physt

DIANA meeting, 1st October 2018

Jan Pipek, Showmax

janpipek/physt

Background

2001-2007 masters in physics (HEP)

2007-2015 Ph.D. in medical physics (Geant4)

2015-2017 post-doc in medical physics (Geant4)

2017- data scientist @ Showmax, Prague

Motivation

```
import numpy as np
histogram = np.histogram(heights)
```

```
(array([  4,   22,   96, 228, 272,          # Frequencies
        226, 104,   38,    9,    1]),


 array([132.1841, 141.0516, 149.9191,      # Edges
        158.7866, 167.6541, 176.5216,
        185.3891, 194.2566, 203.1241,
        211.9916, 220.8591])
```

☹ Tuple of arrays?

Motivation (cont’d)

- In 2016, no adequate histogramming in Python (?)
- Lots of particle/dose distributions (2D, 3D) to visualize
- Will to create a useful open source library on my own

=> Physt

Target use cases

- Data exploration
- Compact representation of distributions
- Visualization / presentation

General, non-field-specific audience.

Design goals

- simple & familiar API (~numpy, ~pandas)
- histogram as first-class object (ROOT-inspired)
- no complex dependencies
    - **numpy** necessary
    - **matplotlib** recommended
- extensibility (visualization, computing engines, IO)

Status

https://github.com/janpipek/physt

528 commits,

2 main branches

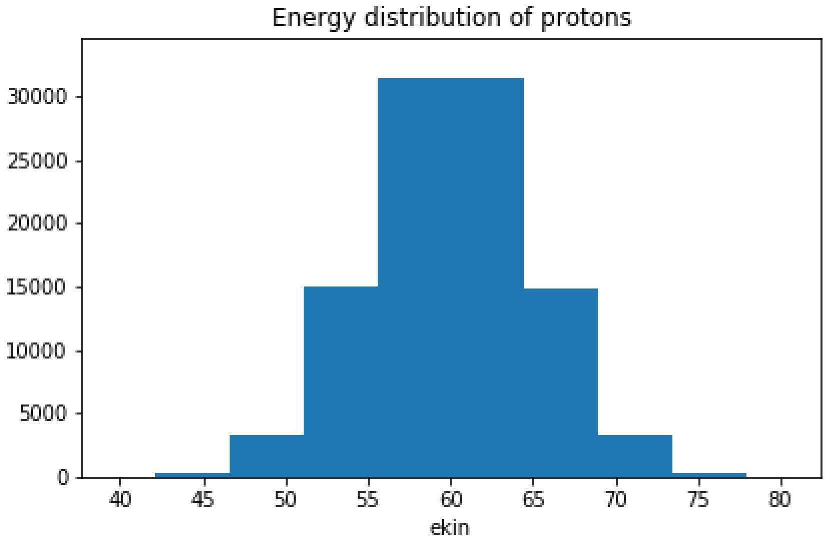
- version 0.3.43 (rich features)
- re-design 0.4 (goal: cleaner API)

Example

```python
import pandas as pd
from physt import h1

particles = pd.read_csv("protons.csv")
h = h1(particles["energy"], title="Energy distribution of protons")
```

```
Histogram1D(bins=(10,), total=100000, dtype=int64)
```

```
h.plot()
```


Energy distribution of protons
30000
25000
20000
15000
10000
5000
0
40
45
50
55
60
65
70
75
80
ekin

```
h.frequencies
```

