ATL-SOFT-PROC-2020-022
12 March 2020

Likelihood preservation and statistical reproduction of searches for new physics

Matthew Feickert[1,*], *Lukas* Heinrich[2,**], and *Giordon* Stark[3,***]

[1]University of Illinois at Urbana-Champaign, Urbana, IL, USA

[2]CERN, Geneva, Switzerland

[3]University of California Santa Cruz SCIPP, Santa Cruz, CA, USA

Abstract. Likelihoods associated with statistical fits in searches for new physics are beginning to be published by LHC experiments on HEPData. The first of these is the search for bottom-squark pair production by ATLAS. These likelihoods adhere to a specification first defined by the HistFactory p.d.f. template. This is per-se independent of its implementation in ROOT and it is useful to be able to run statistical analysis outside of the ROOT and RooStats/RooFit framework. We introduce a JSON schema that fully describes the HistFactory statistical model and is sufficient to reproduce key results from published ATLAS analyses. Using two independent implementations of the model, one in ROOT and one in pure Python, we reproduce the sbottom multi-b limits using the published likelihoods on HEPData underscoring the implementation independence and long-term viability of the archived data.

1 Introduction

Measurements in High Energy Physics (HEP) aim to determine the compatibility of observed events with theoretical predictions. The relationship between them is often formalised in a statistical *model* $f(\boldsymbol{x}|\boldsymbol{\phi})$ describing the probability of data $\boldsymbol{x}$ given model parameters $\boldsymbol{\phi}$. Given observed data, the *likelihood* $\mathcal{L}(\boldsymbol{\phi})$ then serves as the basis to test hypotheses on the parameters $\boldsymbol{\phi}$. For measurements based on binned data (*histograms*), the `HistFactory` [1] family of statistical models has been widely used for likelihood construction in both Standard Model (SM) measurements (e.g. Ref. [2]) as well as searches for new physics (e.g. Ref. [3]). A declarative, plain-text format for describing `HistFactory`-based likelihoods [4] is presented that is targeted for reinterpretation and long-term preservation in analysis data repositories such as HEPData [5].

[*]e-mail: matthew.feickert@cern.ch

[**]e-mail: lukas.heinrich@cern.ch

[***]e-mail: giordon.holtsberg.stark@cern.ch

2 HistFactory

2.1 Formalism

HistFactory statistical models — described in depth in Ref. [4] — center around the simultaneous measurement of disjoint binned distributions (*channels*) observed as event counts $\boldsymbol{n}$. For each channel, the overall expected event rate is the sum over a number of physics processes (*samples*). The sample rates may be subject to parametrised variations, both to express the effect of *free parameters* $\boldsymbol{\eta}$ and to account for systematic uncertainties as a function of *constrained parameters* $\boldsymbol{\chi}$, whose impact on the expected event rates from the nominal rates is limited by *constraint terms*. In a frequentist framework these constraint terms can be viewed as *auxiliary measurements* with additional global observable data $\boldsymbol{a}$, which paired with the channel data $\boldsymbol{n}$ completes the observation $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{a})$. The full parameter set can be partitioned into free and constrained parameters $\boldsymbol{\phi} = (\boldsymbol{\eta}, \boldsymbol{\chi})$, where a subset of the free parameters are declared *parameters of interest* (POI) $\boldsymbol{\psi}$ (e.g. the *signal strength*) and all remaining parameters as *nuisance parameters* $\boldsymbol{\theta}$.

$$f(\boldsymbol{x}|\boldsymbol{\phi}) = f(\boldsymbol{x}|\overset{\text{free}}{\overset{\downarrow}{\boldsymbol{\eta}}}, \underset{\text{constrained}}{\underset{\uparrow}{\boldsymbol{\chi}}}) = f(\boldsymbol{x}|\overset{\text{parameters of interest}}{\overset{\downarrow}{\boldsymbol{\psi}}}, \underset{\text{nuisance parameters}}{\underset{\uparrow}{\boldsymbol{\theta}}}) \tag{1}$$

