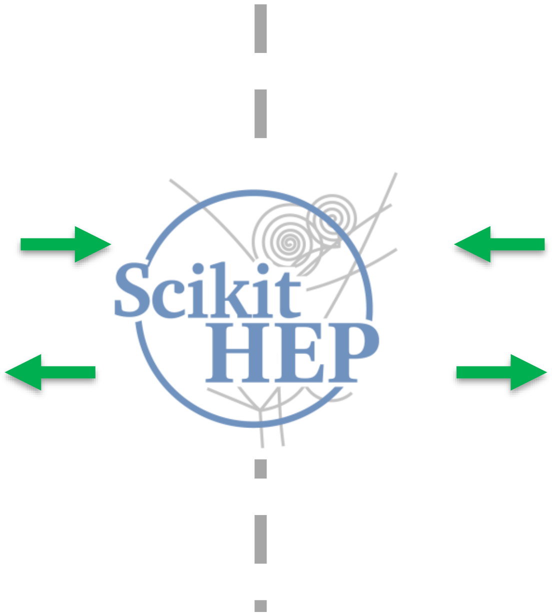
❑ ROOT for almost everything

❑ Toolkit for modeling / fitting: RooFit

❑ Statistics: RooStats

❑ Machine learning: TMVA

❑ Etc. …



Scientific Computing in Python

❑ The father of them all: SciPy

❑ Data manipulation: NumPy, Pandas

❑ Plotting: matplotlib, seaborn, Bokeh

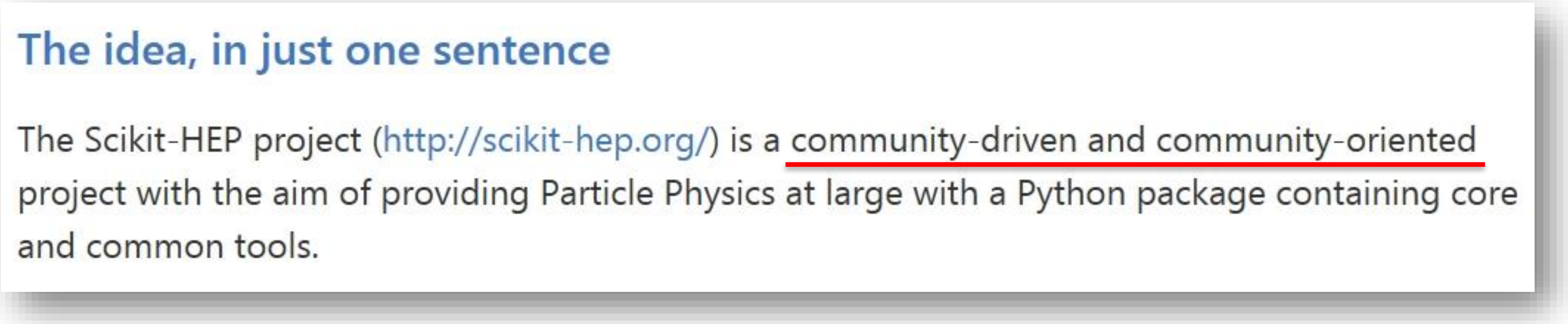
❑ Machine learning: scikit-learn, TensorFlow

❑ Etc. …

❑ Various initiatives here and there to link both worlds, but only tackling specific topics

⇨ ***Need for a more general(ised) effort, domain-specific oriented***

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

❑ **An initiative to improve the interoperability between HEP tools and the Python ecosystem**

- Expand the typical tool~~kit~~set for HEP physicists

- Set common APIs and definitions to ease “cross-talk”

