technische universität
dortmund

# HS$^3$

**Carsten Burgard**

Reinterpretation of LHC results for new physics

9th general workshop of the BSM Reinterpretation Forum

25-28 February 2025, CERN

# What is HS³?

- What is HS³?
  - HS³ is the **High Energy Physics Statistics Serialization Standard**
  - an open data standard for statistical models
- What is a statistical model?
  - a mathematical framework describing relationships between variables in a dataset
  - typically using probability distributions and parameters
- What is a statistical model comprised of?
  - at its core, one or more **probability distributions** depending on a set of **parameters**
  - optionally, any of the following
    - **functions and distributions** that serve as building blocks
    - one or more **datasets** (observed or simulated) to compare to
    - **analysis prescriptions**, e.g. which dataset goes with which distribution, which parameters are POIs, which are NPs, and what external constraints exist
    - **parameter domains** and specific **parameter points** of interest (e.g. SM values)

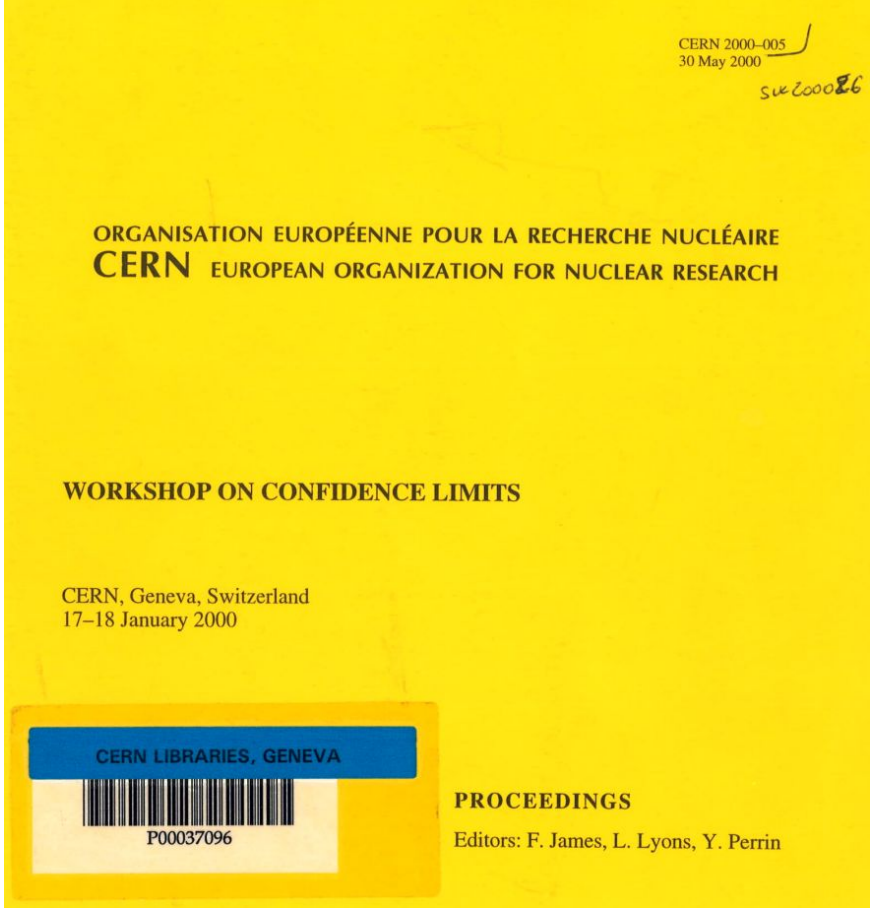
# Why do we need this?

**Massimo Corradi**

It seems to me that there is a general consensus that what is really meaningful for an experiment is likelihood, and almost everybody would agree on the prescription that experiments should give their likelihood function for these kinds of results. Does everybody agree on this statement, to publish likelihoods?

**Louis Lyons**

Any disagreement? Carried unanimously. That's actually quite an achievement for this Workshop.






*this slide was stolen from Lukas Heinrich & Kyle Cranmer*

# Publications as a channel of communication

- As a science community, we communicate our results via publications
- Different “interfaces” of data being published:
  - low-level data: lists of events (e.g. ATLAS OpenData)
  - Very high-level data: numbers shown in plots & tables in a machine-readable form ❤️ HEPData
- Both of these are far away from being a complete representation of our statistical model
- Sometimes publish “simplified representation”
  - unfolded distributions
  - multivariate gaussian models
  - …
- Very rarely, full statistical models are published in earnest!

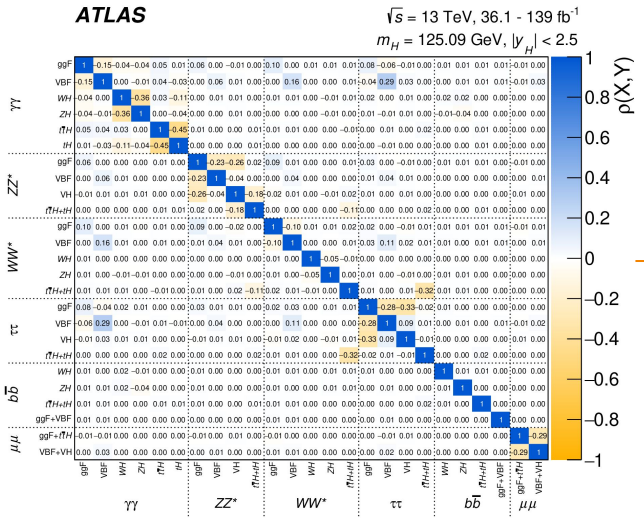
**Disclaimer**

I’m an ATLAS guy
mileage may vary

# Simplified Models: Multivariate Gaussians

- The simplest method: Multivariate Gaussians

$$P(x,\mu) = \exp\left(\frac{1}{2}(\vec{x}-\vec{\mu})C(\vec{x}-\vec{\mu})\right)$$