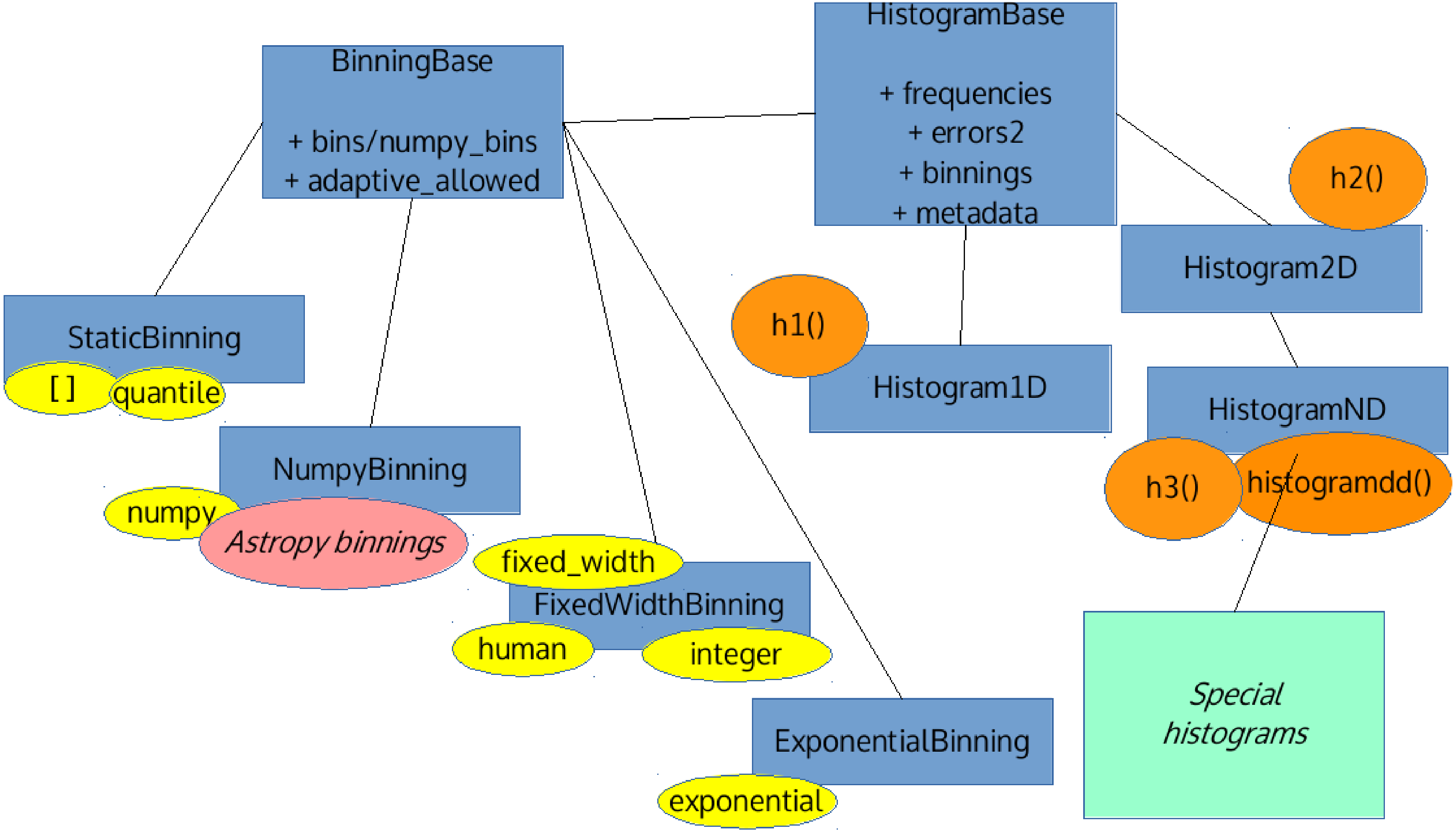
```
array([   18,    346,   3383, 14978, 31434, 31348, 14827,   3318,    330,   18
```

```
h.bins
```

```
array([[ 38.83518235,  ...,  81.791677  ]])
```

```
h.binning
```

```
NumpyBinning(array([ 38.83518235, ...,  81.791677  ]))
```