❑ **An initiative to build a community of developers and users**

❑ **An effort to improve discoverability of relevant tools**

Interoperability

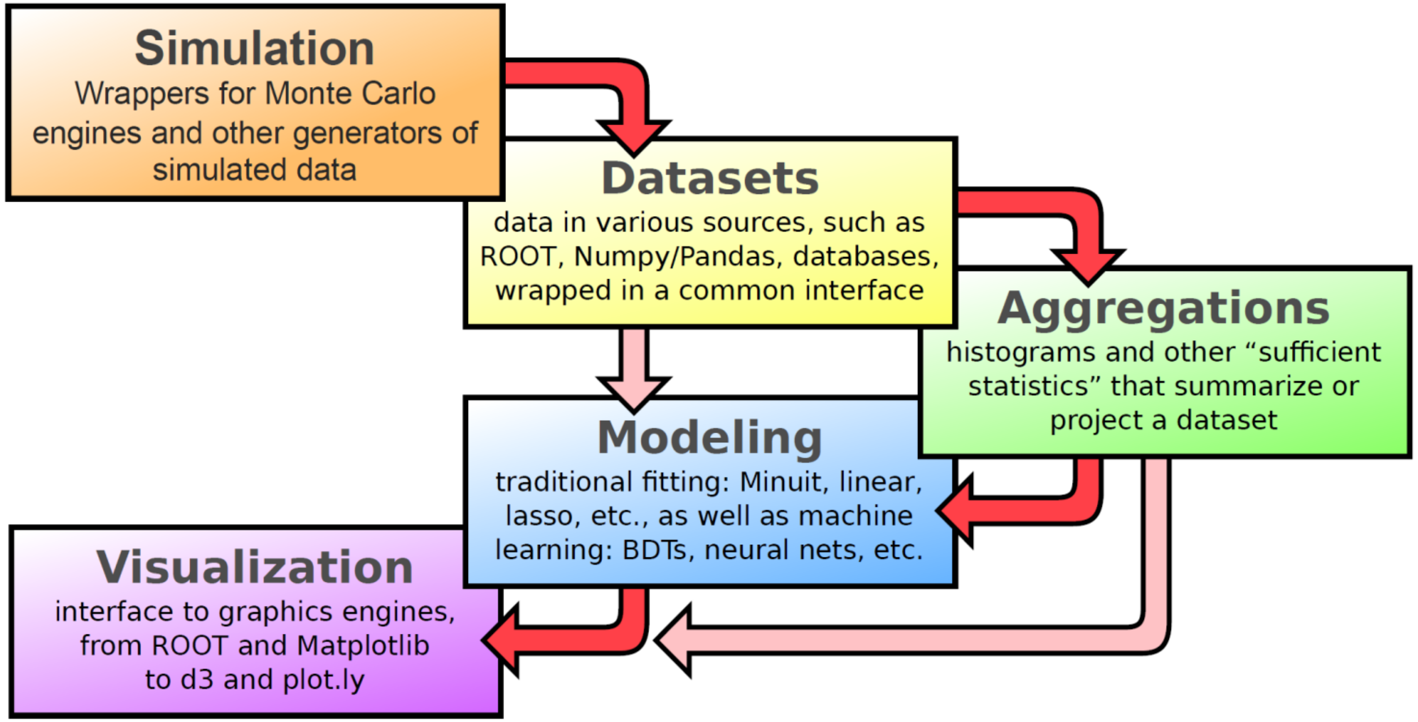
Collaboration

Reproducibility

© dianahep

The Scikit-HEP project – 5 « pillars »

❑ **Project built around 5 “pillars” covering all grand topics**



Simulation
Wrappers for Monte Carlo engines and other generators of simulated data

Datasets
data in various sources, such as ROOT, Numpy/Pandas, databases, wrapped in a common interface

Aggregations
histograms and other “sufficient statistics” that summarize or project a dataset

Modeling
traditional fitting: Minuit, linear, lasso, etc., as well as machine learning: BDTs, neural nets, etc.

Visualization
interface to graphics engines, from ROOT and Matplotlib to d3 and plot.ly

Also other utilities …

- **Modules for units and constants**
- **Maths and statistics**
- **Conversion between style of expressions**
  - ROOT to numexpr
- **Provenances**
- **Etc…**

Scikit-HEP project – toolset / packages overview

Scikit HEP

uproot

Minimalist ROOT I/O in pure Python and Numpy

scikit-hep

Starting point of project. Contains tools for maths, kinematics, units, etc.

histbook

Versatile, high-performance histogram toolkit for Numpy

numpythia

Interface between PYTHIA and NumPy

Vega Scope

Minimal viewer of Vega / Vega-Lite plots in your web browser from local or remote Python processes

formulate

Easy conversions between different styles of expressions

pyjet

Interface between FastJet and NumPy

And other packages, which tend to be now superseded, hence deprecated …

Analysis examples

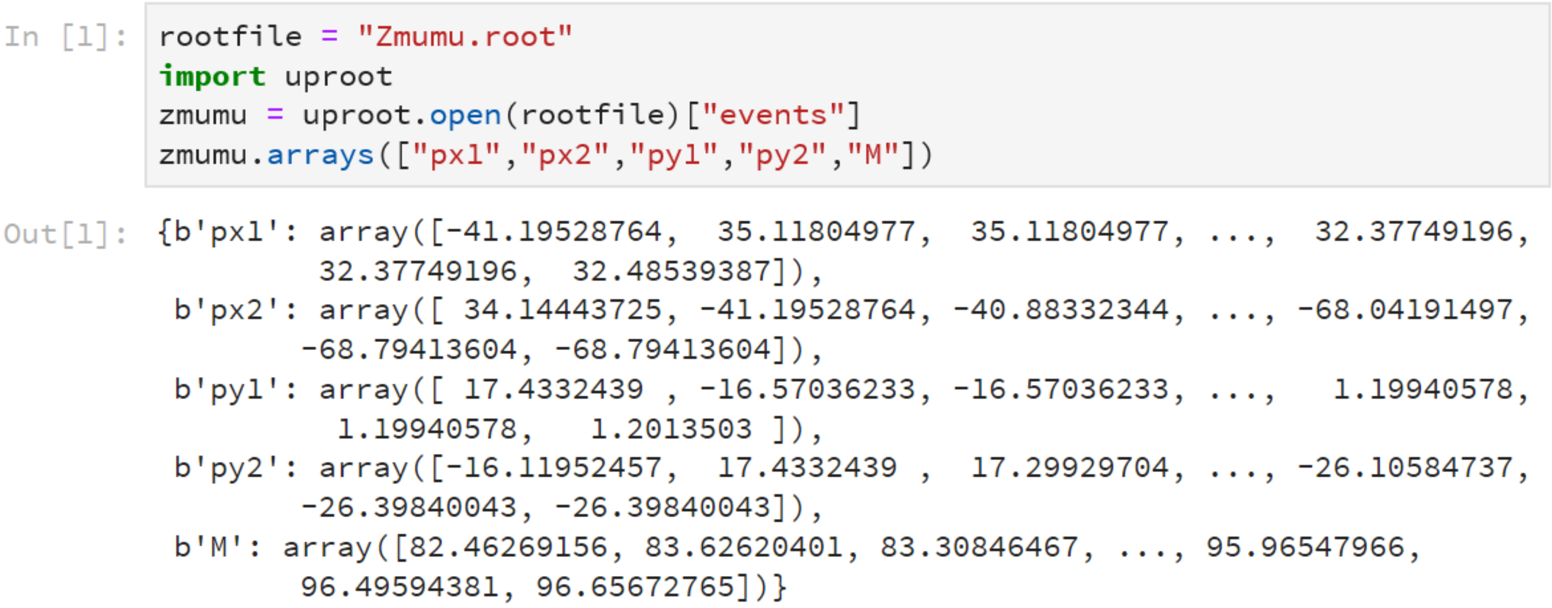
Many thanks to Matthieu Marinangeli
for the examples material!
See also his longer presentation at PyHEP 2018.