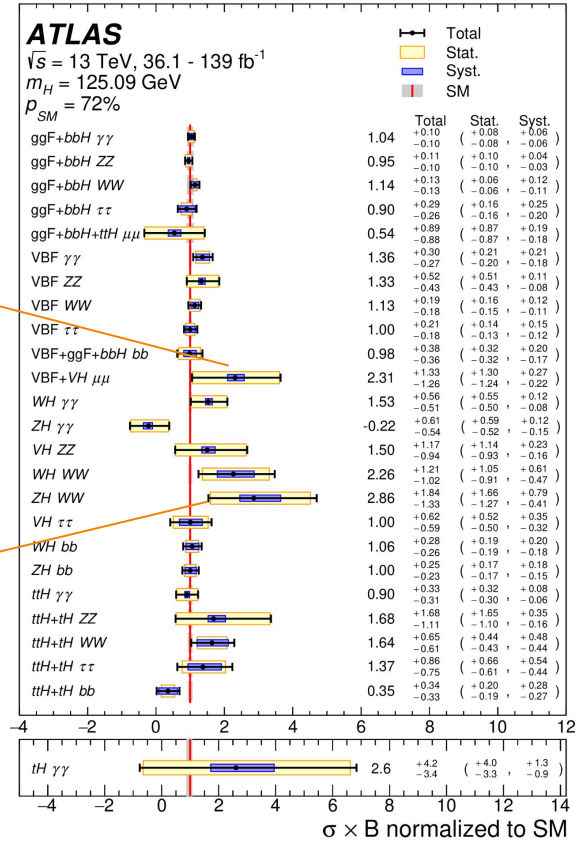



covariance matrix

correlations

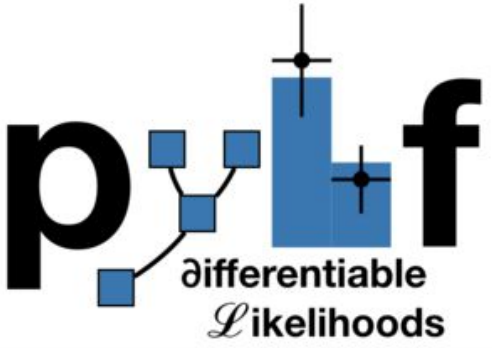
central values

uncertainties



# Public likelihoods *powered by pyhf*: A success story

- Since about 2018, pyhf exists as a pure-python implementation of the HistFactory modeling framework
  - very useful for stacks of histograms
  - somewhat limited in scope (binned fits, finite list of “modifiers”)
  - adopted by large parts of ATLAS (mostly SUSY+Exotics)
- Models can be written down descriptively as JSON files
  - resounding adoption by the theory community




```
{
  "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 ] }
          ]
        }
      ]
    }
  ]
}
```

*Purely descriptive, static model representations are extremely useful*

# pyhf and its influence on likelihood publishing

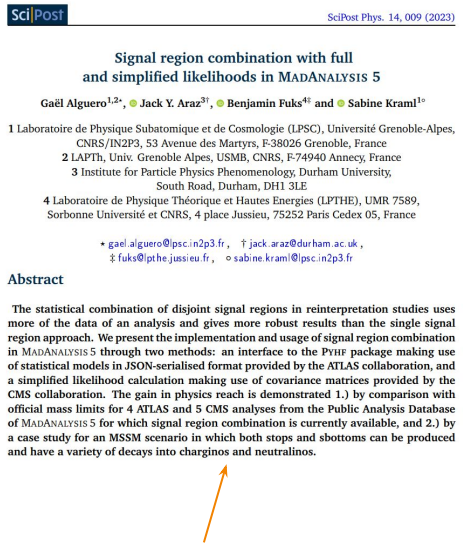
- text-models have been published on HEPdata
  - only for pyhf models so far
  - possible for ROOT workspaces since very recently
- these results have been enthusiastically embraced by theorists

Look out for this button here!

There is significant tooling available around pyhf & JSON:

- cabinetry: build & use pyhf models
- workspaceexplorer: examine pyhf models
- also a julia version available: LiteHF.jl

*... but not everyone uses pyhf!*



SciPost Phys. 14, 009 (2023)

**Signal region combination with full and simplified likelihoods in MadAnalysis 5**

Gaël Alguero[1,2⋆], Jack Y. Araz[3†], Benjamin Fuks[4‡] and Sabine Kraml[1○]

1 Laboratoire de Physique Subatomique et de Cosmologie (LPSC), Université Grenoble-Alpes, CNRS/IN2P3, 53 Avenue des Martyrs, F-38026 Grenoble, France
2 LAPTh, Univ. Grenoble Alpes, USMB, CNRS, F-74940 Annecy, France
3 Institute for Particle Physics Phenomenology, Durham University, South Road, Durham, DH1 3LE
4 Laboratoire de Physique Théorique et Hautes Energies (LPTHE), UMR 7589, Sorbonne Université et CNRS, 4 place Jussieu, 75252 Paris Cedex 05, France

⋆ gael.alguero@lpsc.in2p3.fr, † jack.araz@durham.ac.uk, ‡ fuks@lpthe.jussieu.fr, ○ sabine.kraml@lpsc.in2p3.fr

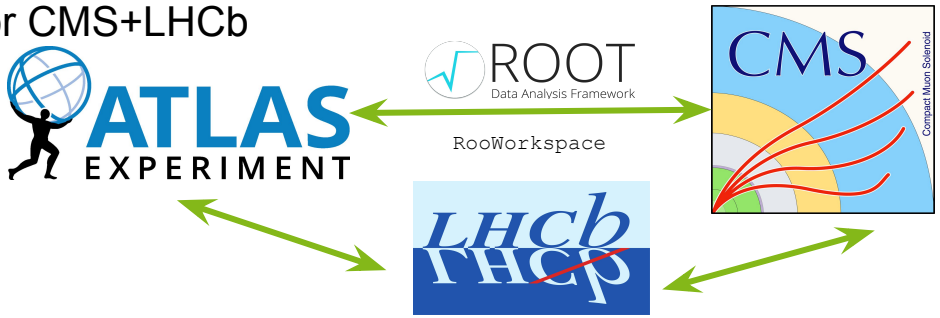
**Abstract**

The statistical combination of disjoint signal regions in reinterpretation studies uses more of the data of an analysis and gives more robust results than the single signal region approach. We present the implementation and usage of signal region combination in MadAnalysis 5 through two methods: an interface to the Pyhf package making use of statistical models in JSON-serialised format provided by the ATLAS collaboration, and a simplified likelihood calculation making use of covariance matrices provided by the CMS collaboration. The gain in physics reach is demonstrated 1.) by comparison with official mass limits for 4 ATLAS and 5 CMS analyses from the Public Analysis Database of MadAnalysis 5 for which signal region combination is currently available, and 2.) by a case study for an MSSM scenario in which both stops and sbottoms can be produced and have a variety of decays into charginos and neutralinos.