The overall structure of a HistFactory probability model is then a product of the analysis-specific model term describing the measurements of the channels and the analysis-independent set of constraint terms:

$$f(\boldsymbol{n}, \boldsymbol{a}\,|\,\boldsymbol{\eta}, \boldsymbol{\chi}) = \underbrace{\prod_{c \in \text{channels}} \prod_{b \in \text{bins}_c} \text{Pois}\left(n_{cb} \,\middle|\, \nu_{cb}\left(\boldsymbol{\eta}, \boldsymbol{\chi}\right)\right)}_{\substack{\text{Simultaneous measurement}\\ \text{of multiple channels}}} \underbrace{\prod_{\chi \in \boldsymbol{\chi}} c_\chi(a_\chi|\chi)}_{\substack{\text{constraint terms}\\ \text{for “auxiliary measurements”}}}, \tag{2}$$

where within a certain integrated luminosity one observes n_{cb} events given the expected rate of events $\nu_{cb}(\boldsymbol{\eta}, \boldsymbol{\chi})$ as a function of unconstrained parameters $\boldsymbol{\eta}$ and constrained parameters $\boldsymbol{\chi}$. The latter has corresponding one-dimensional constraint terms $c_\chi(a_\chi|\chi)$ with auxiliary data a_χ constraining the parameter χ. The expected event rates ν_{cb} are defined as

$$\nu_{cb}\left(\boldsymbol{\phi}\right) = \sum_{s \in \text{samples}} \nu_{scb}\left(\boldsymbol{\eta}, \boldsymbol{\chi}\right) = \sum_{s \in \text{samples}} \underbrace{\left(\prod_{\kappa \in \boldsymbol{\kappa}} \kappa_{scb}\left(\boldsymbol{\eta}, \boldsymbol{\chi}\right)\right)}_{\text{multiplicative modifiers}} \left(\nu_{scb}^0\left(\boldsymbol{\eta}, \boldsymbol{\chi}\right) + \underbrace{\sum_{\Delta \in \boldsymbol{\Delta}} \Delta_{scb}\left(\boldsymbol{\eta}, \boldsymbol{\chi}\right)}_{\text{additive modifiers}}\right) \tag{3}$$

from constant *nominal rate* ν_{scb}^0 and a set of multiplicative and additive *rate modifiers* $\boldsymbol{\kappa}(\boldsymbol{\phi})$ and $\boldsymbol{\Delta}(\boldsymbol{\phi})$.

2.2 JSON Schema

The structure of the JSON specification of HistFactory models closely follows the original XML-based specification [1]. The JSON specification for a HistFactory *workspace* is a primary focus of Ref. [4], but a workspace can be summarised as consisting of a set of channels (an analysis region) that include samples and possible parameterised modifiers, a set of measurements (including the POI), and observations (the observed data). Listing 1 demonstrates a simple workspace representing the measurement of a single two-bin channel with two samples: a signal sample and a background sample. The signal sample has an unconstrained normalisation factor μ, while the background sample carries an uncorrelated shape systematic. The background uncertainties for the bins are 10% and 20% respectively.

```
{
    "channels": [
        { "name": "singlechannel",
          "samples": [
            { "name": "signal",
              "data": [5.0, 10.0],
              "modifiers": [ { "name": "mu", "type": "normfactor", "data": null} ]
            },
            { "name": "background",
              "data": [50.0, 60.0],
              "modifiers": [ {"name": "uncorr_bkguncrt", "type": "shapesys", "data": [5.0,12.0]} ]
            }
          ]
        }
    ],
    "observations": [
        {"name": "singlechannel", "data": [50, 60]}
    ],
    "measurements": [
        { "name": "Measurement", "config": {"poi": "mu", "parameters": []} }
    ]
}
```

Listing 1: A toy example of a 2-bin single channel workspace with two samples. The signal sample has expected event rates of 5.0 and 10.0 in each bin, while the background sample has expected event rates of 50.0 and 60.0 in each bin. An experiment provided the observed event rates of 50.0 and 60.0 for the bins in that channel. The uncorrelated shape systematic on the background has 10% and 20% uncertainties in each bin, specified as absolute uncertainties on the background sample rates. A single measurement is defined which specifies μ as the POI [4].