Analysis examples – read a ROOT file

- ❑ **Exploration of a sample of $Z \rightarrow \mu^+ \mu^-$ events, stored in a ROOT ntuple**

Minimalist ROOT I/O
in pure Python and Numpy

- ❑ **Need only Numpy, no ROOT, using the pure I/O library uproot !**
- ❑ **Ntuple (TTree name="events") read as a dictionary of arrays**



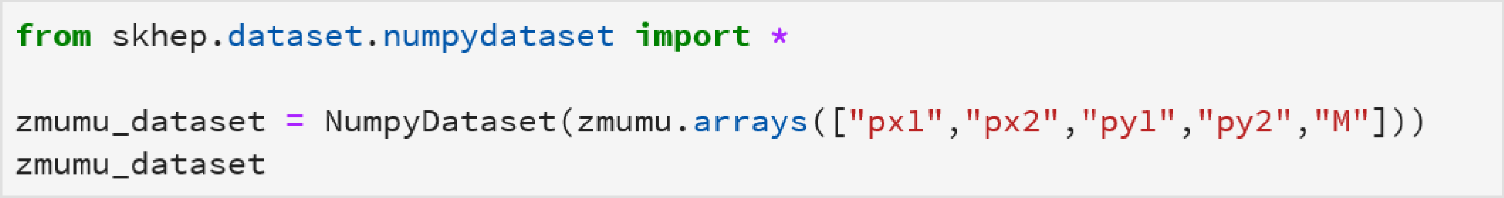
```
In [1]: rootfile = "Zmumu.root"
        import uproot
        zmumu = uproot.open(rootfile)["events"]
        zmumu.arrays(["px1","px2","py1","py2","M"])
```

```
Out[1]: {b'px1': array([-41.19528764,  35.11804977,  35.11804977, ...,  32.37749196,
                 32.37749196,  32.48539387]),
         b'px2': array([ 34.14443725, -41.19528764, -40.88332344, ..., -68.04191497,
                -68.79413604, -68.79413604]),
         b'py1': array([ 17.4332439 , -16.57036233, -16.57036233, ...,   1.19940578,
                  1.19940578,   1.2013503 ]),
         b'py2': array([-16.11952457,  17.4332439 ,  17.29929704, ..., -26.10584737,
                -26.39840043, -26.39840043]),
         b'M': array([82.46269156, 83.62620401, 83.30846467, ..., 95.96547966,
                96.49594381, 96.65672765])}
```

- ❑ **uproot contains much more functionality and in particular can read on trivial data structures in Ttrees …**

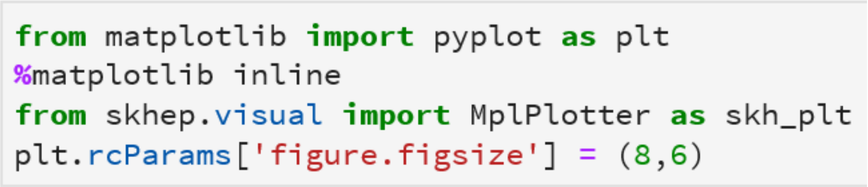
Analysis examples – datasets and their visualisation

Starting point of project. Contains tools for maths, kinematics, units, etc.



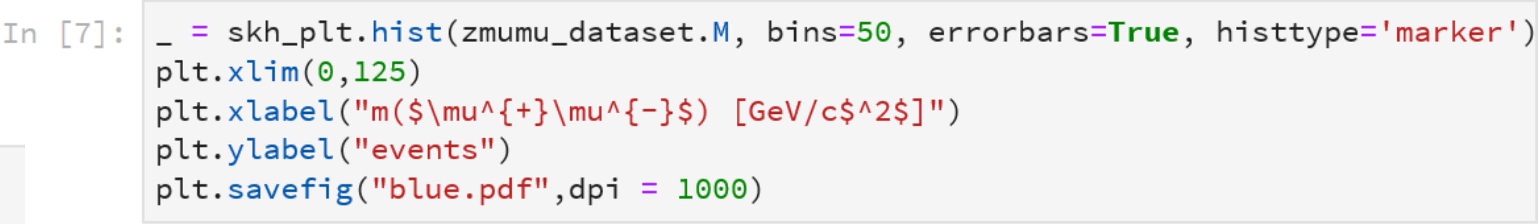
```
from skhep.dataset.numpydataset import *

zmumu_dataset = NumpyDataset(zmumu.arrays(["px1","px2","py1","py2","M"]))
zmumu_dataset
```