Paper combining full likelihoods published by ATLAS with simplified models published by CMS

# RooFit and the LHC

- RooFit was originally developed for BaBar
  - has been extremely successful backend for statistical modeling for 2 decades
- The RooWorkspace is the de-facto standard format for exchanging statistical models
  - for example: CMS+ATLAS combination efforts
  - also used for CMS+LHCb

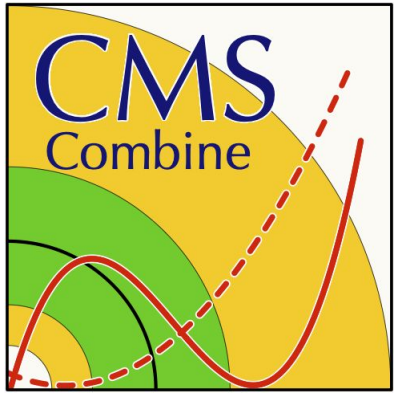



- RooFit workspaces are binary files
  - often only readable by a certain range of ROOT versions (despite valiant efforts to ensure backward-compatibility)
  - hard to understand and edit for humans

*generally not published!*

# CMS Combine

*recently released: Higgs discovery DataCards!*




- CMS recently released their full-suite statistical toolkit COMBINE
- extremely powerful & versatile
- uses DataCard text files as input files
  - can only be read by COMBINE
- uses RooFit as a backend
  - output can be turned into a RooWorkspace

```
imax    1 number of bins
jmax    4 number of processes minus 1
kmax    * number of nuisance parameters
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
bin          signal_region
observation  10.0
--------------------------------------------------------------------------------
bin                          signal_region   signal_region   signal_region   signal_region   signal_region
process                      ttbar           diboson         Ztautau         jetFakes        bbHtautau
process                      1               2               3               4               0
rate                         4.43803         3.18309         3.7804          1.63396         0.711064
--------------------------------------------------------------------------------
CMS_eff_b           lnN      1.02            1.02            1.02            -               1.02
CMS_eff_t           lnN      1.12            1.12            1.12            -               1.12
CMS_eff_t_highpt    lnN      1.1             1.1             1.1             -               1.1
acceptance_Ztautau  lnN      -               -               1.08            -               -
acceptance_bbH      lnN      -               -               -               -               1.05
acceptance_ttbar    lnN      1.005           -               -               -               -
norm_jetFakes       lnN      -               -               -               1.2             -
xsec_diboson        lnN      -               1.05            -               -               -
```

CMS-CAT-23-001

CERN-EP-2024-078
2024/04/11

The CMS statistical analysis and combination tool: COMBINE

The CMS Collaboration*



*... but not everyone uses COMBINE (or RooFit)*

# Diversity of approaches

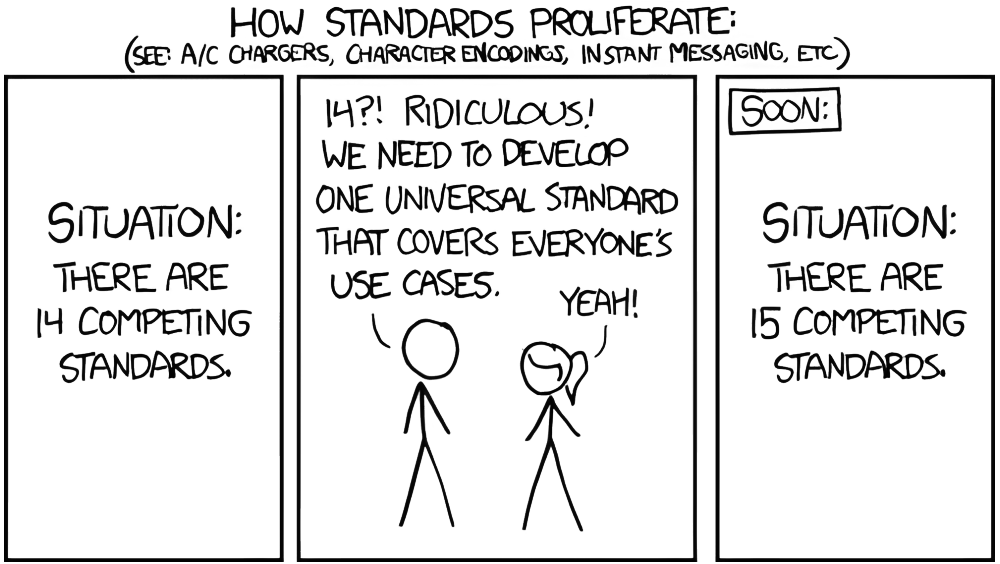
*"Let the thousand flowers bloom"*

- Variety of different approaches & frameworks
- Benefit: Can pick & choose software by feature set
- Downside: Lack of Interoperability



*Can we have a common, standardized format for statistical models?*

# Yet another Standard ?




xkcd.com/927

# What we want

- A descriptive, domain-specific language for serializing models, and it should be
  - feature-complete with respect to ROOT (as the de-facto standard tool in HEP-EX)
  - human-readable (at least in principle)
  - machine-readable for all reasonably common approaches & frameworks
  - well documented to the point where someone implementing a new tool can reproduce the functionality with just the documentation
- Ideally, it should be suitable to
  - exchange models between analysts inside collaborations & outside
  - capable of being published on HEPdata or similar places
  - usable and modifiable by outsiders with little to no domain knowledge
  - useful for outreach and teaching activities
  - carry the legacy of the LHC experiments & their publications

# Where to learn from

- pyhf
  - descriptive
  - human-readable & editable
  - excellent documentation, mostly mathematical
- ROOT
  - RooWorkspace de-facto standard & rich in features