3 Likelihood Preservation and Result Reproduction

Through the use of the HistFactory JSON specification, the statistical model used in a search for sbottom squarks [6] with the ATLAS detector [7], based on the full Run-2 dataset using 139 fb^{-1} of proton-proton collision data, was both preserved and reproduced. The search for new physics performs hypothesis tests on a simplified model that is parameterised by the masses of the sbottom squark $\tilde{b}_1$ and the neutralinos $\tilde{\chi}_2^0, \tilde{\chi}_1^0$ and defines three separate statistical models. The full set of likelihoods for the three models is included as auxiliary material of the HEPData record of the analysis [8] for preservation and can be streamed from HEPData on demand. This is the first open publication of a full likelihood from an LHC experiment, fulfilling a proposal from the first Workshop on Confidence Limits (2000) [9].

In a demonstration of the full encapsulation of the HistFactory model in the JSON specification, a subset of the results from Ref. [6] are reproduced. The original analysis workspaces are converted to a set of XML and ROOT [10] files via RooStats [11], from which a JSON HistFactory workspace is made using the pyhf [12] libary's xml2json command-line tool. The subset of results are then reproduced using both a pyhf implementation and a ROOT implementation of the HistFactory model. pyhf implements the HistFactory model purely within the scientific Python software stack, i.e. using the scipy [13] and numpy [14] libraries. To convert from the JSON HistFactory workspace to a ROOT readable format, the pyhf json2xml command-line tool is used to convert the JSON to a set of XML and ROOT files, which are then converted into a RooFit workspace using the hist2workspace command-line tool. The full process is illustrated in Figure 1.

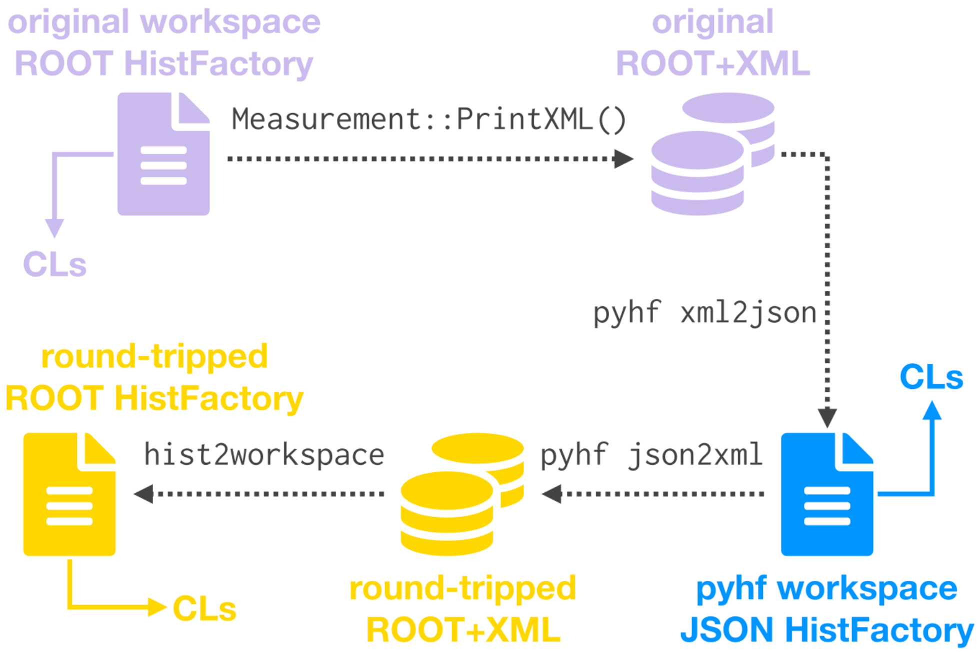

Figure 1: The diagram of the technical procedure to go from the original `ROOT` workspaces provided by the analysis team, to the JSON workspaces used by `pyhf`, and finally the round-trip `ROOT` workspaces. `pyhf` provides utilities to convert between two different `HistFactory` specification formats [4].

In both model implementations the background-only fit from Ref. [6] is reproduced as well as upper limits on the visible cross section of Beyond the Standard Model physics, with excellent agreement as detailed in Ref. [4]. Based on the single-point hypothesis test procedure at fixed $\mu = 1.0$, i.e. the nominal Beyond the Standard Model expectation, a set of tests for all simulated grid points are performed to infer a 95% CL_s exclusion contour. Using the procedure described in Figure 1, the results obtained from the archived statistical models using original `ROOT`, round-tripped `ROOT`, and `pyhf` are overlaid in Figure 2, showing excellent agreement, with only minor numerical differences, validating the completeness of the JSON `HistFactory` specification.