Advantage of these dataset classes is that *provenance information* is stored, variables are easily created on the fly

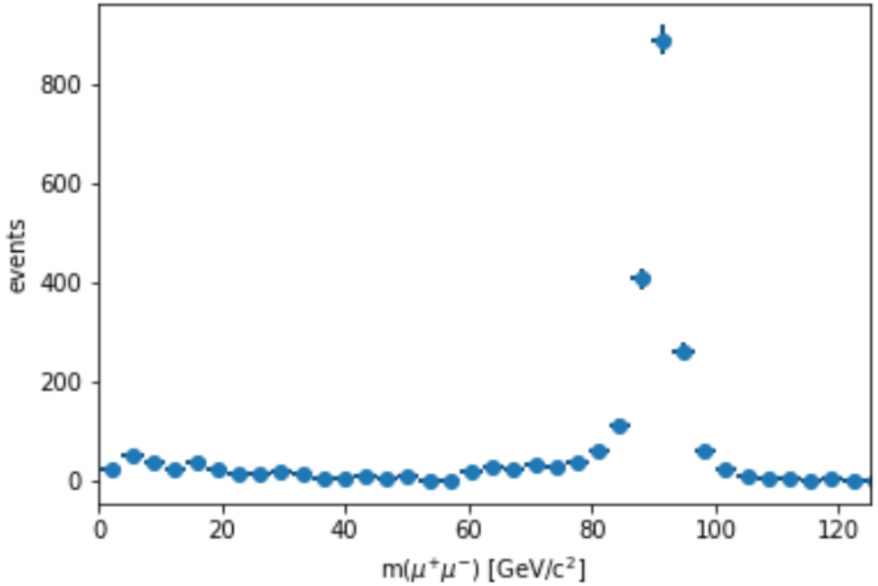


```
from matplotlib import pyplot as plt
%matplotlib inline
from skhep.visual import MplPlotter as skh_plt
plt.rcParams['figure.figsize'] = (8,6)
```

- ❑ **Plotting with matplotlib with error bars**



```
In [7]: _ = skh_plt.hist(zmumu_dataset.M, bins=50, errorbars=True, histtype='marker')
        plt.xlim(0,125)
        plt.xlabel("m($\mu^{+}\mu^{-}$) [GeV/c$^2$]")
        plt.ylabel("events")
        plt.savefig("blue.pdf",dpi = 1000)
```

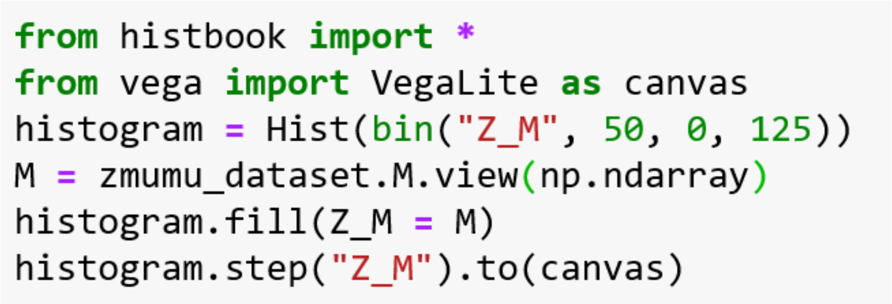


Analysis examples – data aggregation

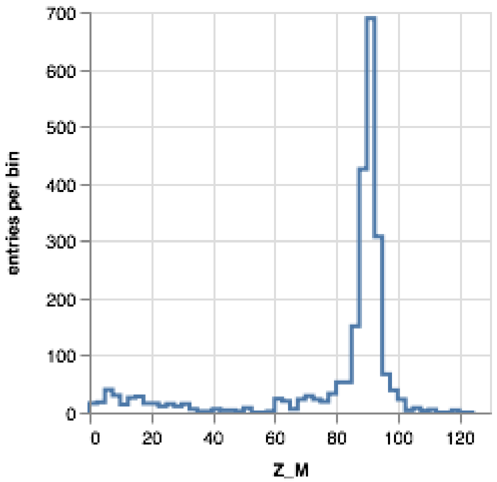
- ❑ Aggregate data – typically, ntuples – in versatile histograms
  - Of arbitrary number of dimensions
  - With methods for selecting, rebinning, and projecting into lower-dimensional space
- ❑ Histogram data exportable to a variety of formats, such as ROOT, Pandas, etc.
- ❑ Histograms can be plotted with Vega-Lite, a high-level grammar of interactive graphics