**Honorable mention**

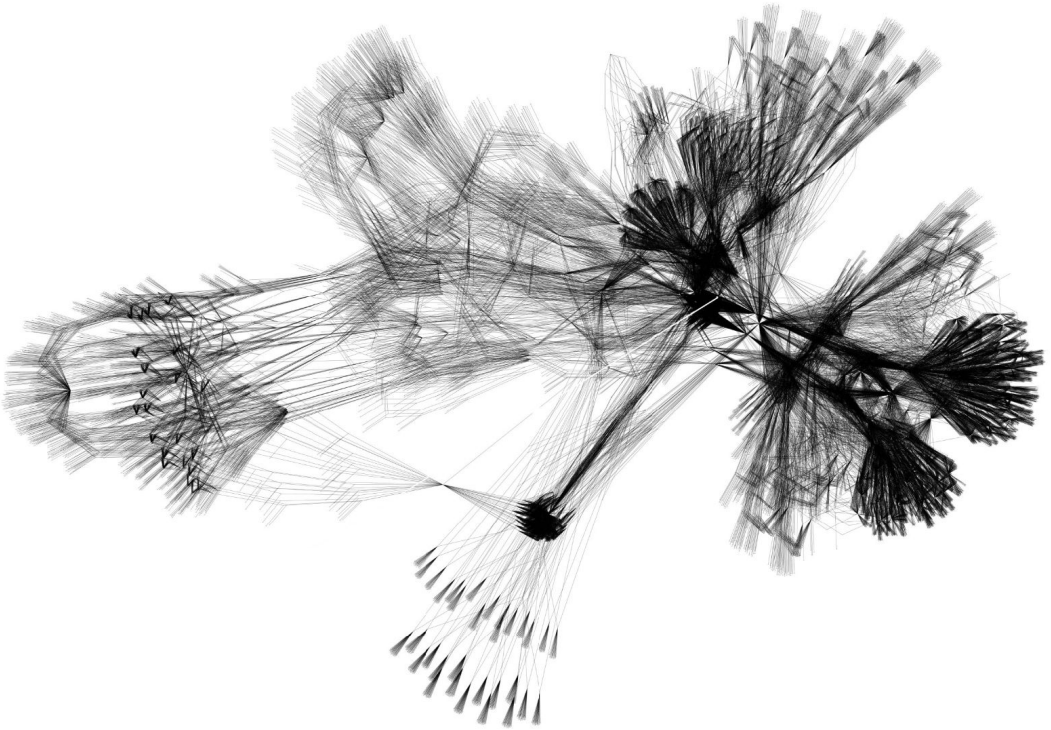
COMBINE datacards are bidirectionally compatible with pyhf JSON!

*If we can have the features of a RooWorkspace convertible to and from ROOT in a format that resembles pyhf JSON with similar quality documentation, we're basically there!*

# ROOT Workspaces

- statistical models are computational graphs
- directed graphs: no cyclic dependencies
- leaves are variables
- intermediate nodes are operations
  - +,-,*,/,...
- higher-level nodes allow to simply the graph
  - exp, Gauss, Poisson, …

computational graph of a Higgs combination workspace

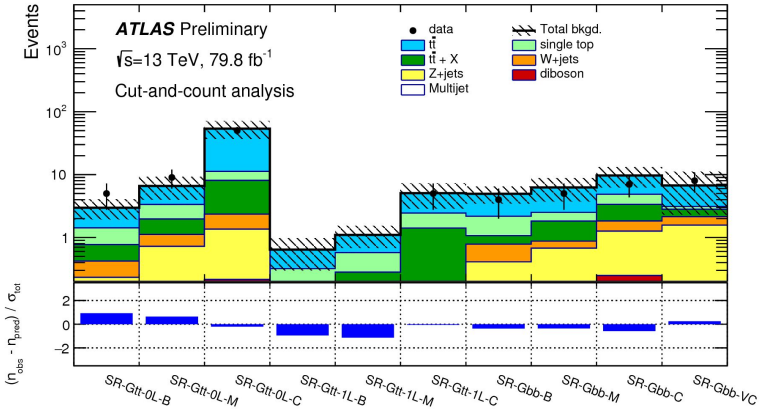




*this graphic was stolen from Wouter Verkerke*

# pyhf JSON

- Model: stacks of histograms with shape & norm modifiers
- Very concise & intuitive
- covery wide variety of use cases



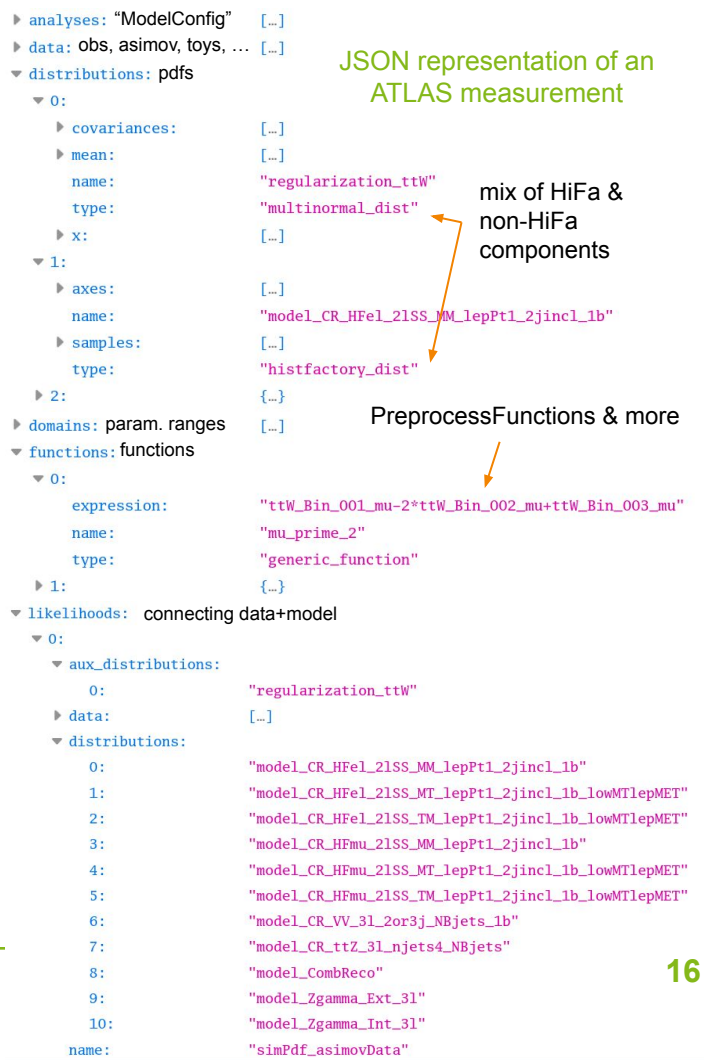

```
{
  "channels": [
    {
      "name": "singlechannel",
      "samples": [
        {
          "name": "signal",
          "data": [ 5.0, 10.0 ],
          "modifiers": [
            { "name": "mu", "type": "normfactor", "data": null }
          ]
        },
        {
          "name": "background",
          "data": [ 50.0, 60.0 ],
          "modifiers": [
            { "name": "uncorr_bkguncrt",
              "type": "shapesys", "data": [ 5.0, 12.0 ] }
          ]
        }
      ]
    }
  ]
}
```



