histbook Documentation

Release 0.0.7

Jim Pivarski

Jun 06, 2018

Contents

Versatile, high-performance histogram toolkit for Numpy.

A histogram is a way to visualize the distribution of a dataset via aggregation: rather than plotting data points individually, we count how many fall within a set of abutting intervals and plot those totals. The resulting chart is an approximate view of the distribution from which the data were derived (see Wikipedia for details).

The **histbook** package defines, fills, and visualizes histograms of Numpy data. Its capabilities extend considerably beyond the numpy.histogram function included in Numpy, as it was designed to serve the needs of particle physicists. Particle physicists have been analyzing data with histograms for decades and have strict requirements on histogramming:

- One must be able to declare an empty histogram as a container to be filled, iteratively or in parallel, and then combine results from multiple sources. An interface that skips directly from data to plot or tries to guess bin edges on the fly is not sufficient.
- It must be possible to fill many histograms in a single pass over the data, as datasets may be huge and I/O-bound.
- Data analysts must be able to access bin contents programmatically, not just visually. They will be performing statistical analyses on the contents.
- It should be possible to make "profile plots" (average one variable, binned in another) in addition to plain histograms.
- The data may be weighted, including negative weights.

CERN HBOOK was created in the 1970's to address the above. Since then, histogramming packages developed for particle physicists (PAW, mn_fit, Jas3, HippoDraw, AIDA, YODA, ROOT) have provided the same capabilities. histbook, deliberately echoing the name, does so for Numpy.

However, histbook has a more streamlined interface that allows users to be "lazy" without giving up performance. Instead of a suite of histogram and profile classes, histbook has a single n-dimensional histogram class, `Hist`. Different histograms and profiles are latent within this `Hist`, allowing data exploration after the time-consuming filling stage. Many `Hist` objects can be filled at once by binding them into a `Book`.

It's usually easier to write analysis scripts as a list of mathematical expressions, which suggests separate passes over the data, but it's much faster to execute them as a single pass. To bridge this gap, histbook takes axis specifications as *symbolic expressions* to collect in a single pass with no duplication of reading or processing. For example, if you wish to plot "`pt`", "`eta`", and "`pt*sinh(eta)`" and they're in the same `Book`, the `pt` array will be read once, the `eta` array will be read once, and they'll be reused to compute `pt*sinh(eta)` (using Numpy ufuncs). If any histograms in the same `Book` apply cuts like "`-10 <= pt*sinh(eta) < 10`", the subexpression array will be retained for that. If not, it will be deleted to minimize the memory footprint.

Thus, you can write your analysis as hundreds of mathematical expressions, without worrying about coding for performance, using a single syntax for any dimensionality. You can combine all of your histograms in a `Book` so that you have only one object to fill as you iterate through data. Since the filled distributions are n-dimensional, you can change your mind about how you want to plot them after the filling stage.

histbook lets you plot interactively with Vega-Lite, dump tables of numbers into Pandas DataFrames, and export histograms to ROOT format.

CHAPTER 1

Installation

Install histbook like any other Python package:

```
pip install histbook --user
```

or similar (use `sudo`, `virtualenv`, or `conda` if you wish).

CHAPTER 2

Strict dependencies:

- Python (2.7+, 3.4+)
- Numpy (1.8.0+)
- meta

CHAPTER 3

Recommended dependencies:

- Pandas for more convenient programmatic access to bin contents
- vega to view plots in a Jupyter notebook or vegascope to view them in a browser window without Jupyter.
- ROOT to analyze histograms in a complete statistical toolkit
- uproot to access ROOT files without the full ROOT framework

3.1 Tutorial

See the homepage for an introduction and examples.

3.2 Interactive tutorial

Coming soon on Binder.

3.3 Reference documentation

3.3.1 Histograms

histbook has only one histogram class, which can have arbitrarily many independent dimensions (binned axes) and dependent dimensions (profiles). Plotable one and two-dimensional histograms are derived by projection.

3.3.2 Books of histograms

Histograms can be collected into "books," both as a user convenience (fill all histograms in a book with a single call to `fill`) and for performance (avoid multiple passes over the data or repeated calculations).

Books of histograms behave like dicts, with access to individual histograms through square brackets (`__getitem__` and `__setitem__`).

3.3.3 Axis descriptors

Each dimension of a `Hist` is specified by an `Axis` object. There are three main categories (abstract classes): `GroupAxis`, `FixedAxis`, and `ProfileAxis`.

Concrete Axis types

The following are intended for use in `Hist` constructors.