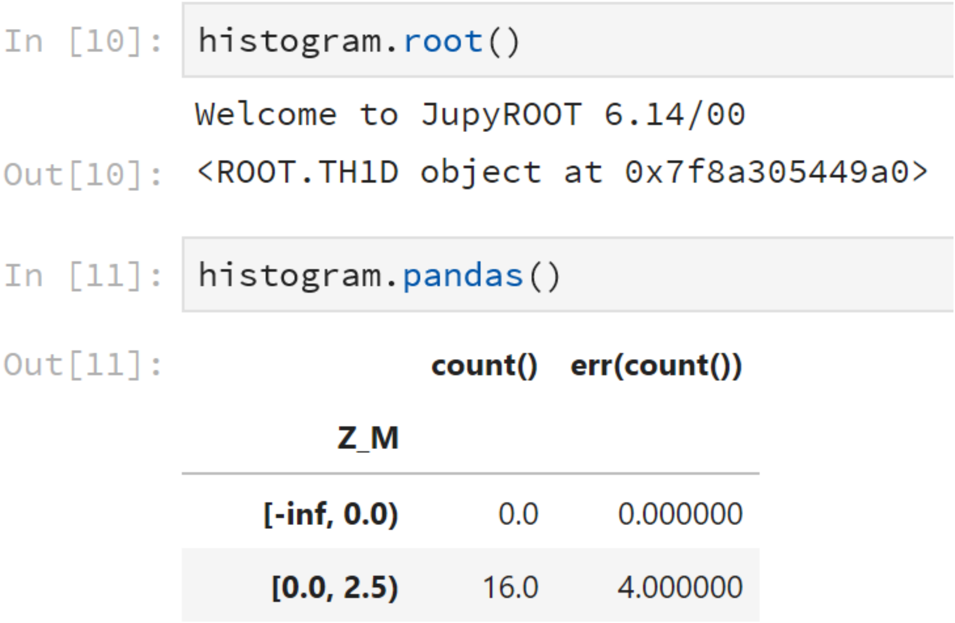
histbook

Versatile, high-performance histogram toolkit for Numpy



```
from histbook import *
from vega import VegaLite as canvas
histogram = Hist(bin("Z_M", 50, 0, 125))
M = zmumu_dataset.M.view(np.ndarray)
histogram.fill(Z_M = M)
histogram.step("Z_M").to(canvas)
```





```
In [10]: histogram.root()

         Welcome to JupyROOT 6.14/00

Out[10]: <ROOT.TH1D object at 0x7f8a305449a0>

In [11]: histogram.pandas()

Out[11]:
```

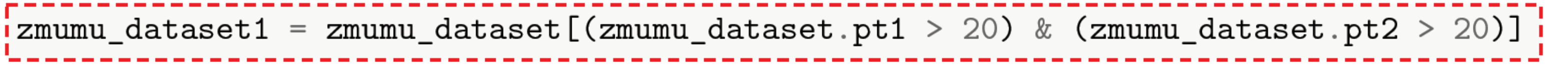
	count()	err(count())
Z_M		
[-inf, 0.0)	0.0	0.000000
[0.0, 2.5)	16.0	4.000000

Analysis examples – dataset selection

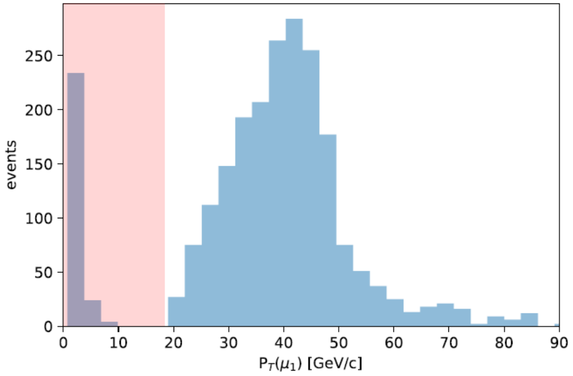
formulate

Easy conversions between different styles of expressions

Use the NumpyDataset for selection: reject low muon p_T events →

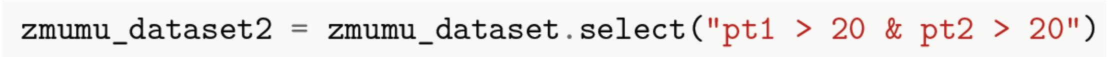


```
zmumu_dataset1 = zmumu_dataset[(zmumu_dataset.pt1 > 20) & (zmumu_dataset.pt2 > 20)]
```



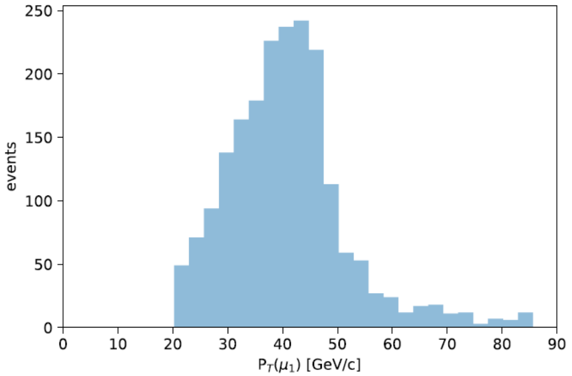
With ROOT I would just do :

- tree.CopyTree(**“pt1 > 20 && pt2 > 20”**), Possible with the select method of NumpyDataset.