*graphics and code taken from pyhf tutorial*

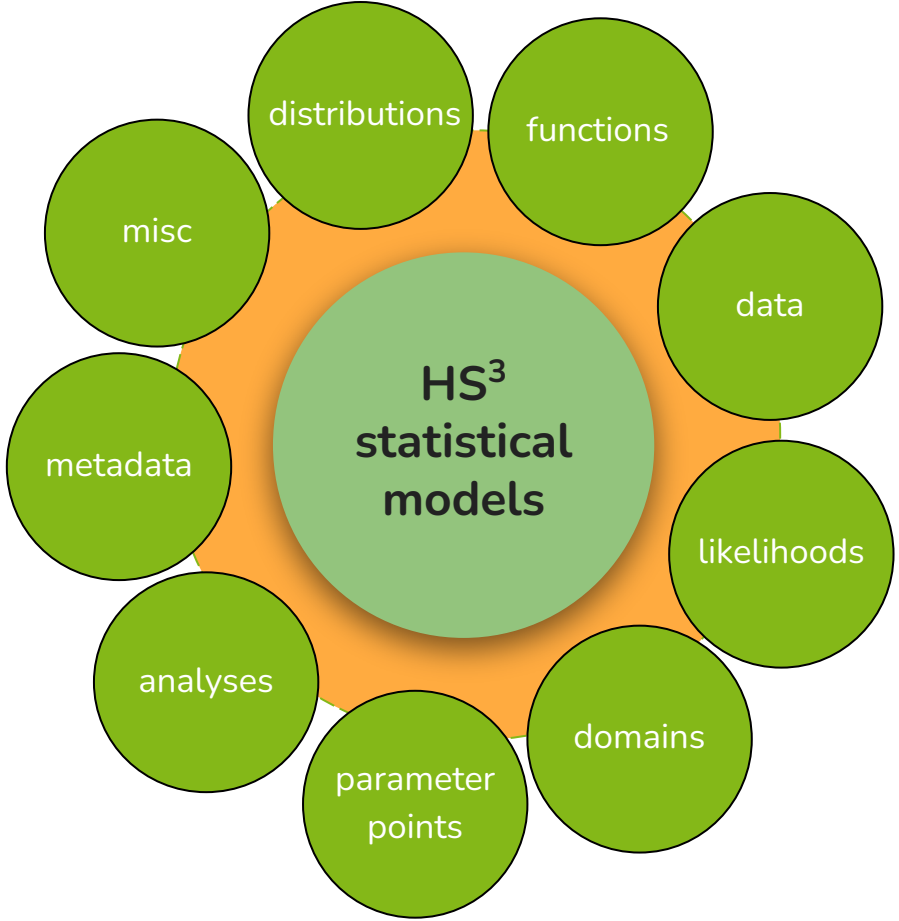
# ROOT Workspaces ⇔ JSON Files

- can easily render dependency graph as text file
  - use JSON, since format is well-received so far
- graphs can be very large
  - what can we do to keep them simple?
  - … other than build simpler models
- common patterns can be abstracted as high-level nodes
  - e.g. HistFactory bin prediction
- work together with ROOT developers to make sure JSON representation is **fully isomorphic** to ROOT Workspace



# Top-level components

- **distributions, functions**: mathematical model
- **data**: observations and toy data
- **likelihoods**: defining loss, constraints
- **parameter points, domains**: info on parameters
- **analyses**: string everything together, identify POIs for interpretations
- **metadata**: version numbers and required packages
- **misc**: custom information dump, should not affect the math






*this graphic was stolen from Robin Pelkner*

# Example: Distributions

```
distributions:
  0:                {…}
  1:                {…}
  2:
    mean:           "nominalLumi"
    name:           "lumiConstraint"
    sigma:          0.1
    type:           "gaussian_dist"
    x:              "Lumi"
  3:                {…}
```

actual usage in JSON

### 2.1.2 Gaussian/Normal distribution

The PDF of a Gaussian/Normal distribution is defined as

$$\text{GaussianPdf}(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{(x-\mu)^2}{\sigma^2}\right).$$

| | |
|---|---|
| `type` | `gaussian_dist` *or* `normal_dist` |
| `x` | value or name of the variable $x$ |
| `mean` | value or name of the parameter used as mean value $\mu$ |
| `sigma` | value or name of the parameter encoding the root-mean-square width $\sigma$. |

definition in the standard

# More examples

**Example: Likelihoods**

```
"likelihoods":[
   {
      "name":"likelihood1",
      "distributions":[
         "dist1",
         "dist2",
         "single_dimensional_dist_1",
         ...
      ],
      "data":[
         "data1",
         "data2",
         0,
         ...
      ]
   },
   ...
]
```

**Example: Analyses**

```
   "analyses": [
       {
       "name" : "analysis1",
       "likelihood" : "likelihood1",
       "aux_likelihood_terms" : ["distribtion_1", "distribution_2", ...]
       "parameters_of_interest" : ["param1"],
       "parameter_domain" : "domain1" ,
       "parameter_init" : "starting_values",
       "prior" : "prior_dist"
       },
       ...
   ]
```

