ATLAS PUB Note

ATL-PHYS-PUB-2019-029

21st October 2019

Reproducing searches for new physics with the ATLAS experiment through publication of full statistical likelihoods

The ATLAS Collaboration

The ATLAS Collaboration is starting to publicly provide likelihoods associated with statistical fits used in searches for new physics on HEPData. These likelihoods adhere to a specification first defined by the `HistFactory` p.d.f. template. This note introduces a JSON schema that fully describes the `HistFactory` statistical model and is sufficient to reproduce key results from published ATLAS analyses. This is per-se independent of its implementation in `ROOT` and it can be used to run statistical analysis outside of the `ROOT` and `RooStats/RooFit` framework. The first of these likelihoods published on HEPData is from a search for bottom-squark pair production. Using two independent implementations of the model, one in `ROOT` and one in pure Python, the limits on the bottom-squark mass are reproduced, underscoring the implementation independence and long-term viability of the archived data.

1 Introduction

Measurements in High Energy Physics (HEP) rely on determining the compatibility of observed collision 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]). In this note, a declarative, plain-text format for describing `HistFactory`-based likelihoods is presented that is targeted for reinterpretation and long-term preservation in analysis data repositories such as HEPData [4].

This note introduces an alternative form of the `HistFactory` specification based on the ubiquitous plain-text JSON format and its schema-language *JSON Schema*. Described in more detail in Section 2, this schema fully specifies both structure and necessary constrained data in a single document and thus is implementation independent. Section 3 demonstrates that the JSON documents describing the statistical model of analyses are sufficient to reproduce key results of the originally published analyses such as best-fit event yields, upper limits on parameters and exclusion contours. The results are reproduced in two *independent* implementations of the `HistFactory` model; one based on the `ROOT` framework and one developed within the scientific Python ecosystem. Finally, Section 4 shows how these preserved likelihoods can be used to derive new results through reinterpretation.

1.1 The `HistFactory` Formalism

Statistical models described using `HistFactory` center around the simultaneous measurement of disjoint binned distributions (*channels*) observed as event counts $\boldsymbol{n}$. For each channel, the overall expected event rate[1] 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* $\boldsymbol{\chi}$. The degree to which the latter can cause a deviation of 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 $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{a})$. In addition to the partition of the full parameter set into free and constrained parameters $\boldsymbol{\phi} = (\boldsymbol{\eta}, \boldsymbol{\chi})$, a separate partition $\boldsymbol{\phi} = (\boldsymbol{\psi}, \boldsymbol{\theta})$ will be useful in the context of hypothesis testing, where a subset of the parameters are declared *parameters of interest* (POI) $\boldsymbol{\psi}$ and the remaining ones as *nuisance parameters* $\boldsymbol{\theta}$. Often the *signal strength* of a process, the ratio of its cross section to a particular reference cross section such as that expected from the Standard Model or a given BSM scenario, is part of the parameters of interest $\boldsymbol{\psi}$ and the set of unconstrained, free parameters $\boldsymbol{\eta}$.

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

[1] Here rate refers to the number of events expected to be observed within a given data-taking interval defined through its integrated luminosity. It often appears as the input parameter to the Poisson distribution, hence the name "rate".

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

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

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 $c_\chi(a_\chi \mid \chi)$ with auxiliary data a_χ constraining the parameter χ. The expected event rates ν_{cb} are defined as

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

As shown in Equation (3), the total rates are the sum over sample rates ν_{scb}, each determined from a constant *nominal rate* ν^0_{scb} and a set of multiplicative and additive *rate modifiers* denoted $\boldsymbol{\kappa}(\boldsymbol{\phi})$ and $\boldsymbol{\Delta}(\boldsymbol{\phi})$. These modifiers are functions of model parameters, often a single parameter.

As summarised in Table 1, modifiers implementing uncertainties are paired with a corresponding default constraint term on the parameter limiting the rate modification. In some cases, the available modifiers only affect the total number of expected events of a sample within a given channel, i.e. only change its normalisation, while holding the distribution of events across the bins of a channel, i.e. its “shape”, invariant. Alternatively, modifiers may change the sample shapes. For the case of shape modifications, `HistFactory` supports correlated and uncorrelated bin-by-bin modifications. In the former, a single nuisance parameter affects the expected sample rates within the bins of a given channel, while the latter introduces one nuisance parameter for each bin, each with its own constraint term. For the correlated shape and normalisation uncertainties, `HistFactory` makes use of interpolating functions, $f_p(\alpha)$ and $g_p(\alpha)$, constructed from a small number of evaluations of the expected rate at fixed values of the nuisance parameter[2] α. For the remaining modifiers, the parameter directly affects the rate.

Given the likelihood $\mathcal{L}(\boldsymbol{\phi})$, constructed from observed data in all channels and the implied auxiliary data, *measurements* in the form of point and interval estimates can be defined. The majority of the parameters are *nuisance parameters* — parameters that are not the main target of the measurement but are necessary to correctly model the data. A small subset of the unconstrained parameters may be declared as *parameters of interest* for which measurement hypothesis tests are performed, e.g. profile likelihood methods [5]. Table 2 provides a summary of all the notation introduced in this note.

1.2 Declarative Formats

While flexible enough to describe a wide range of LHC measurements, the design of the `HistFactory` specification is sufficiently simple to admit a *declarative format* that fully encodes the statistical model of

[2] This is usually constructed from the nominal rate and measurements in the event rate at $\alpha = \pm 1$, where the value of the modifier at $\alpha = \pm 1$ must be provided and the value at $\alpha = 0$ corresponds to the corresponding identity operation of the modifier, i.e. $f_p(\alpha = 0) = 0$ and $g_p(\alpha = 0) = 1$ for additive and multiplicative modifiers respectively. See Section 4.1 in [1].