4 Reinterpretation

The preservation of the statistical model in a structured form also aids in the derivation of new results through the method of reinterpretation. In reinterpretations a subset of the samples contributing to the expected event rates, most commonly those associated to Beyond the Standard Model processes, are *replaced* with alternative predictions derived from a new theoretical model, while keeping the remaining estimates, typically those derived for Standard Model processes, unchanged.

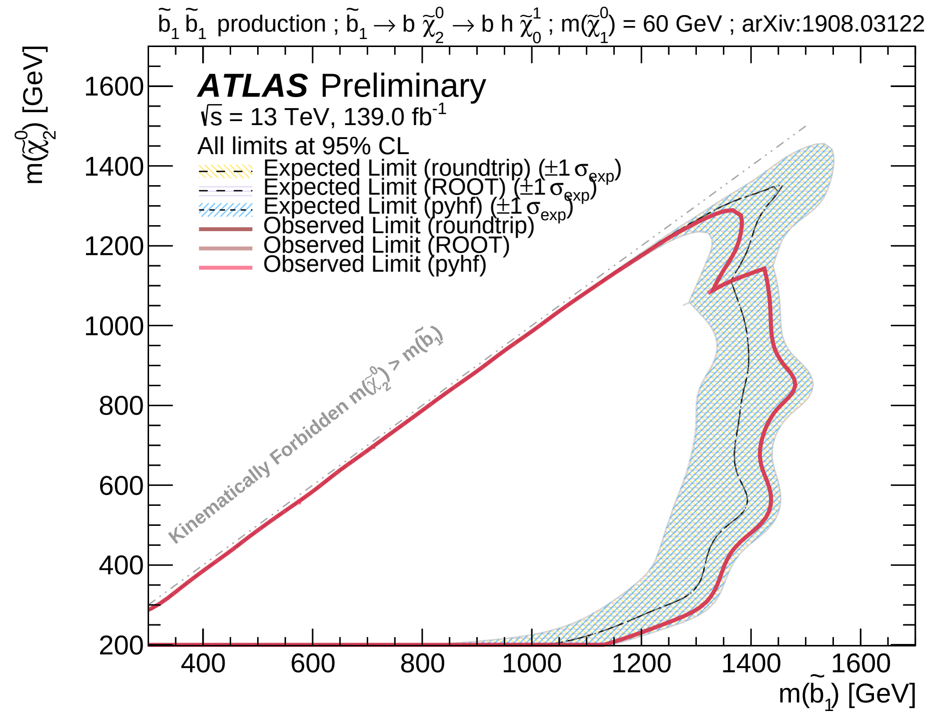

Figure 2: Exclusion contours at the 95% CL in the $m(\tilde{b}_1, \tilde{\chi}_2^0)$ phase space for the $m(\tilde{\chi}_1^0) =$ 60 GeV signal scenario using the SR with the best-expected sensitivity. The shaded band shows the impact of the theory uncertainties on the SM background, and the experimental uncertainty on both the background and the signal. The contours labeled `ROOT` are calculated from the original workspaces of the analysis. From these original workspaces, `xml2json` was run and `pyhf` was used to produce the contours labeled `pyhf`. Finally, `json2xml` was used to generate XML and `ROOT` files, from which `ROOT` workspaces can be built, to produce the contours labeled `roundtrip`. The overlaid expected and observed limits and the exclusion contours, produced by `pyhf` and `ROOT`, reproduce the contours of Figure 8(a) in Ref. [6] with excellent agreement. All curves are superimposed at the level of graphical precision [4].