*This is reflective of the current version of HS³ (0.2), next release will follow soon*

# This is not a pipe-dream!

- Need to make sure the standard works for everyone
  - get on board major tool developers
  - exercise roundtrips with complicated models
  - obtain & incorporate suggestions
- So far, reasonably successful
  - fully working implementation in ROOT
    - also available in `combine`!
  - BAT.jl and pyhf implementation are WIP
  - zfit has vowed to implement

HS$^3$

Overview of supported types and components

Carsten Burgard, Tomas Dado, Jonas Eschle, Matthew Feickert, Cornelius Grunwald, Alexander Held, Jerry Ling, Robin Pelkner, Jonas Rembser, Oliver Schulz

HS$^3$ v0.2.9
February 26, 2025

## About this document

To the extent possible under law, the authors have waived all copyright and related or neighboring rights to this document. For details, please refer to the *Creative Commons* CC0 license [1]. This work is published from: CERN, Geneva.

## 1 Introduction

*This section is non-normative.*

It is widely agreed upon that the publication of likelihood models from high energy physics experiments is imperative for the preservation and public access to these results. Detailed arguments have been made elsewhere already [2]. This document sets out to provide a standardized format to publish and archive statistical models of any size and across a wide range of mathematical formalisms in a way that is readable by both humans and machines.

take a look at the overleaf or github to view the current draft!

# A glimpse into the future

- **Successfully** applied for funding for a DFG “Sachbeihilfe”: 1 PhD student for 3 years
  - take the perspective of an “outside researcher” (theorist)
  - validate applicability of HS$^3$ public likelihoods for EFT combinations
  - full treatment of sys. uncertainties en par with collaboration-internal combinations
  - publish an EFT interpretation of some ATLAS+CMS+LHCb public likelihoods
  - publish the entire toolchain used alongside documentation
- *Currently applying* for an ErUM data research data management project
  - collaborative project with astrophysics, laser physics, particle physics, hadron physics, neutrino physics, nuclear physics
  - plan to promote self-governance of HS³ through formalization of review procedure
- Python implementation being actively developed by core members of the pyhf team
  - hope to have a working prototype by the summer
- Julia implementation migrated to JuliaHEP for easier community involvement
- Several ATLAS analyses published HS$^3$ likelihoods on HEPdata (list)

DFG Deutsche Forschungsgemeinschaft

Bundesministerium für Bildung und Forschung

# Conclusions

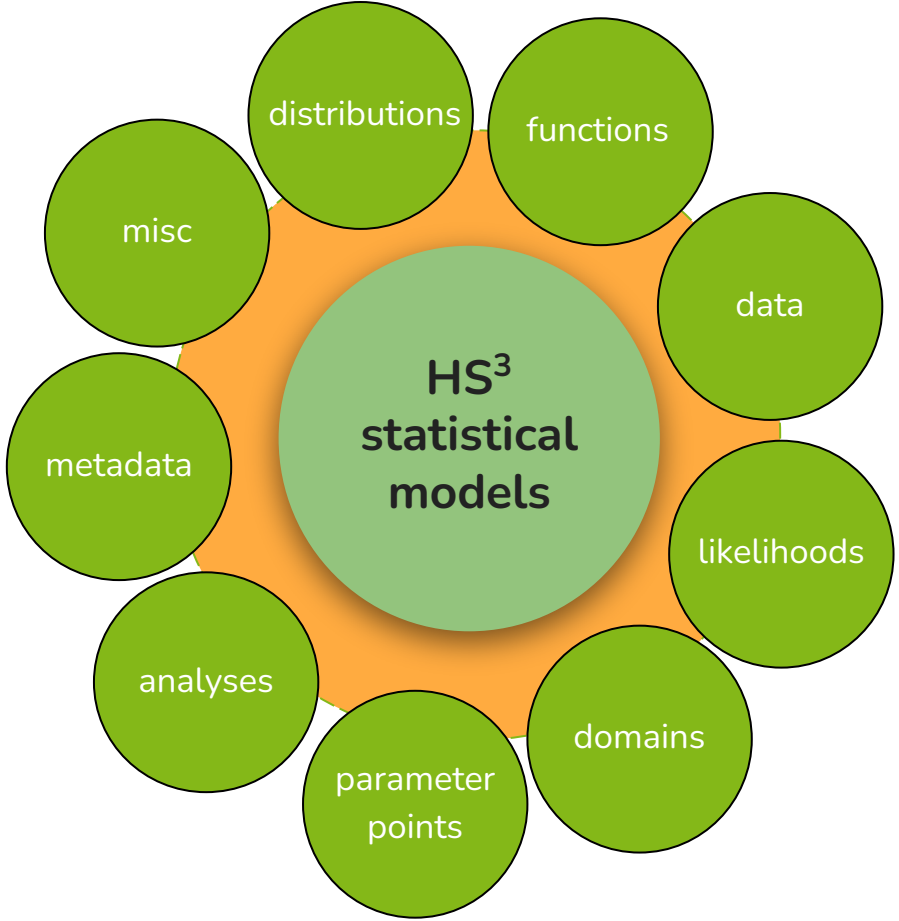
- Publishing full likelihoods of experimental measurements has been an agreed-upon goal of the High Energy Physics community for decades
  - actual implementation has been extremely slow
  - First success stories exist (pyhf, COMBINE), but no common interoperable format
- $HS^3$ is a new and developing standard to serialize statistical models
  - isomorphic to RooWorkspace, fully implemented in ROOT
  - ROOT 6.34/04 implements $HS^3$ v 0.2.9 (released Feb 10, 2025)
  - $HS^3$ v 0.3 is WIP, please discuss on the github page!
- First measurements published in $HS^3$ format
  - non-ROOT implementations are under way, hoping for community involvement
  - community adoption in full swing, reaching out to other branches of physics

**$HS^3$ is an attempt at developing a format that works for everyone.**
**Make your voice heard!**

# Backup

# Top-level components

- **distributions, functions**: mathematical model
- **data**: observations and toy data
- **likelihoods**: defining loss, constraints
- **parameter points, domains**: info on parameters
- **analyses**: string everything together, identify POIs for interpretations
- **metadata**: version numbers and required packages
- **misc**: custom information dump