	Description	Modification	Constraint Term c_χ	Input
constrained	Uncorrelated Shape	$\kappa_{scb}(\gamma_b) = \gamma_b$	$\prod_b \text{Pois}\left(r_b = \sigma_b^{-2} \middle\vert \rho_b = \sigma_b^{-2}\gamma_b\right)$	σ_b
	Correlated Shape	$\Delta_{scb}(\alpha) = f_p\left(\alpha \middle\vert \Delta_{scb,\alpha=-1}, \Delta_{scb,\alpha=1}\right)$	$\text{Gaus}\,(a = 0 \vert\, \alpha, \sigma = 1)$	$\Delta_{scb,\alpha=\pm1}$
	Normalisation Unc.	$\kappa_{scb}(\alpha) = g_p\left(\alpha \middle\vert \kappa_{scb,\alpha=-1}, \kappa_{scb,\alpha=1}\right)$	$\text{Gaus}\,(a = 0 \vert\, \alpha, \sigma = 1)$	$\kappa_{scb,\alpha=\pm1}$
	MC Stat. Uncertainty	$\kappa_{scb}(\gamma_b) = \gamma_b$	$\prod_b \text{Gaus}\left(a_{\gamma_b} = 1 \middle\vert \gamma_b, \delta_b\right)$	$\delta_b^2 = \sum_s \delta_{sb}^2$
	Luminosity	$\kappa_{scb}(\lambda) = \lambda$	$\text{Gaus}\,(l = \lambda_0 \vert\, \lambda, \sigma_\lambda)$	$\lambda_0, \sigma_\lambda$
free	Normalisation	$\kappa_{scb}(\mu_b) = \mu_b$		
	Data-driven Shape	$\kappa_{scb}(\gamma_b) = \gamma_b$		

Table 1: Rate modifications defined in `HistFactory` for bin b, sample s, channel c. Each modifier is represented by a parameter $\phi \in \{\gamma, \alpha, \lambda, \mu\}$. By convention bin-wise parameters are denoted with γ and interpolation parameters with α. The luminosity λ and scale factors μ affect all bins equally. For constrained modifiers, the implied constraint term is given as well as the necessary input data required to construct it. σ_b corresponds to the relative uncertainty of the event rate, whereas δ_b is the event rate uncertainty in the sample relative to the total event rate $\nu_b = \sum_s \nu_{sb}^0$.

	Symbol	Name
	$f(\boldsymbol{x}\vert\boldsymbol{\phi})$	model
	$\mathcal{L}(\boldsymbol{\phi})$	likelihood
data	$\boldsymbol{x} = \{\boldsymbol{n}, \boldsymbol{a}\}$	full dataset (including auxiliary data)
	$\boldsymbol{n}$	channel data (or event counts)
	$\boldsymbol{a}$	auxiliary data
	$\nu(\boldsymbol{\phi})$	calculated event rates
parameters	$\boldsymbol{\phi} = \{\boldsymbol{\eta}, \boldsymbol{\chi}\} = \{\boldsymbol{\psi}, \boldsymbol{\theta}\}$	all parameters
	$\boldsymbol{\eta}$	free parameters
	$\boldsymbol{\chi}$	constrained parameters
	$\boldsymbol{\psi}$	parameters of interest
	$\boldsymbol{\theta}$	nuisance parameters
rate modifiers	$\boldsymbol{\kappa}(\boldsymbol{\phi})$	multiplicative rate modifier
	$\boldsymbol{\Delta}(\boldsymbol{\phi})$	additive rate modifiers
	$c_\chi(a_\chi\vert\chi)$	constraint term for constrained parameter χ
	σ_χ	relative uncertainty in the constrained parameter

Table 2: Summary of notation introduced in this paper. Many of these will be subscripted to indicate that they affect bin b of sample s of channel c. Bolded symbols indicate that they represent a set.

the analysis. This format defines the channels, all associated samples, their parameterised rate modifiers and implied constraint terms as well as the measurements. Additionally, the format represents the mathematical model, leaving the implementation of the likelihood maximisation to be analysis-dependent and/or language-dependent. Originally XML was chosen as a specification language to define the structure of the model while introducing a dependence on `ROOT` to encode the nominal rates and required input data of the constraint terms [1]. Using this specification, a model can be constructed and evaluated within the `RooFit` [6] framework.

2 JSON Schema for HistFactory

The structure of the JSON specification of `HistFactory` models follows closely the original XML-based specification [1]. This is described below with the channel specification, available modifier definitions, definition of the observed data, and specification of the measurements to perform as part of a statistical fit.

2.1 Workspace

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "$id": "https://diana-hep.org/pyhf/schemas/1.0.0/workspace.json",
    "type": "object",
    "properties": {
        "channels": { "type": "array", "items": {"$ref": "defs.json#/definitions/channel"} },
        "measurements": { "type": "array", "items": {"$ref": "defs.json#/definitions/measurement"} },
        "observations": { "type": "array", "items": {"$ref": "defs.json#/definitions/observation" } },
        "version": { "const": "1.0.0" }
    },
    "additionalProperties": false,
    "required": ["channels", "measurements", "observations", "version"]
}
```

Listing 1: [7] The workspace schema specifies three top-level properties.

The overall document in Listing 1 describes a *workspace*, which includes

channels	The channels in the model, which include a description of the samples within each channel and their possible parametrised modifiers.
measurements	A set of measurements, which define among others the parameters of interest for a given statistical analysis objective.
observations	The observed data, with which a likelihood can be constructed from the model.

A workspace consists of the channels and one set of observed data, but can include multiple measurements. If provided a JSON file, one can quickly check that it conforms to the provided workspace specification as in Listing 2.

```
import json, requests, jsonschema
workspace = json.load(open('/path/to/analysis_workspace.json'))
# if no exception is raised, it found and parsed the schema
schema = requests.get('https://diana-hep.org/pyhf/schemas/1.0.0/workspace.json').json()
# If no exception is raised by validate(), the instance is valid.
jsonschema.validate(instance=workspace, schema=schema)
```

Listing 2: Python code demonstrating how an analysis workspace file can be validated against the workspace schema.

2.2 Channel

The Channel specification consists of a list of channel descriptions. Each channel, an analysis region encompassing one or more measurement bins, consists of a `name` field and a `samples` field (see Listing 3), which holds a list of sample definitions (see Listing 4). Each sample definition in turn has a `name` field,

```
"channel": {
    "type": "object",
    "properties": {
        "name": { "type": "string" },
        "samples": { "type": "array", "items": {"$ref": "#/definitions/sample"}, "minItems": 1 }
    },
    "required": ["name", "samples"],
    "additionalProperties": false
},
```

Listing 3: [8] A channel is defined by a channel name and a list of samples.

a `data` field for the nominal event rates for all bins in the channel, and a `modifiers` field of the list of modifiers for the sample.