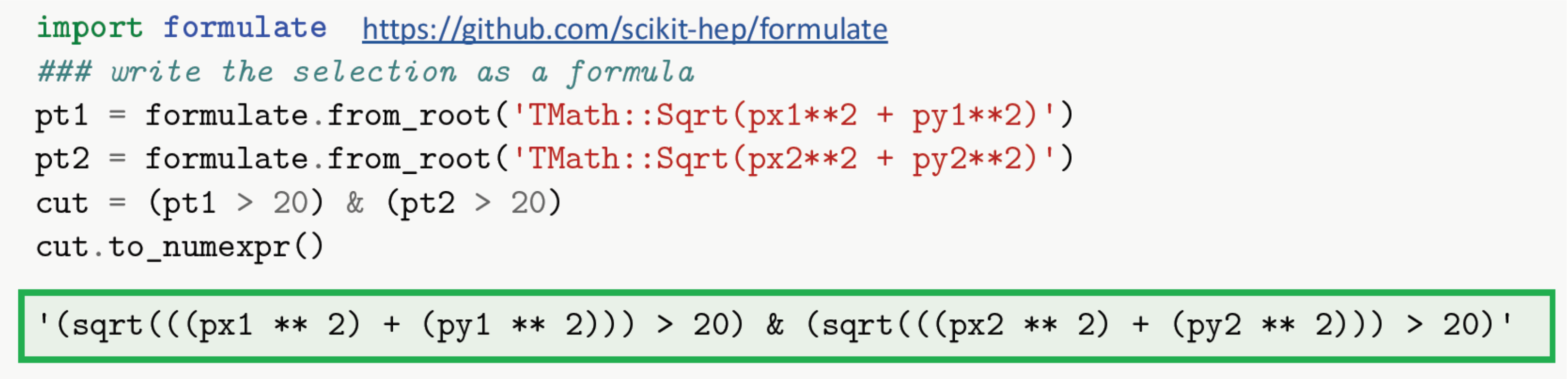


```
zmumu_dataset2 = zmumu_dataset.select("pt1 > 20 & pt2 > 20")
```

→

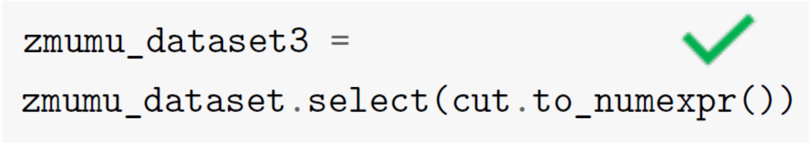


- tree.CopyTree(**“TMath::Sqrt(px1**2 + py1**2) > 20 && TMath::Sqrt(px2**2 + py2**2) > 20”**). Need to be converted to **numexpr style.**



```
import formulate   https://github.com/scikit-hep/formulate
### write the selection as a formula
pt1 = formulate.from_root('TMath::Sqrt(px1**2 + py1**2)')
pt2 = formulate.from_root('TMath::Sqrt(px2**2 + py2**2)')
cut = (pt1 > 20) & (pt2 > 20)
cut.to_numexpr()
```

```
'(sqrt(((px1 ** 2) + (py1 ** 2))) > 20) & (sqrt(((px2 ** 2) + (py2 ** 2))) > 20)'
```

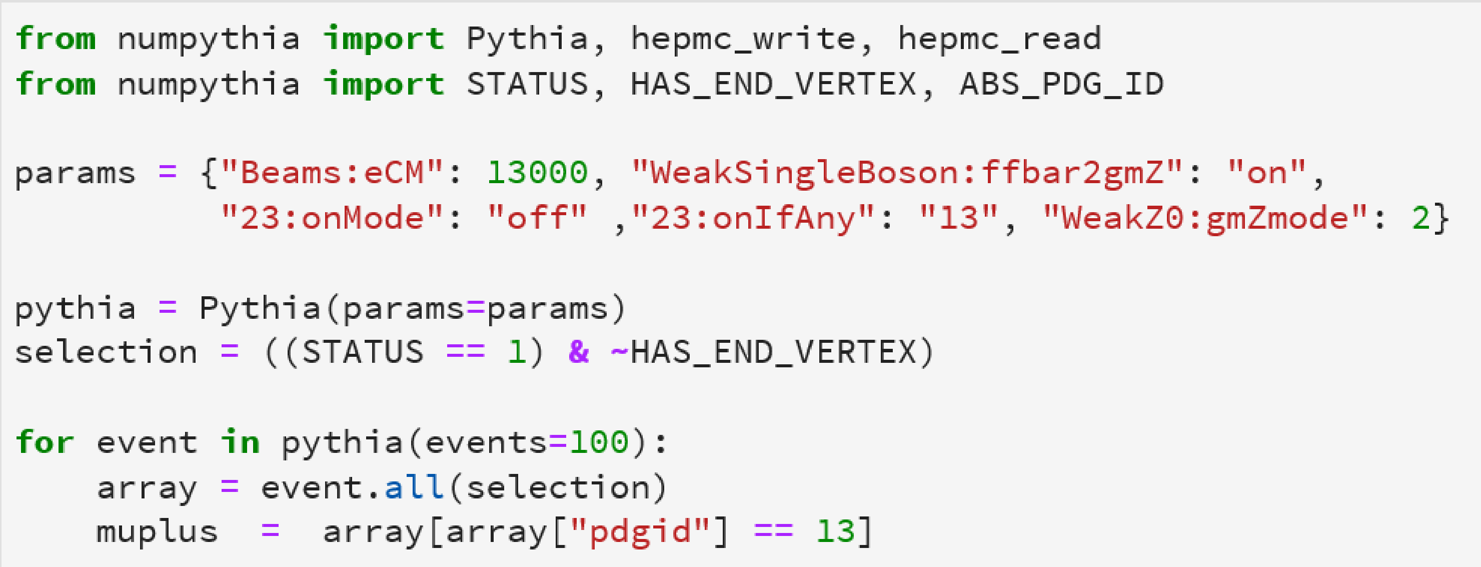


```
zmumu_dataset3 =
zmumu_dataset.select(cut.to_numexpr())
```