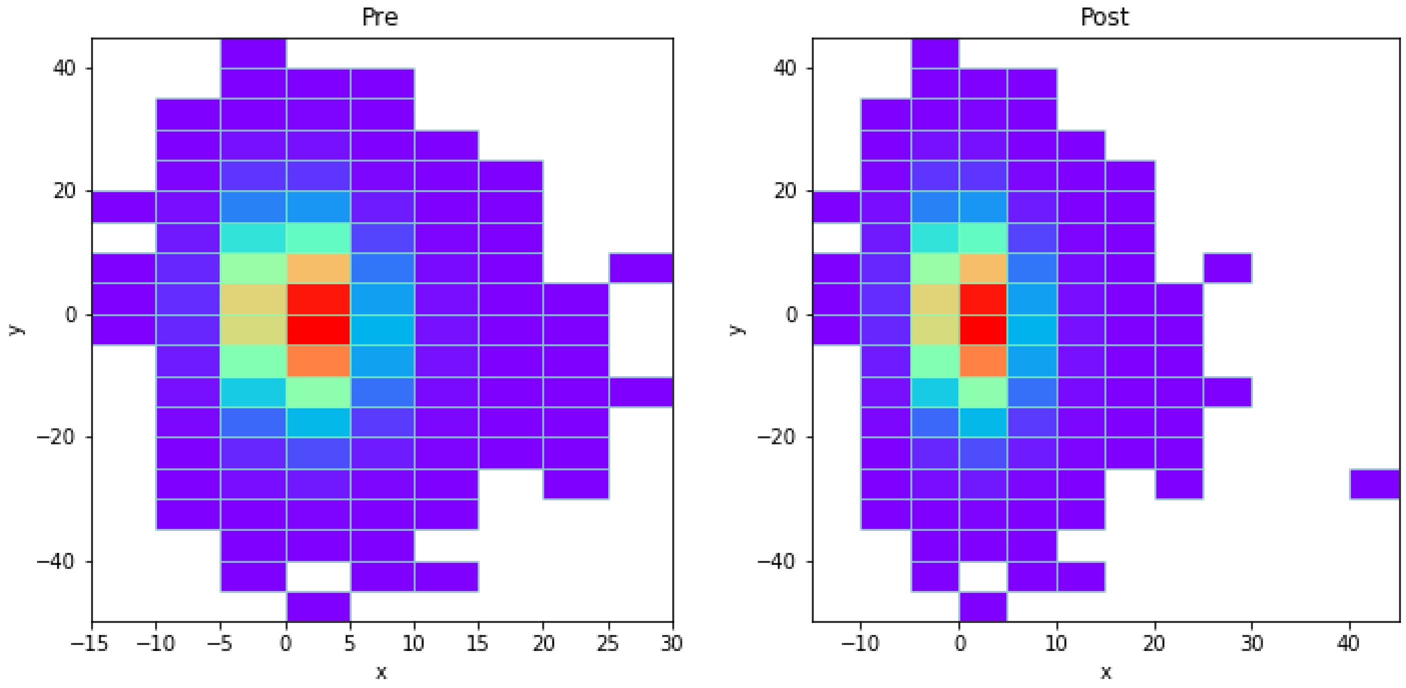

HistogramBase
+ frequencies
+ errors2
+ binnings
+ metadata
BinningBase
+ bins/numpy_bins
+ adaptive_allowed
h2()
Histogram2D
h1()
Histogram1D
StaticBinning
[ ]
quantile
HistogramND
NumpyBinning
h3()
histogramdd()
numpy
Astropy binnings
fixed_width
FixedWidthBinning
human
integer
Special histograms
ExponentialBinning
exponential

Binning schemas

- numpy (+ optimized bin counts)
- fixed-width (adaptive)
    - human (special case)
    - integer (special case)
- exponential
- quantile

Adaptive binning

```
hx = h2(particles["x"], particles["y"], "fixed_width", 5, adaptive=True)

hx.plot(figsize=(5, 5), show_zero=False, show_colorbar=False, cmap="rainb

hx << (43.4, -27.5)
hx.plot(figsize=(5, 5), show_zero=False, show_colorbar=False, cmap="rainb
```



Other features

- arithmetics (+ - * /)
- statistics (mean, bin variance…)
- projections, slicing
- coordinate transformations (cylindrical, spherical)

Computation engines

- Currently, **numpy** is doing most of the work.
- Experimental usage of **dask** for "big" data.
- **tensorflow**?
- **HDembinski/histogram**?

