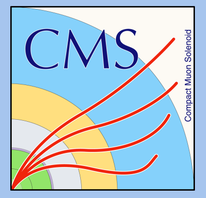


PRINCETON
UNIVERSITY

HEP analyses with gradients

Differentiating a CMS-like analysis with the scikit-HEP ecosystem

Mohamed Aly and Lino Gerlach

09.06.2025 @ MODE 2025

What are HEP measurements

What is HEP work about?

The Standard Model (SM)

Our best description of the universe at the quantum scale is the *Standard Model (SM)*

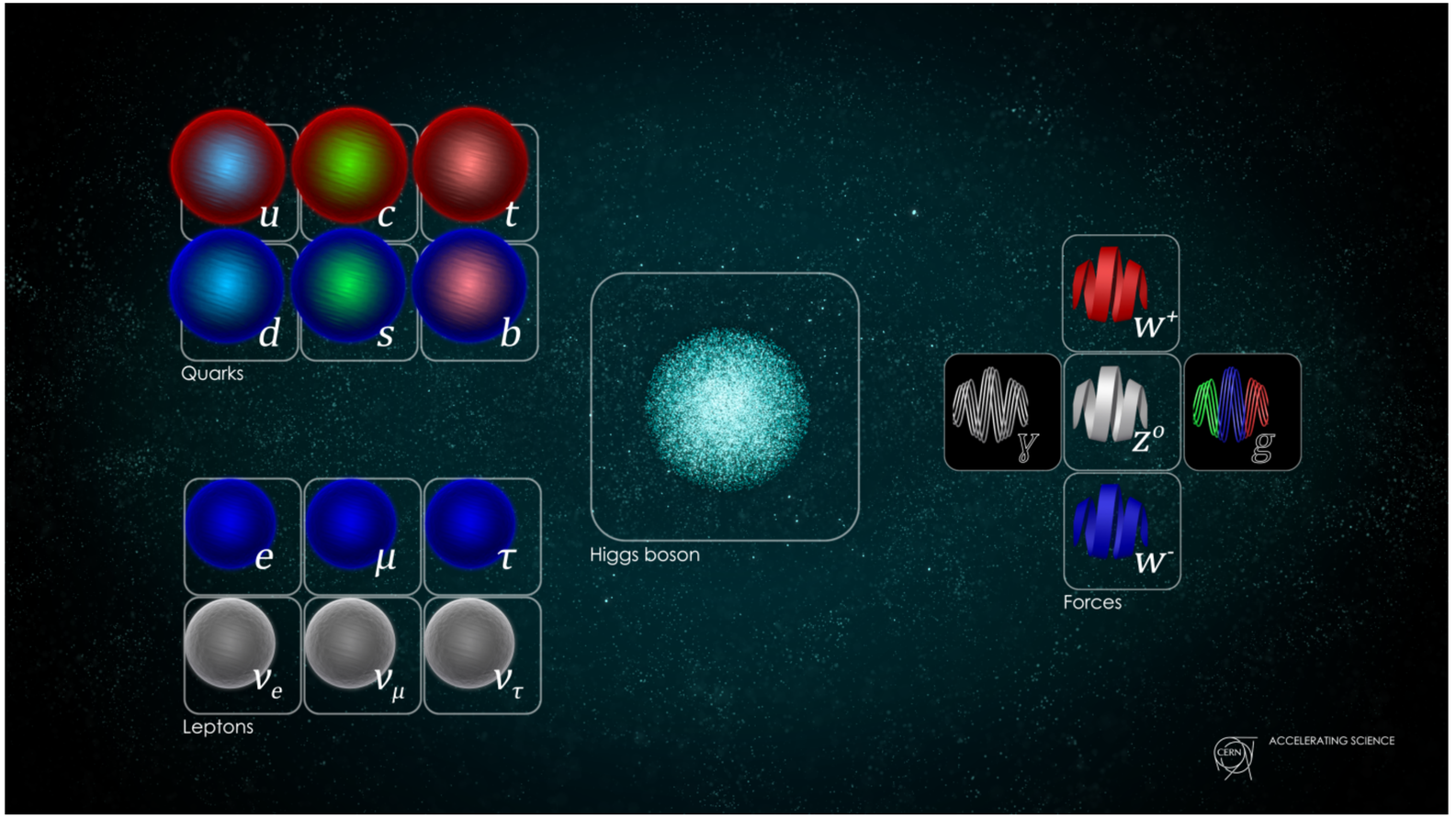


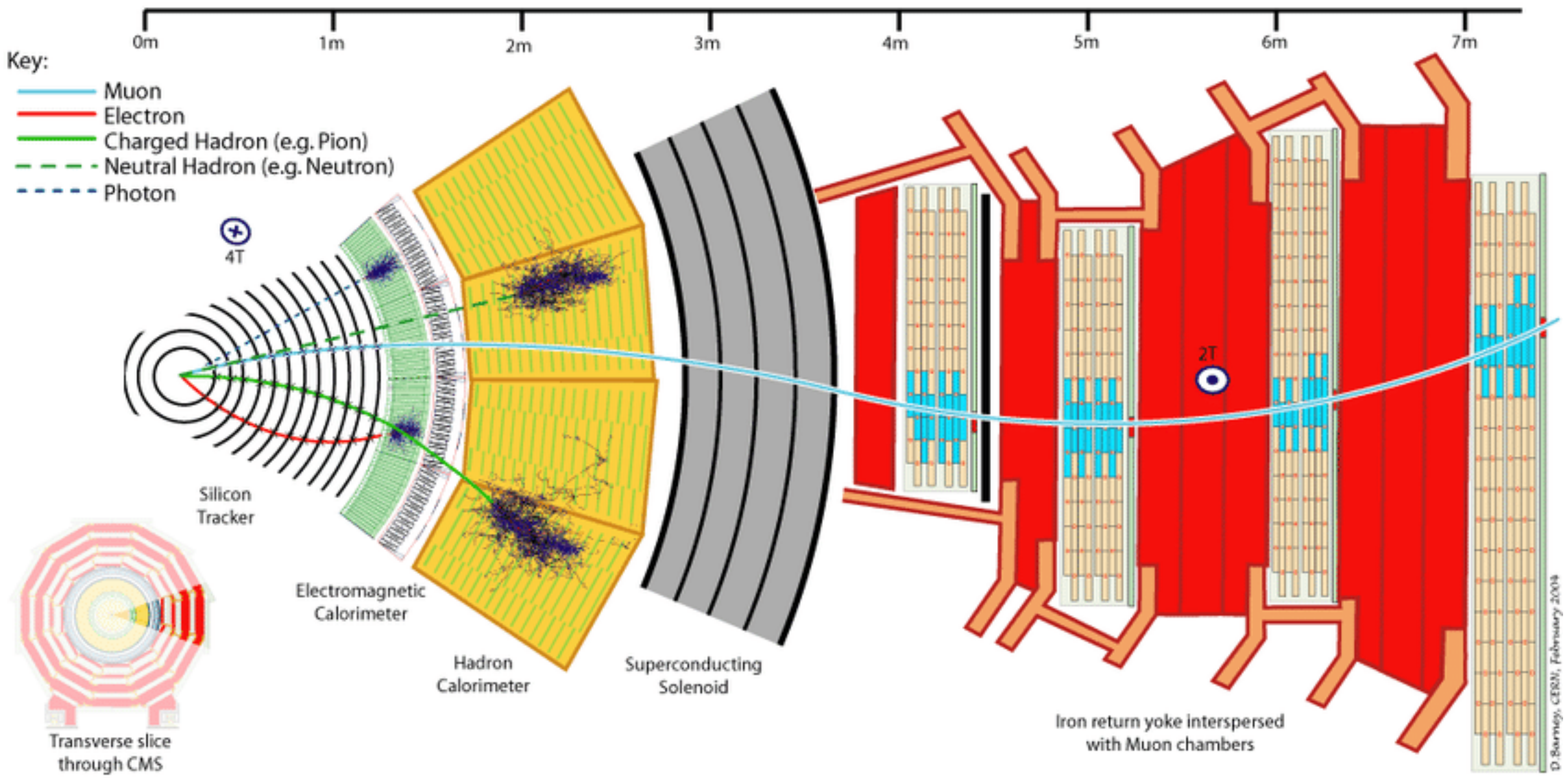
Image from CERN website

What is HEP work about?

Testing the SM (and other theories)

We build big expensive experiments (e.g. CMS) to test theoretical predictions

Schematic from CERN website



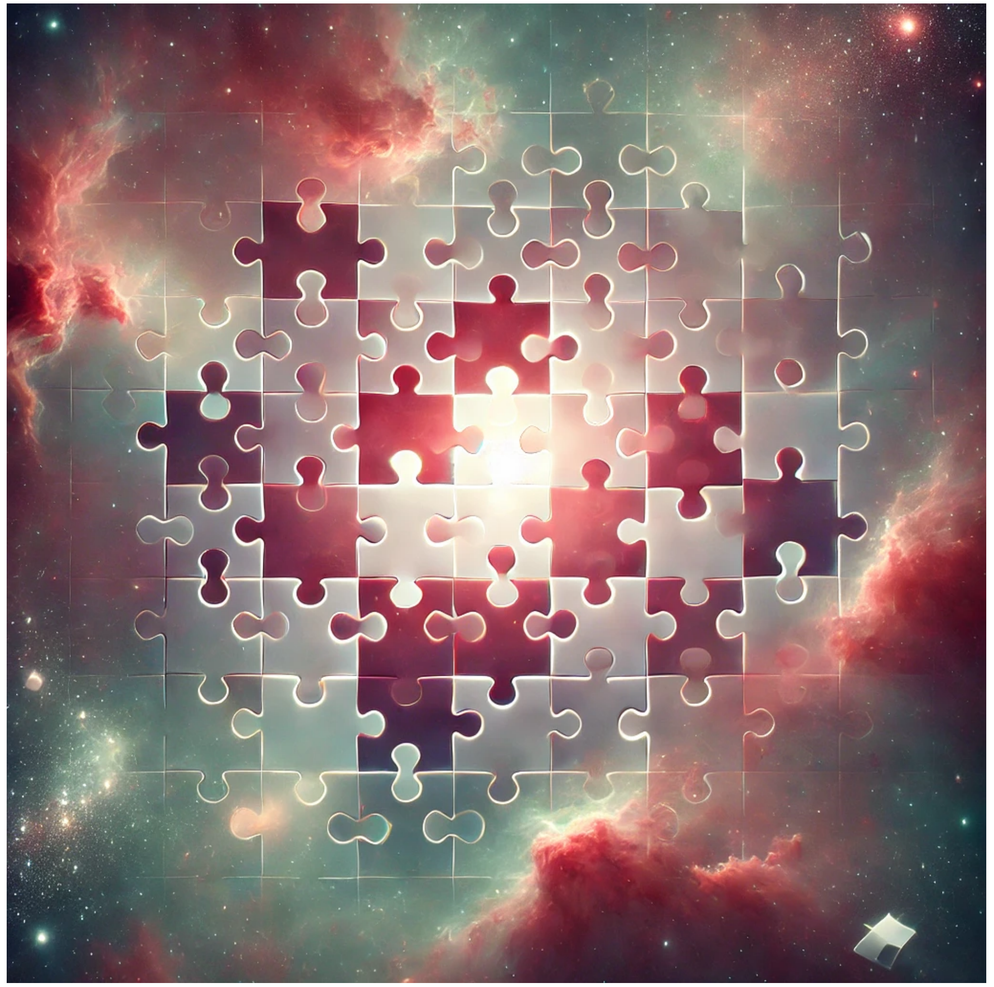
What is HEP work about?