✓

Analysis examples – simulation & jet clustering

❑ **Generate events with Pythia and pipe them into NumPy arrays**

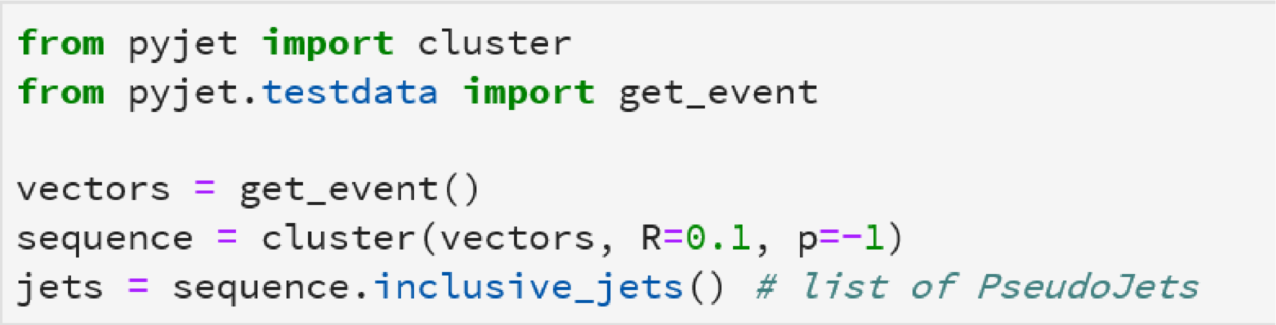


```
from numpythia import Pythia, hepmc_write, hepmc_read
from numpythia import STATUS, HAS_END_VERTEX, ABS_PDG_ID

params = {"Beams:eCM": 13000, "WeakSingleBoson:ffbar2gmZ": "on",
          "23:onMode": "off" ,"23:onIfAny": "13", "WeakZ0:gmZmode": 2}

pythia = Pythia(params=params)
selection = ((STATUS == 1) & ~HAS_END_VERTEX)

for event in pythia(events=100):
    array = event.all(selection)
    muplus  =  array[array["pdgid"] == 13]
```

numpythia

Interface between PYTHIA and NumPy

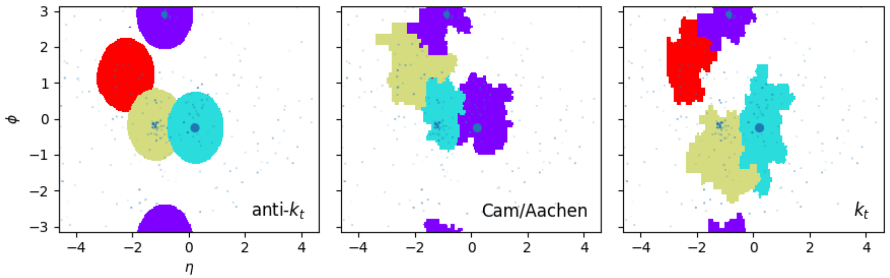
❑ **Possible to feed those events into FastJet**



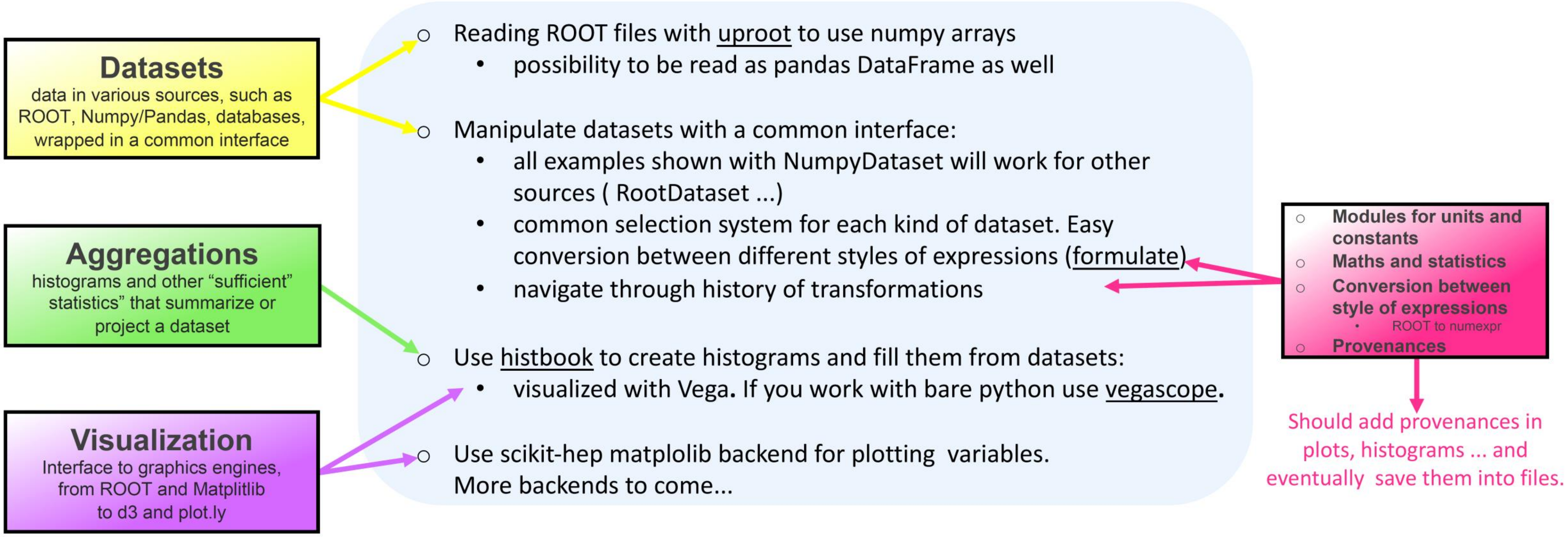
```
from pyjet import cluster
from pyjet.testdata import get_event

vectors = get_event()
sequence = cluster(vectors, R=0.1, p=-1)
jets = sequence.inclusive_jets() # list of PseudoJets
```