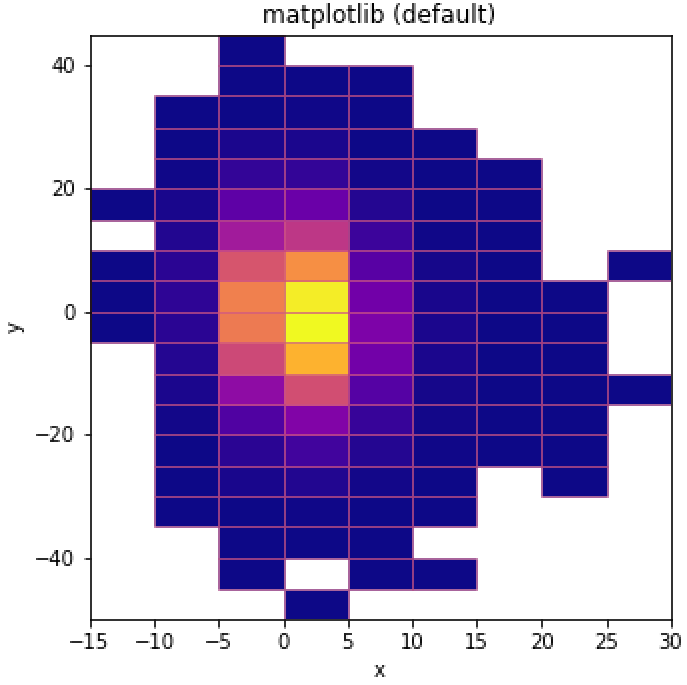
Interoperability

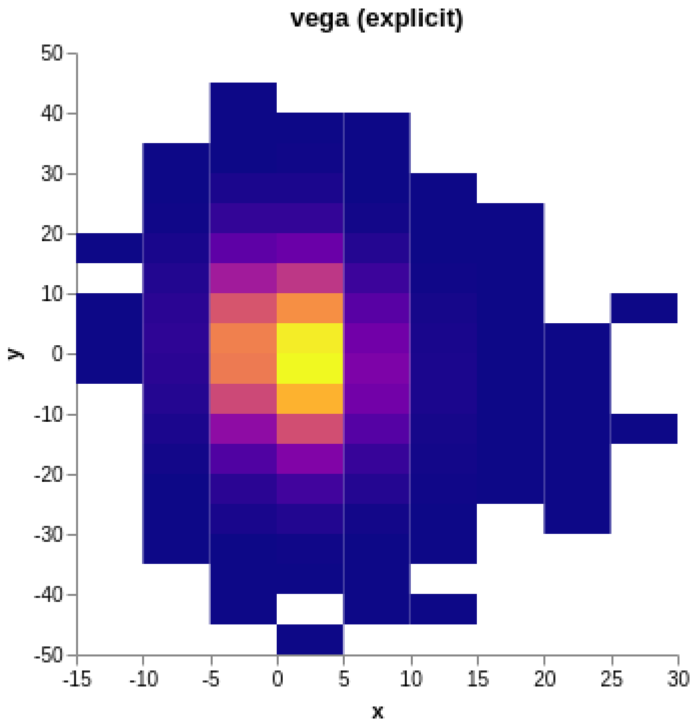
- pandas, xarray, numpy
- ROOT? Geant4 histograms CSV
- file I/0: JSON, protobuf, HDF5

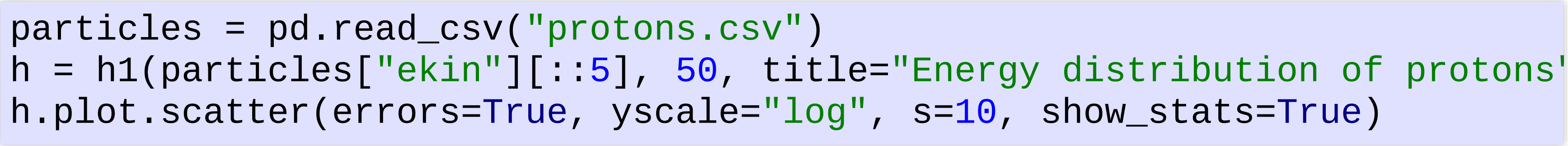
Plotting backends

- **matplotlib** (standard)
- **vega** (for notebooks)
- **plotly** (way to go?)
- **ascii** (wish I had it)

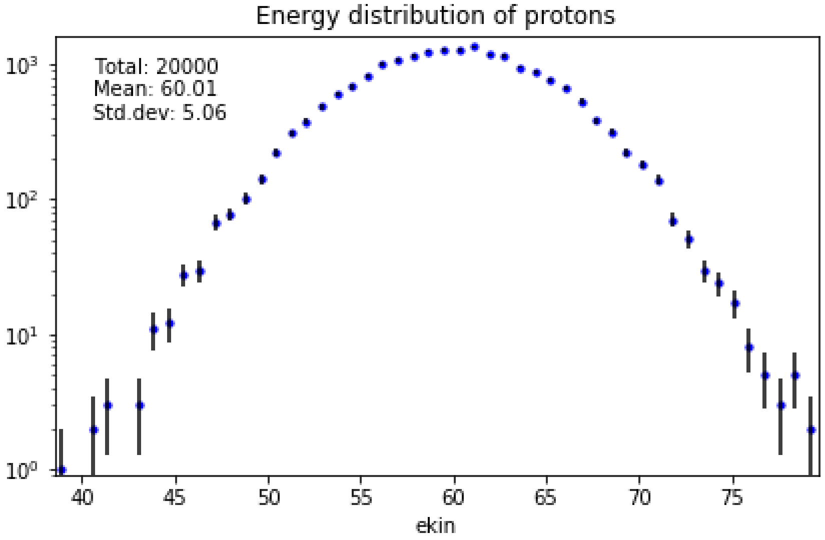
```
hx = h2(particles["x"], particles["y"], "fixed_width", 5)

# Matplotlib
hx.plot(show_zero=False, cmap="plasma", title="matplotlib (default)")

# Vega
hx.plot(backend="vega", show_zero=False, cmap="plasma", title="vega (expl
```



