Likelihood Preservation

Dr. Giordon Stark (on behalf of the ATLAS Collaboration)
SUSY2019
May 23rd, 2019
giordonstark.com

PyLlf
differentiable Likelihoods
Likelihood Preservation

Dr. Giordon Stark (on behalf of the ATLAS Collaboration)
SUSY2019
May 23rd, 2019

“piff”? diferentiable Likelihoods

if you can read this, you’re too close
9 years ago...

ATLAS reminded everyone that we all agreed in 2000 to publish likelihoods!

https://indico.cern.ch/event/100458/
ATLAS reminded everyone that we all agreed in 2000 to publish likelihoods!

IT'S BEEN 19 YEARS...
Overview of today’s talk

**multi-bin histogram-based statistical fits**

and how to preserve them

- HistFactory: ROOT+XML
- pyhf: Python+JSON

G. Stark | M. Feickert | L. Heinrich

THE DEVELOPERS
HistFactory

- A flexible **p.d.f template specification** for the building of statistical models from binned distributions and data
- Developed by Cranmer, Lewis, Moneta, Shibata, and Verkerke
- Widely used by the HEP community for standard model measurements and BSM searches

**Calculated using HistFactory**

HistFactory is partially independent of its implementation in ROOT

K. Cranmer
HistFactory? It’s just math!

Multiple, disjoint channels of binned distributions with multiple samples contributing to each with additional (shared?) systematics between sample estimates

- An XML specification with data stored in ROOT files — it’s been the *only implementation* of this calculation
  - Poisson p.d.f. for bins observed in all channels
  - Constraint p.d.f. (and data) for auxiliary measurements (systematics: normalization, shape, etc)
- Tied to ROOT ecosystem
- How do we scale? (No multi-threading for larger workspaces e.g. combinations)
- How do we preserve?
- What if there’s a bug in ROOT’s HistFactory implementation? No cross-check!
What else uses HistFactory?
What else uses HistFactory?

WHAT IF I TOLD YOU

YOU COULD REPRODUCE AND REINTERPRET THESE RESULTS YOURSELF?
What is pyhf? (I)

it would be useful to run statistical analysis outside of ROOT, RooFit, RooStats framework

```
pip install pyhf
```

A **python-only** (scipy, numpy) implementation of the HistFactory model + profile likelihood hypothesis tests

**For free**: a single plain-text file (JSON) specifies the entire workspace

https://diana-hep.org/pyhf/
What is pyhf? (II)

- pyhf implements all numeric operations through a thin layer of abstract n-D array operations to various **tensor algebra** backends
- Rely on industry-standard open-source libraries to gain (instantaneous) benefits in speed ups and calculations as they come out
Hello World

>>> import pyhf
>>> import pyhf.simplemodels
>>> import pyhf.utils
>>> pdf = pyhf.simplemodels.hepdata_like(signal_data=[12.,11.], ... bkg_data=[50.,52.], bkg_uncerts=[3.,7.])
>>> results = pyhf.utils.runOnePoint(1.0, [51, 48] + pdf.config.auxdata, pdf)
>>> print('Observed: {} Expected: {}'.format(results[-2], results[-1][2]))
Observed: [0.05290116] Expected: [0.06445521]

- Want to use...
  - tensorflow? pip install pyhf[tensorflow]
  - pytorch? pip install pyhf[pytorch]
  - mxnet? pip install pyhf[mxnet]

- If the JSON workspace is online, can pipe and calculate CLs instantly

$ curl http://url-to-json/workspace.json | pyhf cls
Demo (I)

- Interactive / real-time likelihood calculation and visualization with pyhf
Demo (II) — Simple CLs

JSON defining a single channel, two bin counting experiment with systematics

```json
{
  "channels": [
    {
      "name": "singlechannel",
      "samples": [
        {
          "name": "sig",
          "data": [12.0, 11.0],
          "modifiers": [{"name": "mu", "data": null, "type": "normfactor" }]
        },
        {
          "name": "bkg",
          "data": [50.0, 52.0],
          "modifiers": [{"name": "uncorr_bkguncrt", "data": [3.0, 7.0], "type": "shapesys" }]
        }
      ]
    },
    "data": {
      "singlechannel": [51.0, 48.0]
    },
    "toplvl": {
      "measurements": [{
        "config": {"poi": "mu" },
        "name": "singlechannel"
      }]
    }
  ]
}
```