pyjet

Interface between FastJet and NumPy



Analysis examples – outlook



Datasets
data in various sources, such as ROOT, Numpy/Pandas, databases, wrapped in a common interface

Aggregations
histograms and other "sufficient" statistics" that summarize or project a dataset

Visualization
Interface to graphics engines, from ROOT and Matplitlib to d3 and plot.ly

- ○ Reading ROOT files with uproot to use numpy arrays
  - possibility to be read as pandas DataFrame as well
- ○ Manipulate datasets with a common interface:
  - all examples shown with NumpyDataset will work for other sources (RootDataset ...)
  - common selection system for each kind of dataset. Easy conversion between different styles of expressions (formulate)
  - navigate through history of transformations
- ○ Use histbook to create histograms and fill them from datasets:
  - visualized with Vega. If you work with bare python use vegascope.
- ○ Use scikit-hep matplolib backend for plotting variables. More backends to come...

- ○ **Modules for units and constants**
- ○ **Maths and statistics**
- ○ **Conversion between style of expressions**
  - ROOT to numexpr
- ○ **Provenances**

Should add provenances in plots, histograms ... and eventually save them into files.

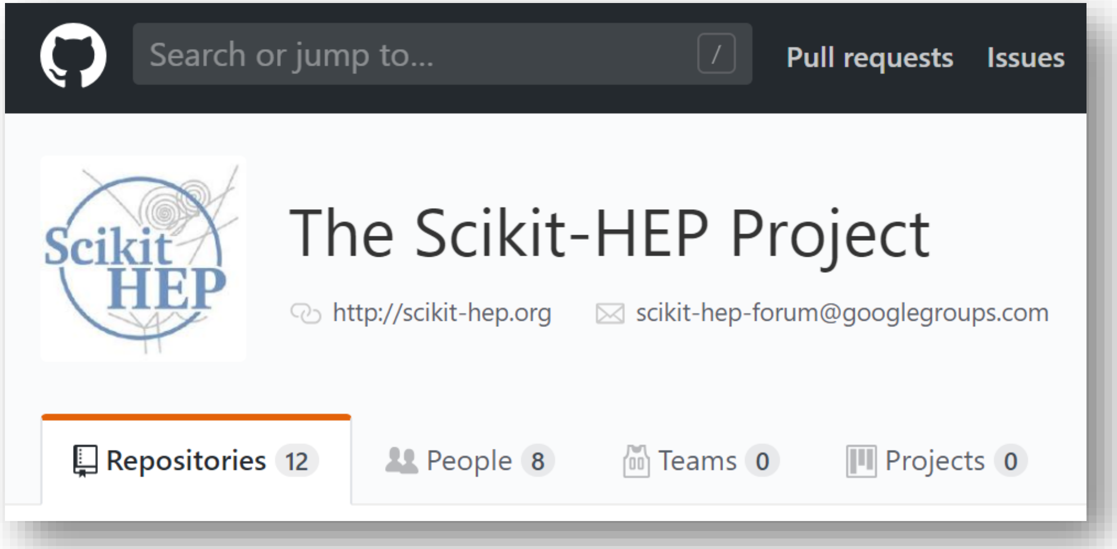
Interested ? Want to try it ? And contribute ?

❑ We are a community ⇒ everybody welcome !

- Particularly interesting to have a good sampling from the various experiments

❑ A lot to be done …

❑ … and we need feedback too !



The Scikit-HEP Project

http://scikit-hep.org scikit-hep-forum@googlegroups.com

Repositories 12 People 8 Teams 0 Projects 0

Links

❑ GitHub: https://github.com/scikit-hep/

❑ Website: http://scikit-hep.org/

Mailing lists

❑ Get in touch with the team “privately”: scikit-hep-admins@googlegroups.com

❑ Forum for anyone: scikit-hep-forum@googlegroups.com

Thank you