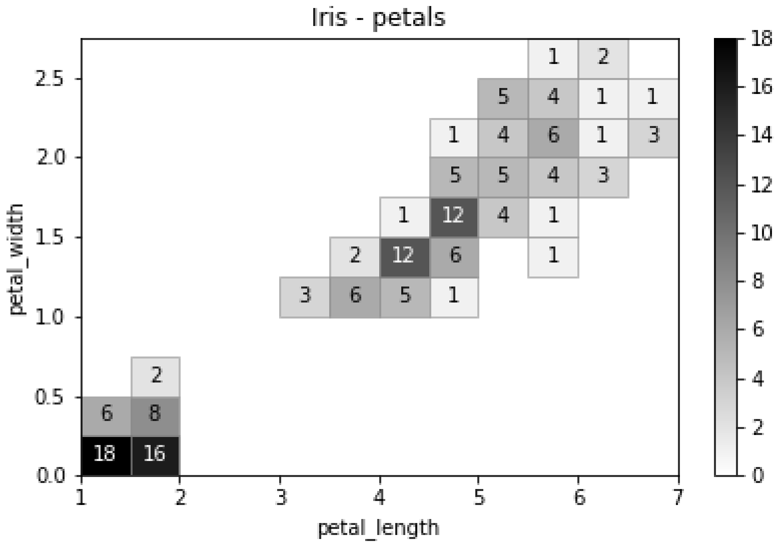


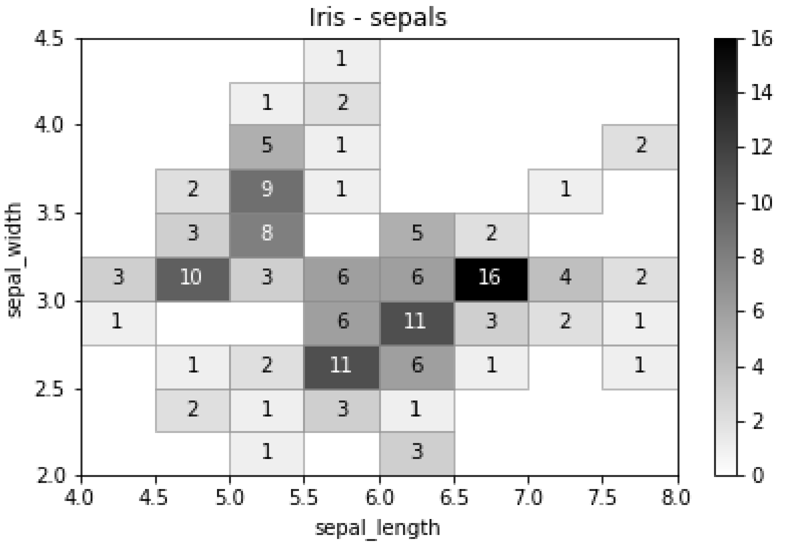


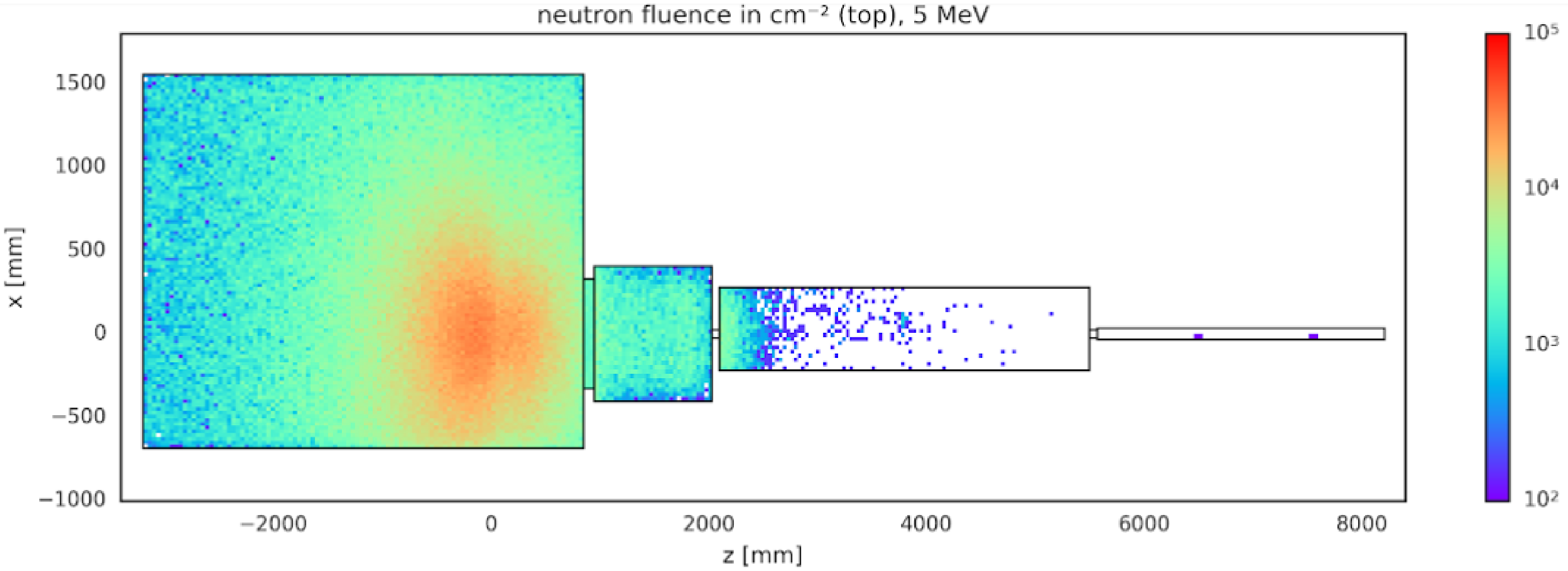
```
particles = pd.read_csv("protons.csv")
h = h1(particles["ekin"][::5], 50, title="Energy distribution of protons'
h.plot.scatter(errors=True, yscale="log", s=10, show_stats=True)
```

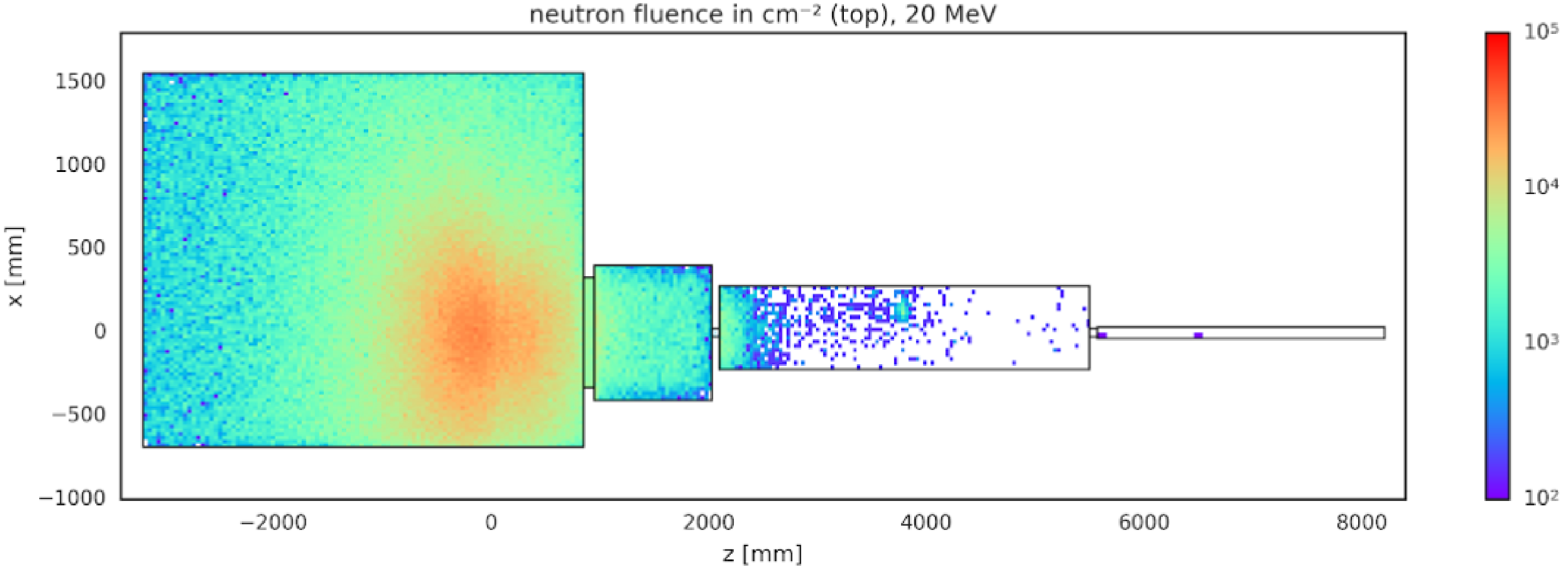


```
iris = seaborn.load_dataset('iris')
iris_hist = h(iris[["sepal_length", "sepal_width", "petal_length", "petal
              "human", name="Iris")

sepals = iris_hist.projection("sepal_length", "sepal_width")
petals = iris_hist.projection("petal_length", "petal_width")

sepals.plot(show_zero=False, show_values=True, title="Iris - sepals")
petals.plot(show_zero=False, show_values=True, title="Iris - petals")
```