```
"sample": {
    "type": "object",
    "properties": {
        "name": { "type": "string" },
        "data": { "type": "array", "items": {"type": "number"}, "minItems": 1 },
        "modifiers": {
            "type": "array",
            "items": {
                "anyOf": [
                    { "$ref": "#/definitions/modifier/histosys" },
                    { "$ref": "#/definitions/modifier/lumi" },
                    { "$ref": "#/definitions/modifier/normfactor" },
                    { "$ref": "#/definitions/modifier/normsys" },
                    { "$ref": "#/definitions/modifier/shapefactor" },
                    { "$ref": "#/definitions/modifier/shapesys" },
                    { "$ref": "#/definitions/modifier/staterror" }
                ]
            }
        }
    },
    "required": ["name", "data", "modifiers"],
    "additionalProperties": false
},
```

Listing 4: [8] A sample is defined by a sample name, the sample event rate, and a list of modifiers.

2.3 Modifiers

The modifiers that are applicable for a given sample are encoded as a list of JSON objects with three fields. A name field, a type field denoting the class of the modifier, and a data field which provides the necessary input data as specified in Table 1.

Based on the declared modifiers, the set of parameters and their constraint terms are derived implicitly as each type of modifier unambiguously defines the constraint terms it requires. Correlated shape modifiers and normalisation uncertainties have compatible constraint terms and thus modifiers can be declared that *share* parameters by re-using the same name[3] for multiple modifiers. That is, a variation of a single parameter can cause a change in expected sample rates due to both shape and normalisation variations.

The structure of each modifier type is reviewed below.

[3] The name of a modifier specifies the associated parameter set such that modifiers with the same name share parameter sets.

2.3.1 Uncorrelated Shape

The absolute event rate uncertainty for each bin of the sample is stored as an array of floats. The relative uncertainties σ_b for each bin, which are used to construct the constraint term, can be computed from the absolute uncertainties using the nominal sample data yield. An example is shown in Listing 5.

```
{ "name": "mod_name", "type": "shapesys", "data": [1.0, 1.5, 2.0] }
```

Listing 5: An example of an uncorrelated shape modifier with three absolute uncertainty terms for a 3-bin channel. The `data` represent the absolute uncertainty for each bin of the sample.

2.3.2 Correlated Shape

This modifier represents a single source of uncertainty which has a different per-bin additive factor $\Delta(\alpha)$ on the various bins of a sample, hence a correlated sample shape. To implement an interpolation between sample distribution shapes, the distributions with a "downward variation"("lo") associated with $\Delta(\alpha = -1)$ and an "upward variation" ("hi") associated with $\Delta(\alpha = +1)$ are provided as arrays of floats. An example is shown in Listing 6.

```
{ "name": "mod_name", "type": "histosys", "data": {"hi_data": [20,15], "lo_data": [10, 10]} }
```

Listing 6: An example of a correlated shape modifier with absolute shape variations for a 2-bin channel. The `hi_data`(`lo_data`) are the expected event rates for the upward(downward) variations for each bin of the sample.

2.3.3 Normalisation Uncertainty

The normalisation uncertainty modifies the sample rate by an overall factor $\kappa(\alpha)$ constructed as the interpolation between downward ("lo") and upward ("hi") as well as the nominal setting, i.e. $\kappa(-1) = \kappa_{\alpha=-1}$, $\kappa(0) = 1$ and $\kappa(+1) = \kappa_{\alpha=+1}$. In the modifier definition, $\kappa_{\alpha=+1}$ and $\kappa_{\alpha=-1}$ are stored as floats. An example is shown in Listing 7.

```
{ "name": "mod_name", "type": "normsys", "data": {"hi": 1.1, "lo": 0.9} }
```

Listing 7: An example of a normalisation uncertainty modifier with scale factors recorded for the up/down variations of an n-bin channel. The `data` represent the absolute upward/downward variations in uncertainty of the overall normalisation for all bins of the sample.

2.3.4 Monte Carlo (MC) Statistical Uncertainty

As the sample counts are often derived from MC datasets, they necessarily carry an uncertainty due to the finite sample size of the datasets. As explained in detail in Ref. [1], adding uncertainties for each sample would yield a very large number of nuisance parameters with limited utility. Therefore a set of bin-wise scale factors γ_b is introduced to model the overall uncertainty in the bin due to limited MC statistics. The constraint term is constructed as a set of Gaussian distributions with a central value equal to unity for each bin in the channel. The scales σ_b of the constraint are computed from the individual uncertainties of samples defined within the channel relative to the total event rate of all samples: $\delta_{csb} = \sigma_{csb} / \sum_s \nu^0_{scb}$.

All samples within a channel are estimated from MC simulations. Only the samples with a declared MC statistical uncertainty modifier enter the sum. In this declaration, the absolute bin-wise uncertainty σ_{csb} of the channel is provided under the `data` property as a list of floating point numbers. An example is shown in Listing 8.

```
{ "name": "mod_name", "type": "staterror", "data": [0.1, 0.0, 0.2] }
```

Listing 8: An example of a statistical uncertainty modifier with absolute bin-by-bin uncertainties recorded for a 3-bin channel.

2.3.5 Luminosity

Sample rates derived from theory calculations, as opposed to data-driven estimates, are scaled to the integrated luminosity corresponding to the observed data. As the luminosity measurement is itself subject to an uncertainty, it must be reflected in the rate estimates of such samples. As this modifier is of global nature, no additional per-sample information is required and thus the data field is nulled. This uncertainty is relevant, in particular, when the parameter of interest is a signal cross section. The luminosity uncertainty σ_λ is provided as part of the parameter configuration included in the measurement specification discussed in Section 2.5. An example is shown in Listing 9.