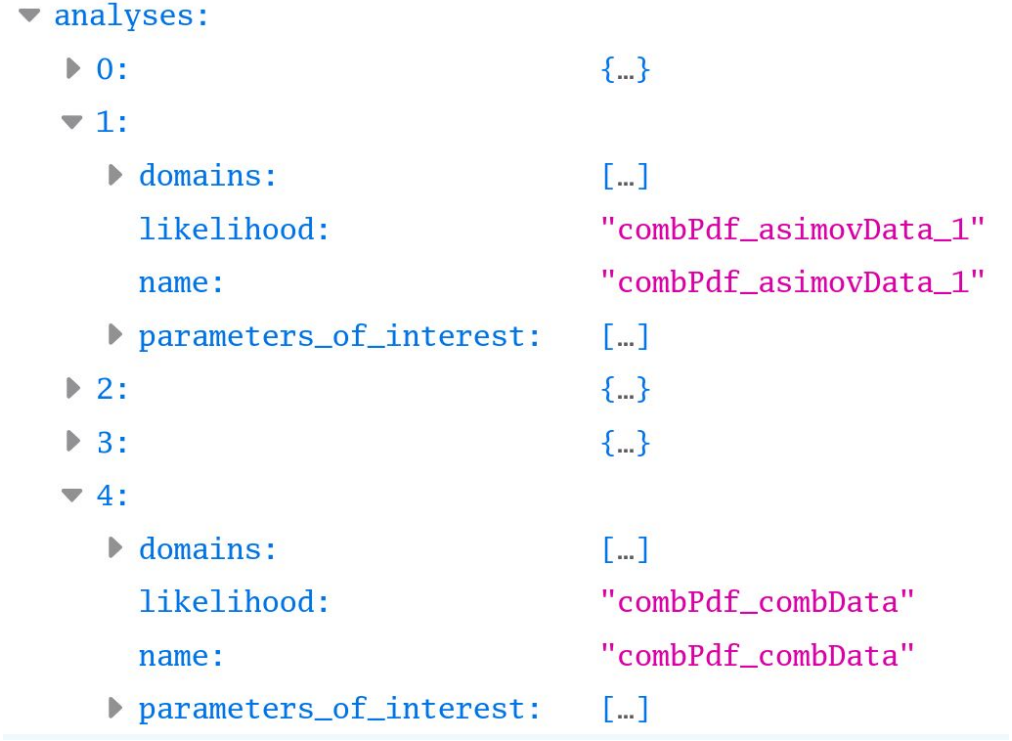



*this graphic was stolen from Robin Pelkner*

# analyses

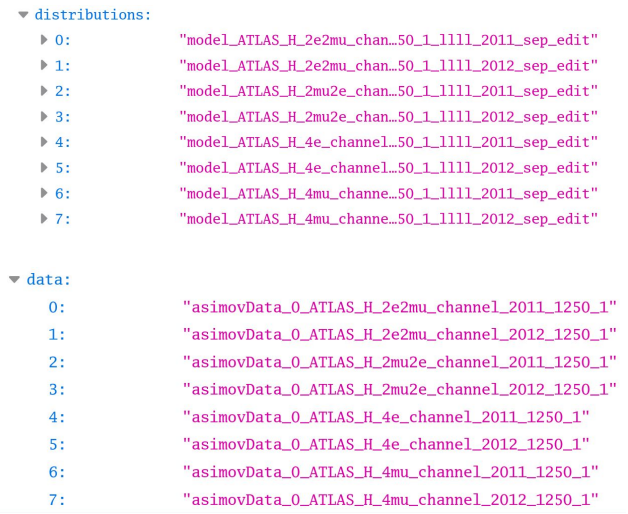
- if you know RooFit:
  - analysis = ModelConfig
- contains inference suggestions
  - list of POIs
  - parameter domains (ranges)
  - point to likelihood
- optional element

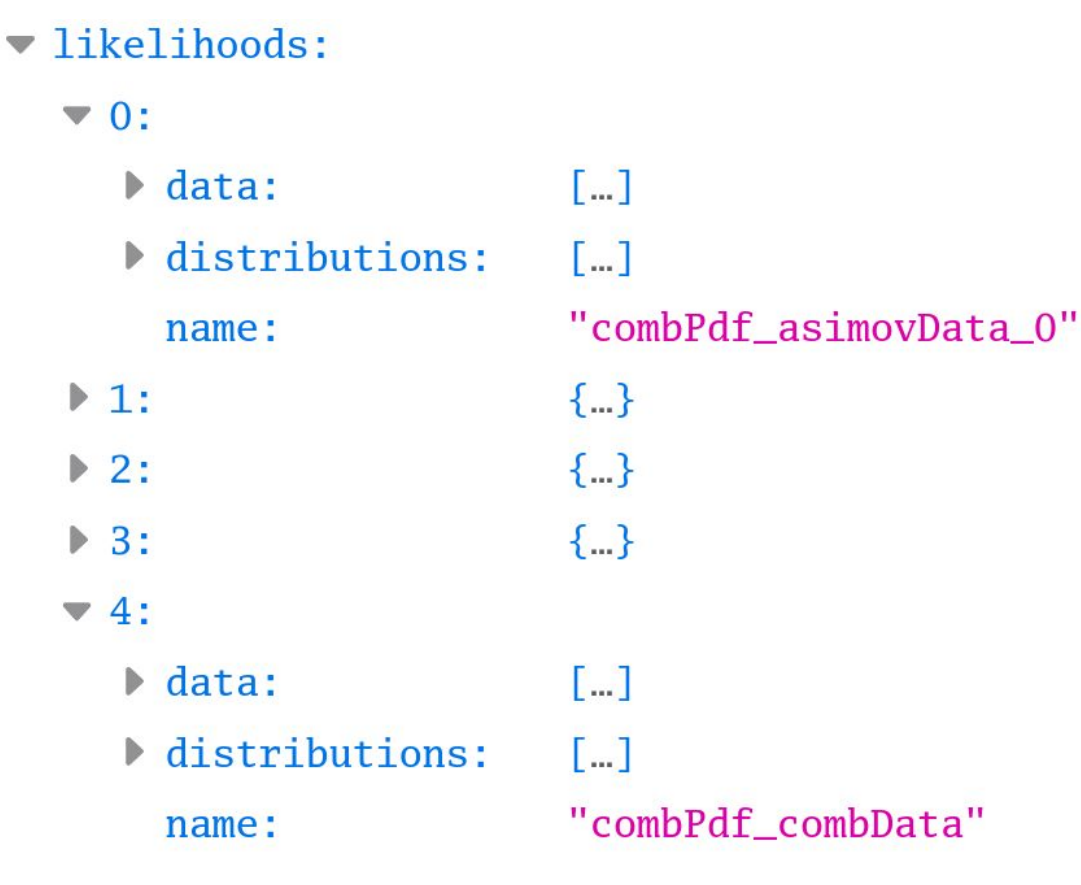
*you can always leave it to the user to come up with descriptions of what they want to do*