$ curl -sL https://git.io/fpuyB | pyhf cls | jq .CLs_obs

0.053404965240922135
Let’s patch the pyhf JSON spec provided with a different signal and recalculate!

```json
{  
  "channels": [  
    {  
      "name": "singlechannel",  
      "samples": [  
        {  
          "name": "sig",  
          "data": [12.0, 11.0],  
          "modifiers": [{"name": "mu", "data": null, "type": "normfactor"}]
        },  
        {  
          "name": "bkg",  
          "data": [50.0, 52.0],  
          "modifiers": [{"name": "uncorr_bkguncrt", "data": [3.0, 7.0], "type": "shapesys"}]
        }
      ],  
      "data": {  
        "singlechannel": [51.0, 48.0]
      },  
      "toplvl": {  
        "measurements": [{  
          "config": {"poi": "mu"},  
          "name": "singlechannel"
        }]
      }
    }
  ],  
  "data": {
    "singlechannel": [51.0, 48.0]
  },  
  "toplvl": {
    "measurements": [{
      "config": {"poi": "mu"},
      "name": "singlechannel"
    }]
  }
}
```

```json
[{
  "op": "replace",
  "path": "/channels/0/samples/0/data",
  "value": [5.0, 6.0]
}]
```
Demo (III) — Simple Re-use

$ curl pdf.json | pyhf cls --patch patch.json

$ curl -sL https://git.io/fpuyB | pyhf cls | jq .CLs_obs
0.053404965240922135

# reinterpretation time
0.34238068407624395

Patch with JSONPatch (http://jsonpatch.com/)

- Let’s patch the pyhf JSON spec provided with a different signal and recalculate!

```json
# new_signal.json
[
  {
    "op": "replace",
    "path": "/channels/0/samples/0/data",
    "value": [5.0, 6.0]
  }
]
```
PyHF in the Wild

NuTheories 2018

Matthew Feickert @HEPfeickert

It is still incredibly exciting to see your colleagues using software you help develop to do actual physics! Thanks to @Holger_Schulz, Jessica, and Ye-Ling for using pyhf and thanks to @lukasheinrich_ and @kratsg for making this thing a reality with me. twitter.com/Holger_Schulz/ ...

Kyle Cranmer @KyleCranmer

Cool stuff! 👏 @lukasheinrich_ @HEPfeickert @pablodecm @kratsg created a pure python (with @TensorFlow & @PyTorch backends) implementation of HistFactory, a tool I originally wrote with @HerbieLewis & Akira Shibata. @diana_hep github.com/diana-hep/pyhf

Lukas Heinrich @lukasheinrich_

Paper with Jessica, Ye-Ling and @Holger_Schulz. This is the first paper that uses pyhf for reinterpretation!

arxiv.org/pdf/1810.05648…
pyhf in the wild

NuTheories 2018

[1810.05648]

It is still incredibly exciting to see your colleagues using software you help develop to do actual physics! Thanks to @Holger_Schulz, Jessica, and Ye-Ling for using pyhf and thanks to @lukasheinrich_ and @kratsg for making this thing a reality with me. twitter.com/Holger_Schulz/ ...

Cool stuff! 🎉
@lukasheinrich_ @HEPfeickert
@pablodecm @kratsg created a pure python (with @TensorFlow & @PyTorch backends) implementation of HistFactory, a tool I originally wrote with @HerbieLewis & Akira Shibata. @diana hep
github.com/diana-hep/pyhf

Paper with Jessica, Ye-Ling and @Holger_Schulz. This is the first paper that uses pyhf for reinterpretation!

arxiv.org/pdf/1810.05648...
Preserving sbottom multi-b
Conclusion

- pyhf provides **JSON specification of likelihoods**
  - plain-text format is advantageous for archivability and reusability
  - “HEPData”-friendly

- pyhf provides **bidirectional translation of likelihood specifications**
  - from ROOT workspaces to JSON: `xml2json`
  - from JSON to ROOT workspace: `json2xml` + `hist2workspace`

- pyhf provides **independent python-only implementation of HistFactory**
  - hypothesis testing
  - take advantage of industry-developed tools such as numpy and tensorflow

Connect with us on GitHub!