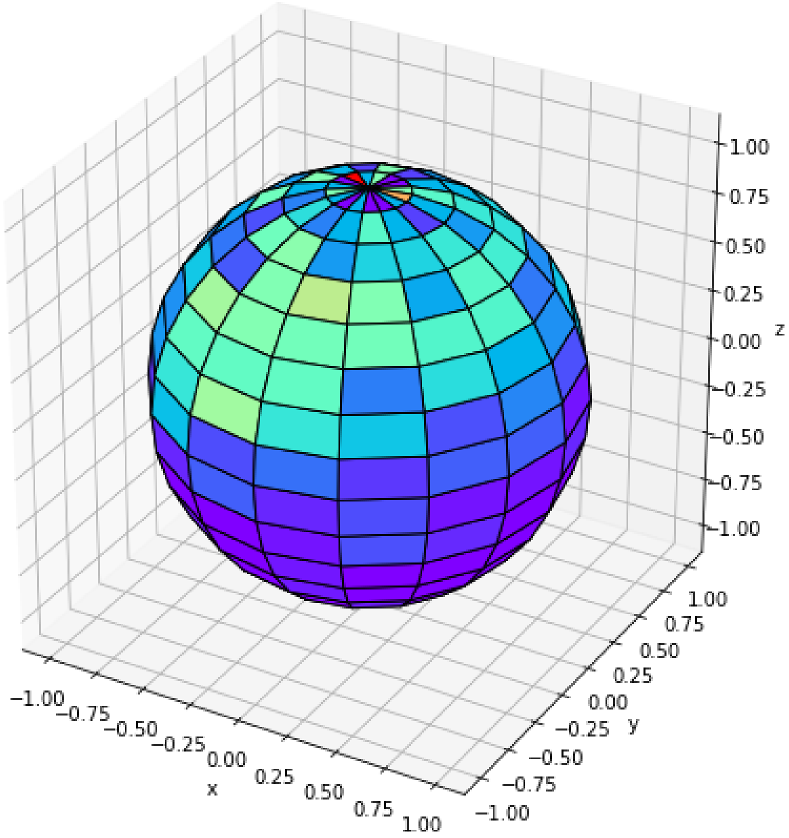





neutron fluence in cm⁻² (top), 5 MeV
x [mm]
1500
1000
500
0
−500
−1000
−2000
0
2000
4000
6000
8000
z [mm]
10⁵
10⁴
10³
10²


neutron fluence in cm⁻² (top), 20 MeV
x [mm]
1500
1000
500
0
−500
−1000
−2000
0
2000
4000
6000
8000
z [mm]
10⁵
10⁴
10³
10²

```
data = ...    # Some random x, y, z points

h = special.spherical_histogram(data)
h = h.projection("theta", "phi")

h.plot.globe_map(density=True, figsize=(7, 7), cmap="rainbow")
```



janpipek/physt

jan.pipek@gmail.com

janpipek