The process of replacing certain samples of the original likelihood with updated ones can be viewed as *applying a patch* p to the likelihood $\mathcal{L}$ to derive a new one $\mathcal{L}'$: $\mathcal{L} \xrightarrow{p} \mathcal{L}'$. The choice of JSON as a serialisation format for the likelihood also enables an unambiguous definition of such *likelihood patches* using the JSONPatch format [15] — an ordered array of transformations applied to the original document. The patch format provides a well-defined target for reinterpretation tools to produce, when combined with the original likelihood, likelihoods for a reinterpretation. Using the 2-bin toy example in Listing 1, a JSON patch, seen in Listing 2, can be applied to replace the nominal expected event rates, an array of two floats, with new values. This patch, provided as a file `patch.json`, can be applied to the original likelihood, stored in a file `original.json`, using the `jsonpatch` command-line tool[1] which produces the result in Listing 3. The new JSON file can then be processed either through the `ROOT` implementation or the `pyhf` implementation.

[1] For example, running: `jsonpatch original.json patch.json > new.json`.

```
[{
  "op": "replace",
  "path": "/channels/0/samples/0/data",
  "value": [8.0, 3.0]
}]
```

Listing 2: A JSON patch with a single transformation to replace the nominal expected event rates [4].

```
{
    "channels": [
        { "name": "singlechannel",
          "samples": [
            { "name": "signal",
              "data": [8.0, 3.0],
              "modifiers": [ { "name": "mu", "type": "normfactor", "data": null} ]
            },
            { "name": "background",
              "data": [50.0, 60.0],
              "modifiers": [ {"name": "uncorr_bkguncrt", "type": "shapesys", "data": [5.0,12.0]} ]
            }
          ]
        }
    ],
    "observations": [
        {"name": "singlechannel", "data": [50, 60]}
    ],
    "measurements": [
        { "name": "Measurement", "config": {"poi": "mu", "parameters": []} }
    ]
}
```

Listing 3: The result of applying the JSON patch in Listing 2 to Listing 1 [4].

5 Conclusions

HistFactory statistical models are widely used for published results within HEP to model the analysis and perform statistical tests. The simple structure of HistFactory allows for easily archiving the full statistical model in a JSON format introduced in Ref. [4], which is optimised for long-term archival on data repositories such as HEPData. The ability to archive the full models from a recent search for sbottom squarks using 139 fb^{-1} of proton-proton collision data recorded with the ATLAS detector is demonstrated for the first time by an LHC experiment using the plain-text JSON specification. Finally, key statistical results of the analysis are reproduced with two independent implementations of the HistFactory model — the ROOT and Python scientific software ecosystems — underscoring the implementation independence and long-term viability of the archived data.

References

[1] K. Cranmer, G. Lewis, L. Moneta, A. Shibata, W. Verkerke, Tech. Rep. CERN-OPEN-2012-016 (2012), https://cds.cern.ch/record/1456844

[2] ATLAS Collaboration, Phys. Lett. B **726**, 88 (2013)

[3] ATLAS Collaboration, ATLAS-CONF-2018-041 (2018), https://cds.cern.ch/record/2632347

[4] ATLAS Collaboration, ATL-PHYS-PUB-2019-029 (2019), https://cds.cern.ch/record/2684863

[5] E. Maguire, L. Heinrich, G. Watt, J. Phys. Conf. Ser. **898**, 102006 (2017)

[6] ATLAS Collaboration, JHEP **12**, 060 (2019)

[7] ATLAS Collaboration, JINST **3**, S08003 (2008)

[8] ATLAS Collaboration, HEPData (2019), https://www.hepdata.net/record/ins1748602?version=1

[9] F. James, Y. Perrin, L. Lyons, eds., *Workshop on confidence limits: Proceedings* (2000), https://cds.cern.ch/record/411537

[10] R. Brun, F. Rademakers, Nucl. Inst. Meth. in Phys. Res. A **389**, 81 (1997)

[11] L. Moneta et al., PoS **ACAT2010**, 057 (2010)

[12] L. Heinrich, M. Feickert, G. Stark, *pyhf: v0.4.1* (2020), https://doi.org/10.5281/zenodo.1169739

[13] P. Virtanen et al., Nature Methods **17**, 261 (2020)

[14] S. van der Walt, S.C. Colbert, G. Varoquaux, Computing in Science Engineering **13**, 22 (2011)

[15] P.C. Bryan, M. Nottingham, RFC 6902 (2013), https://rfc-editor.org/rfc/rfc6902.txt