```
{ "name": "mod_name", "type": "lumi", "data": null }
```

Listing 9: An example of a luminosity modifier.

2.3.6 Unconstrained Normalisation

The unconstrained normalisation modifier scales the event rates of a sample by a free parameter μ. Common use cases are the signal rate of a possible BSM signal or simultaneous in-situ measurements of background sample rates (e.g. scaling a background using a control region). Such parameters are frequently the parameters of interest of a given measurement. No additional per-sample data is required. An example is shown in Listing 10.

```
{ "name": "mod_name", "type": "normfactor", "data": null }
```

Listing 10: An example of a normalisation modifier.

2.3.7 Data-driven Shape

In order to support data-driven estimation of sample rates (e.g. for multijet backgrounds), the data-driven shape modifier adds free, bin-wise multiplicative parameters. Similarly to the normalisation factors, no additional data is required as no constraint is defined. An example is shown in Listing 11.

```
{ "name": "mod_name", "type": "shapefactor", "data": null }
```

Listing 11: An example of an uncorrelated shape modifier.

2.4 Observations

The observations provided by the analysis are the observed data for each channel (or region). These data are provided as an array mapping the channel name to an array of floats, which provide the observed rates in each bin of the channel. The auxiliary data are not included as they are an input to the likelihood that does not need to be archived and can be determined automatically from the specification.

An example is shown in Listing 12.

```
{ "observations": [
  {"name": "chan_name_one", "data": [1.0, 2.0, 3.0]},
  {"name": "chan_name_two", "data": [2.0, 0.0, 1.0]}
]}
```

Listing 12: An example of channel data for two 3-bin channels.

2.5 Measurements

Given the data and the model definitions, a measurement can be defined. In the current schema, the measurement defines the name of the parameter of interest as well as parameter set configurations[4]. Here, the remaining information not covered through the channel definition is provided, e.g. for the luminosity parameter. For all modifiers, the default settings can be overridden where possible:

inits	Initial value of the parameter.
bounds	Interval bounds of the parameter.
auxdata	Auxiliary data for the associated constraint term.
sigmas	Associated uncertainty of the parameter.
fixed	Boolean flag indicating whether or not the parameter is set constant.

An example is shown in Listing 13.

```
{
    "name": "MyMeasurement",
    "config": {
        "poi": "SignalCrossSection", "parameters": [
            { "name":"lumi", "auxdata":[1.0],"sigmas":[0.017], "bounds":[[0.915,1.085]],"inits":[1.0] },
            { "name":"mu_ttbar", "bounds":[[0, 5]] },
            { "name":"rw_1CR", "fixed":true }
        ]
    }
}
```

Listing 13: An example of a measurement. This measurement, which scans over the parameter of interest `SignalCrossSection`, is setting configurations for the luminosity modifier, changing the default bounds for the normfactor modifier named `mu_ttbar`, and specifying that the modifier `rw_1CR` is held constant (`fixed`).

[4] In this context a parameter set corresponds to a named lower-dimensional subspace of the full parameters $\boldsymbol{\phi}$. In many cases these are one-dimensional subspaces, e.g. a specific interpolation parameter α or the luminosity parameter λ. For multi-bin channels, however, e.g. all bin-wise nuisance parameters of the uncorrelated shape modifiers are grouped under a single name. In general, a parameter set definition provides arrays of initial values, bounds, etc.

2.6 Toy example

Listing 14 demonstrates a simple 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 controlled by parameters γ_1 and γ_2. 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 14: 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.

3 Search for Sbottom Squarks

This section demonstrates the use of the `HistFactory` JSON schema to preserve the statistical model of ATLAS analyses in the context of a recent search for sbottom squarks based on the full Run-2 dataset using 139 fb^{-1} of proton-proton collision data [9]. 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 $\tilde{\chi}_2^0$, $\tilde{\chi}_1^0$. The search defines three separate statistical models labeled A, B, and C. Within each model all channels correspond to disjoint regions[5] in phase space.

Model A Two channels are defined: A control channel `CRtt_meff` and a signal channel `SRs_meff`. Both channels have three bins.

Model B Two single-bin channels are defined: `CRtt_cuts` and `SR_cuts`.

Model C Three channels are defined: Two control regions (`CRtt_cuts` with three bins, `CRz_cuts` with one bin) and a signal region `SR_metsigST` channel with four bins.

[5] *Region* and *Channel* are often used interchangeably among experimental particle physicists.

As no significant excess has been observed, exclusion hypothesis tests are performed for each model on a grid of points defined by the two-dimensional parameter space of the simplified model to infer a set of expected and observed CL_s values [10]. Based on the hypothesis tests results, a 95% CL_s exclusion contour has been derived delineating the parameter sub-space excluded by the analysis. Here for each point, the CL_s result of the statistical model with the most stringent expected limit is taken.

3.1 Preserved Likelihood

The full set of likelihoods for the three models is included as auxiliary material of the the HepData record of the analysis [11]. In Listing 15, an abridged structure of the workspace is shown, leaving out the modifier specifications.