# likelihoods

- list of likelihoods in the model
- each one identifies
  - list of distributions (named)
  - list of datasets (named)



# data

- one entry per dataset
  - one per “channel” or “region”
- binned and unbinned supported
  - regular and irregular binnings
  - only rectangular for now
- use `axes` to identify dimensions
  - arbitrary dimensionality
  - name-matching to observables in pdfs

# distributions

- one entry per pdf
- “type” encodes the type of Pdf
- definition + explanation found in the standard document (evolving)

```
▼ 40:
    alpha:      "tailAlpha_PttEtaConvVBFCat2_gg_2011"
    m:          "atlas_invMass_PttEtaConvVBFCat2_gg_2011"
    m0:         "mHiggs_PttEtaConvVBFCat2_gg_2011"
    n:          10
    name:       "peakPdf_PttEtaConvVBFCat2_gg_2011"
    sigma:      "atlas_expected_mRes_PttEtaConvVBFCat2_gg_2011"
    type:       "crystalball_dist"
```

*Please take a look at the standard and find whether any of the distributions important to you is not yet included*

*Long term, we strive to include all reasonably frequently used pdf types in the standard!*

# functions

- very similar to distributions
  - not self-normalizing
  - allowed to yield negative numbers
- auxiliary objects
  - not needed for many models
  - main use case: reparametrization
- arithmetic functions to transform variables (product, sum, …)
- special feature: `generic_function`
  - needed to cover RooFormulaVar
  - simple arithmetic transformations
  - use should be avoided if possible

```
▼ 4:
  ▼ factors:
      0:          "mu_XS_ggF_gg_2011"
      1:          "mu_gg_2011"
      2:          "mu_BR_gamgam_gg_2011"
    name:         "mu_ggF_PttEtaConvVBFCat1_gg_2011"
    type:         "product"
```

```
▼ 85:
  ▶ expression:   "-atlas_log_kappa_val_sca…tEtaConvVBFCat2_gg_2011"
  ▶ name:         "atlas_log_kappa_val_scal…11_exponential_inverted"
    type:         "generic_function"
```

# parameter_points

- store parameter configurations
  - needed for some inference techniques
- convenience feature
  - starting points for minimizations
  - best-fit-values for easy reruns
  - true parameter sets for toy data
  - …
- maps to snapshots in RooFit

```
▼ parameter_points:
  ▼ 0:
      name:          "default_values"
    ▼ parameters:
      ▼ 0:
          const:     true
          name:      "ATLAS_CELLOUT_In"
          value:     0
```

# domains

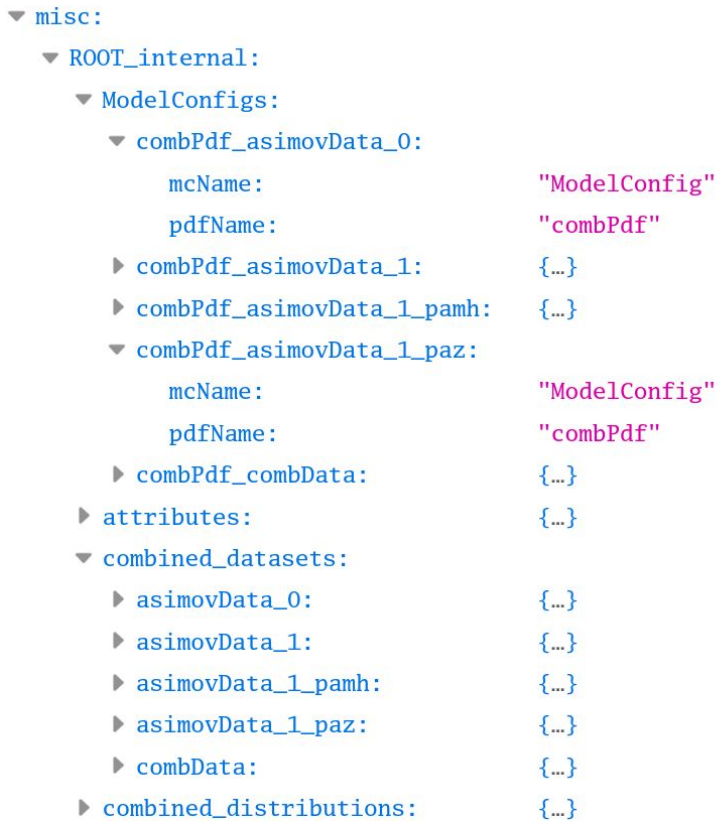
- store parameter ranges
  - needed for some bayesian inference techniques
  - useful for closure with RooFit

```
▼ domains:
  ▼ 0:
    ▶ axes:          […]
      name:          "default_domain"
      type:          "product_domain"
```

# metadata

- track model provenance
  - who created this model?
  - when?
  - how?
  - which packages were used?
- will be extended in the future
  - include HEPdata fields?
  - …?
- heavily formalized
- suggestions welcome

# misc

- additional field to store arbitrary, possibly framework-related information
  - tags for categorization of objects
  - pretty-print titles
  - plotting colors
  - …
- contents are not standardized
  - must not affect the result of inference in any way

*anything that is nice-to-have but not standardizable enough to fit in metadata goes here!*