```
{
  "channels": [
  {
    "name": "CRtt_meff",
    "samples": [
      {"data": [0.44396111369132996, 0.549005925655365, 0.2581719756126404], "name": "diboson"},
      {"data": [3.6903815269470215, 1.1958770751953125, 1.0939558744430542], "name": "W"},
      {"data": [0.22261416912078857, 0.04251996800303459, 0.0], "name": "Z"},
      {"data": [3.7213690280914307, 1.1334068775177002, 0.26062288880348206], "name": "ttZ"},
      {"data": [0.9883568286895752, 0.8140284419059753, 0.346008837223053], "name": "ttW"},
      {"data": [8.139641761779785, 4.380062580108643, 2.243513584136963], "name": "st"},
      {"data": [118.6123046875, 51.03012466430664, 20.230464935302734], "name": "ttbar"},
      {"data": [8.659423828125, 2.7713887691497803, 0.6784200668334961], "name": "ttH"},
      {"data": [0.46871495246887207, 2.207951784133911, 1.8667230606079102], "name": "sbottom_1000_131_1"}
    ]
  },
  {
    "name": "SR_meff",
    "samples": [
      {"data": [0.3429398238658905, 0.0, 0.04357590526342392], "name": "diboson"},
      {"data": [0.2169857919216156, 0.13739198446273804, 0.0430174358189106], "name": "W"},
      {"data": [1.2066327333450317, 0.8350478410720825, 0.4176981449127197], "name": "Z"},
      {"data": [0.6686786413192749, 0.25316545367240906, 0.10552376508712769], "name": "ttZ"},
      {"data": [0.06484010815620422, 0.08128178119659424, 0.009203530848026276], "name": "ttW"},
      {"data": [0.4294568598270416, 0.33714672923088074, 0.5907787084579468], "name": "st"},
      {"data": [6.269027233123779, 4.050149440765381, 1.817328929901123], "name": "ttbar"},
      {"data": [0.6425122618675232, 0.3274693489074707, 0.07939066737890244], "name": "ttH"},
      {"data": [0.008575495332479477, 0.13540373742580414, 0.18992283940315247], "name": "sbottom_1000_131_1"}
    ]
  }
  ],
  "measurements": [
    {
      "config": {
        "parameters": [
          {"sigmas": [0.017], "inits": [1.0 ], "auxdata": [1.0],  "name": "lumi", "bounds": [[0.915,1.085]]
          }
        ],
        "poi": "mu_SIG"
      },
      "name": "NormalMeasurement"
    }
  ],
  "observations": [ {"name": "SR_meff", "data": [12.0, 3.0, 2.0]},  {"name": "CRtt_meff", "data": [153.0, 52.0, 19.0]} ]
}
```


Listing 15: A slimmed down specification of a likelihood in Region A of the analysis for the $\left(\tilde{b}_1, \tilde{\chi}_2^0, \tilde{\chi}_1^0\right) = (1000, 131, 1)$ GeV point. The modifier specifications (and schema version) are left out for readability.

3.2 Result Reproduction

The original workspaces are converted to a set of `XML` and `ROOT` [12] files via `RooStats` [13]. From these files the `pyhf xml2json` command-line tool produces a JSON `HistFactory` workspace.

Based on the set of JSON documents defining the search grid, a subset of results from the original analysis are reproduced, using two implementations of the `HistFactory` model.

`pyhf` Implementation The `pyhf` [14] package implements the mathematical model of `HistFactory` purely within the scientific Python software stack, i.e. using the `scipy` [15] and `numpy` [16] libraries.

round-trip `ROOT` Implementation In this approach, the JSON workspace is converted into a set of `XML` and `ROOT` files using the `json2xml` subcommand of the `pyhf` command line tool. From these files the `hist2workspace` command-line tool produces a `RooFit` workspace, from which hypothesis tests can be performed using the `RooStats` interfaces.

This section compares these two sets of results, both of which are shown to agree well with the original results of the analysis. Figure 1 highlights the procedure to generate the workspaces for the implementations described below.

3.2.1 Background-only Fit

In the search for sbottom squarks a *background-only* fit is performed. This is a fit where the POI is held constant at zero and the rest of the nuisance parameters are allowed to float. Here, only a subset of the channels, the *control regions*, enter the likelihood that is subsequently maximised to infer the best-fit values of the parameters $\boldsymbol{\phi}$. In this fit all parameters that give rise to non-zero expected rates for signal samples, usually the signal strength that is the POI of the model, are fixed to zero. With these parameter values, background yields in the remaining channels can be computed. The event yields in the signal regions of all three models are shown in Table 3. The agreement is excellent and the size of the residual differences are largely due to numerical effects (e.g. p.d.f. approximation choices in `RooFit`). In addition, the best fit values under both frameworks and their post-fit uncertainties are shown in Figure 2 for Model A. Here, excellent agreement not only in the parameter values but also their uncertainties is observed as well.

3.2.2 Upper Limits

Based on the statistical models, hypothesis tests can be performed to determine the compatibility of the data with and without a Beyond the Standard Model component. The tests are performed using the asymptotic formulae of the profile likelihood-based test statistic $\tilde{q}_\mu$ [5]. The results of these tests are provided as upper limits on the visible cross section of Beyond the Standard Model physics. The results obtained from `ROOT` and `pyhf` are shown in Figure 3.

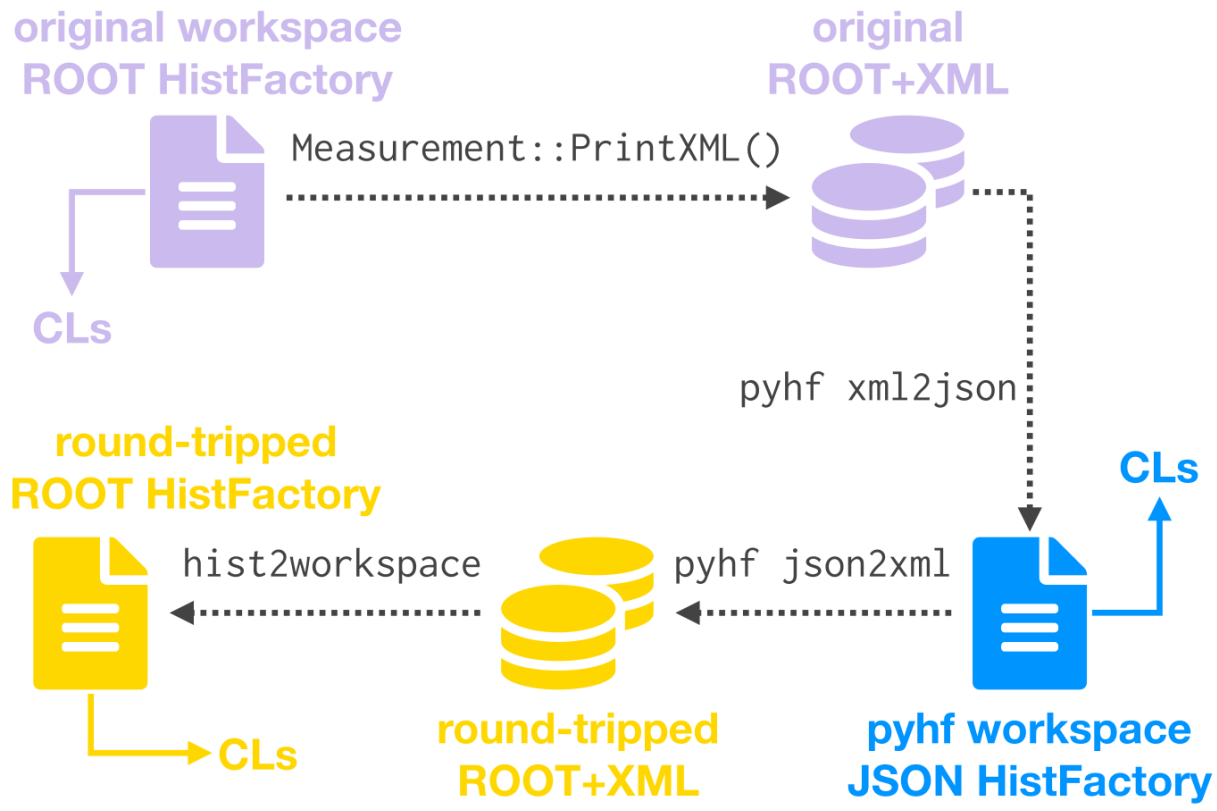

Figure 1: The diagram of the technical procedure to go from the original ROOT workspaces provided by the analysis team, to the JSON workspaces used by `pyhf`, and finally the round-trip ROOT workspaces. `pyhf` provides utilities to convert between two different `HistFactory` specification formats.

3.2.3 Exclusion Contours

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 can be performed to infer a 95% CL_s exclusion contour. `pyhf` ships with utilities that allow a conversion between ROOT `HistFactory` workspaces and python `HistFactory` workspaces: `pyhf xml2json` and `pyhf json2xml`. 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 4. This shows excellent agreement as well, with minor numerical differences. Additionally, it validates the completeness of the JSON `HistFactory` specification.

4 Reinterpretation

Beyond the reproduction of the original results, the preservation of the statistical model in a structured form also aids in the derivation of new results through the method of reinterpretation. In the reinterpretation of analyses a subset of the samples contributing to the expected event rates, most commonly those associated

	Model A			Model B	Model C			
	SRA-L	SRA-M	SRA-H	SRB	SRC22	SRC24	SRC26	SRC28
Observed events	12.00	3.00	2.00	3.00	28.00	12.00	4.00	3.00
Fitted SM bkg events	8.35	5.66	3.01	3.29	20.87	10.29	3.95	2.45
$t\bar{t}$	4.77	3.69	1.73	2.31	3.89	1.08	0.34	0.12
Z+jets	1.21	0.84	0.41	0.28	8.50	5.73	1.92	1.08
Single+top	0.43	0.33	0.59	0.48	2.70	1.21	0.68	0.44
$t\bar{t} + W/Z$	0.73	0.33	0.12	0.08	2.52	1.01	0.52	0.25
$t\bar{t} + h$	0.65	0.33	0.08	0.12	0.16	0.04	0.08	0.00
W+jets	0.22	0.13	0.04	0.02	2.16	0.63	0.24	0.42
Diboson	0.34	0.00	0.04	0.00	0.94	0.58	0.17	0.13

(a) `ROOT`

	Model A			Model B	Model C			
	SRA-L	SRA-M	SRA-H	SRB	SRC22	SRC24	SRC26	SRC28
Observed events	12.00	3.00	2.00	3.00	28.00	12.00	4.00	3.00
Fitted SM bkg events	8.37	5.66	3.01	3.30	20.85	10.28	3.95	2.45
$t\bar{t}$	4.79	3.70	1.73	2.31	3.88	1.08	0.34	0.12
Z+jets	1.20	0.84	0.41	0.28	8.49	5.72	1.92	1.08
Single+top	0.43	0.33	0.58	0.48	2.71	1.22	0.68	0.44
$t\bar{t} + W/Z$	0.73	0.33	0.12	0.08	2.52	1.01	0.52	0.25
$t\bar{t} + h$	0.65	0.33	0.08	0.12	0.16	0.04	0.08	0.00
W+jets	0.22	0.13	0.04	0.02	2.16	0.63	0.24	0.42
Diboson	0.34	0.00	0.04	0.00	0.94	0.59	0.17	0.13

(b) `pyhf`

Table 3: Comparison of background-only fit results between (a) `ROOT` and (b) `pyhf`. The results correspond to Table 6 in [9] and use its region definitions.

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. Reinterpretation is efficient when the resources to compute the former are small compared to those required to derive the latter. New rate estimates for additional physics processes can be derived through either re-executing the original analysis as forseen in RECAST [17] or approximate re-implementations implemented in third-party frameworks.

4.1 Likelihood Patches

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 [18] — 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.

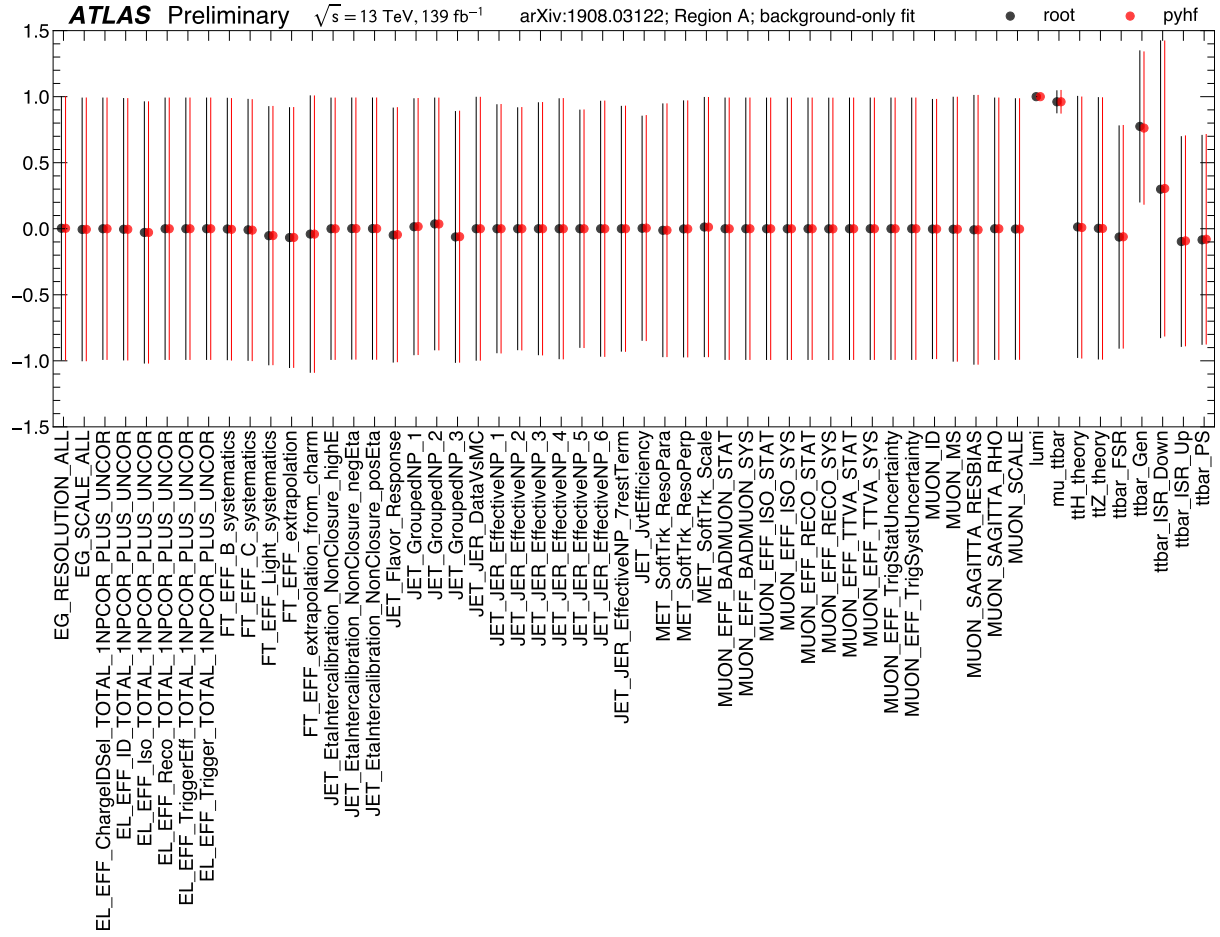

Figure 2: Best-fit parameter values and their uncertainties for the background only fit of the ATLAS sbottom search Model A likelihood. Analysis specific nuisance parameters are described in Ref. [9] and systematic variations are described in public notes from the ATLAS combined performance groups.

4.1.1 Example

This sub-section demonstrates the use of JSON patches in the context of `HistFactory` JSON documents. Using the 2-bin toy example in Listing 14, a JSON patch can be applied to replace the nominal expected event rates, an array of two floats, with new values. This can be seen in Listing 16.

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

Listing 16: A JSON patch with a single transformation to replace the nominal expected event rates.

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[6] which produces the result in Listing 17.

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

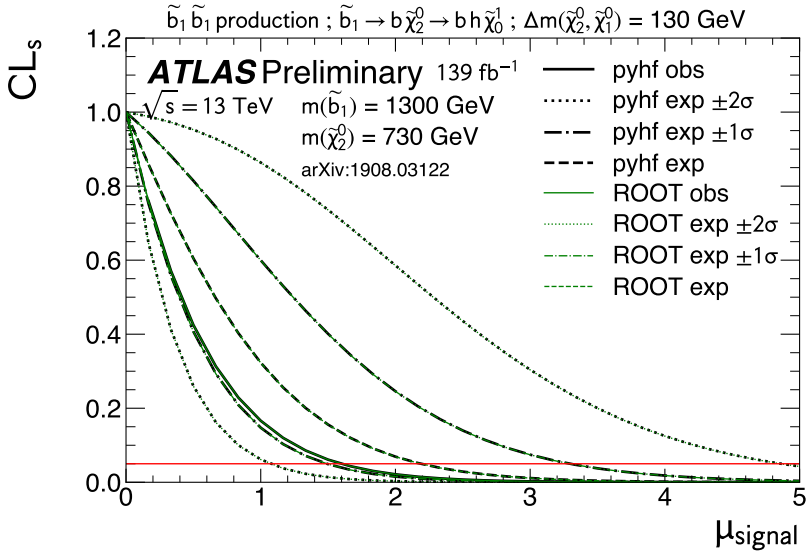

Figure 3: Scan of CL$_s$ for a range of fixed signal strengths μ. Shown are the observed CL$_s$ values as well as those expected of datasets with test-statistic values corresponding to various percentiles of test-statistic distribution of the background-only hypothesis.

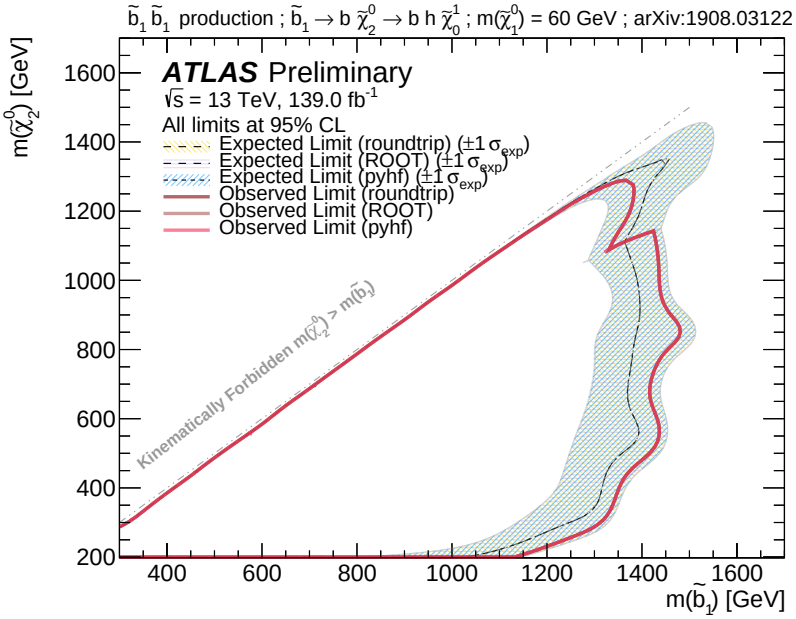

Figure 4: Exclusion contours at the 95% CL in the $m(\tilde{b}_1, \tilde{\chi}_2^0)$ phase space for the $m(\tilde{\chi}_1^0)$ = 60 GeV signal scenario using the SR with the best-expected sensitivity. The shaded band shows the impact of the theory uncertainties on the SM background, and the experimental 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 exclusion contours, produced by `pyhf` and `ROOT`, reproduce the contours of Figure 8(a) in Ref. [9]. All curves are superimposed at the level of graphical precision.

```
{
    "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 17: The result of applying the JSON patch in Listing 16 to Listing 14.

The new JSON file can then be processed either through the `ROOT` implementation or the `pyhf` implementation.

5 Conclusions

A large number of results published within HEP use a single family of statistical models — `HistFactory` — to model the analysis and perform statistical tests. The simple structure of `HistFactory` allows for an archiving of the full statistical model in a JSON format which has been introduced in this note, which is optimised for long-term archival on data repositories such as HEPData. This note demonstrates the ability to archive the models from a recent search for sbottom squarks using 139 fb^{-1} of proton-proton collision data recorded with the ATLAS detector using the plain-text JSON specifications introduced in this note. 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 and W. Verkerke, *HistFactory: A tool for creating statistical models for use with RooFit and RooStats*, tech. rep. CERN-OPEN-2012-016, New York U., 2012, URL: https://cds.cern.ch/record/1456844.

[2] ATLAS Collaboration, *Measurements of Higgs boson production and couplings in diboson final states with the ATLAS detector at the LHC*, Phys. Lett. B **726** (2013) 88, arXiv: 1307.1427 [hep-ex], Erratum: Phys. Lett. B **734** (2014) 406.

[3] ATLAS Collaboration, *Search for supersymmetry in final states with missing transverse momentum and multiple b-jets in proton–proton collisions at* $\sqrt{s} = 13$ *TeV with the ATLAS detector*, ATLAS-CONF-2018-041, 2018, URL: https://cds.cern.ch/record/2632347.

[4] E. Maguire, L. Heinrich and G. Watt, *HEPData: a repository for high energy physics data*, J. Phys. Conf. Ser. **898** (2017) 102006, arXiv: 1704.05473 [hep-ex].

[5] G. Cowan, K. Cranmer, E. Gross and O. Vitells, *Asymptotic formulae for likelihood-based tests of new physics*, Eur. Phys. J. C **71** (2011) 1554, arXiv: 1007.1727 [physics.data-an], Erratum: Eur. Phys. J. C **73** (2013) 2501.

[6] W. Verkerke and D. Kirkby, *The RooFit toolkit for data modeling*, 2003, arXiv: physics/0306116 [physics.data-an].

[7] *Workspace Schema, v1.0.0*, https://diana-hep.org/pyhf/schemas/1.0.0/workspace.json, Accessed: 2019-07-01.

[8] *Base Definitions Schema, v1.0.0*, https://diana-hep.org/pyhf/schemas/1.0.0/defs.json, Accessed: 2019-07-01.

[9] ATLAS Collaboration, *Search for bottom-squark pair production with the ATLAS detector in final states containing Higgs bosons, b-jets and missing transverse momentum*, (2019), arXiv: 1908.03122 [hep-ex].

[10] A. L. Read, *Presentation of search results: the* CL_S *technique*, J. Phys. G **28** (2002) 2693.

[11] *Search for bottom-squark pair production with the ATLAS detector in final states containing Higgs bosons, b-jets and missing transverse momentum*, Data Collection, 2019, URL: https://www.hepdata.net/record/ins1748602?version=1.

[12] R. Brun and F. Rademakers, *ROOT — An object oriented data analysis framework*, Nucl. Inst. Meth. A: Accelerators, Spectrometers, Detectors and Associated Equipment (1997) 81, New Computing Techniques in Physics Research V, URL: http://www.sciencedirect.com/science/article/pii/S016890029700048X.

[13] L. Moneta et al., *The RooStats Project (proceedings)*, PoS **ACAT2010** (2010) 057, arXiv: 1009.1003 [physics.data-an].

[14] L. Heinrich, M. Feickert, G. Stark and K. Cranmer, *diana-hep/pyhf: v0.1.2*, 2019, URL: https://doi.org/10.5281/zenodo.3334365.

[15] E. Jones, T. Oliphant, P. Peterson et al., *SciPy: Open source scientific tools for Python*, [Online; accessed 24.04.2019], 2001–, URL: http://www.scipy.org/.

[16] S. van der Walt, S. C. Colbert and G. Varoquaux, *The NumPy Array: A Structure for Efficient Numerical Computation*, Computing in Science Engineering **13** (2011) 22, ISSN: 1521-9615.

[17] K. Cranmer and I. Yavin, *RECAST — extending the impact of existing analyses*, Journal of High Energy Physics **2011**, 38 (2011) 38, arXiv: 1010.2506 [hep-ex].

[18] P. C. Bryan and M. Nottingham, *JavaScript Object Notation (JSON) Patch*, RFC 6902, 2013, URL: https://rfc-editor.org/rfc/rfc6902.txt.