What we know so far

Two main headlines:

1. Almost all SM predictions have been confirmed to high precision

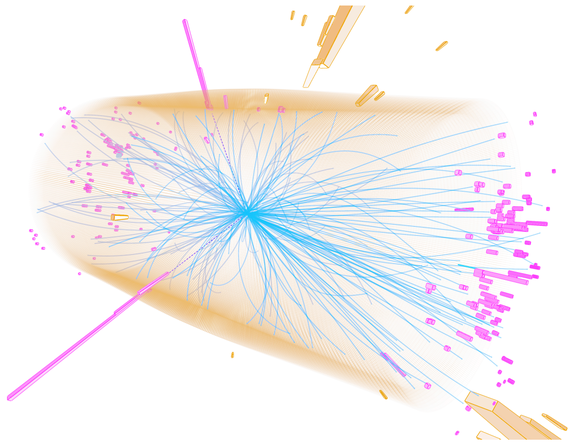
2. The SM is unable to explain some fundamental observations in nature — must look Beyond SM (BSM)



What is a HEP measurement?

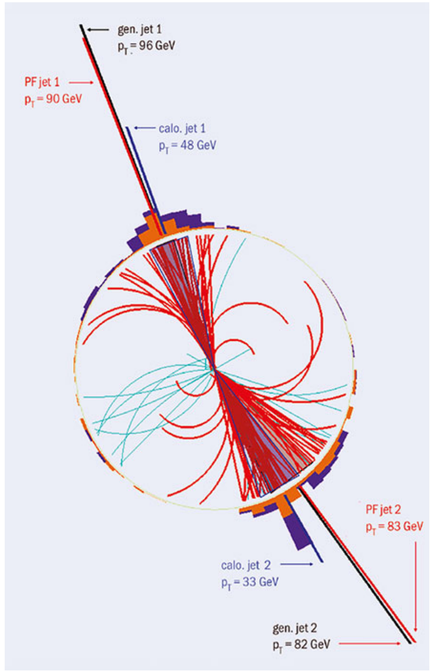
An overview of the pipeline

Raw data

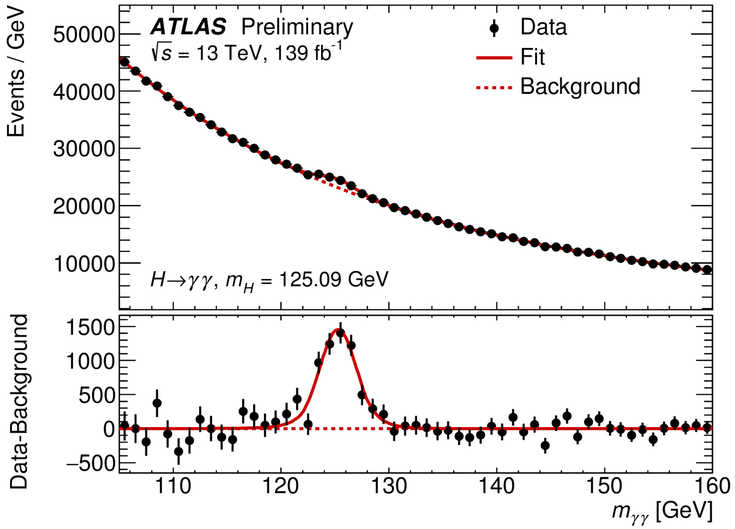


CMS event displays of Higgs to two photon candidate

Object reconstruction



Statistical analysis



What is a HEP measurement?

An overview of the pipeline

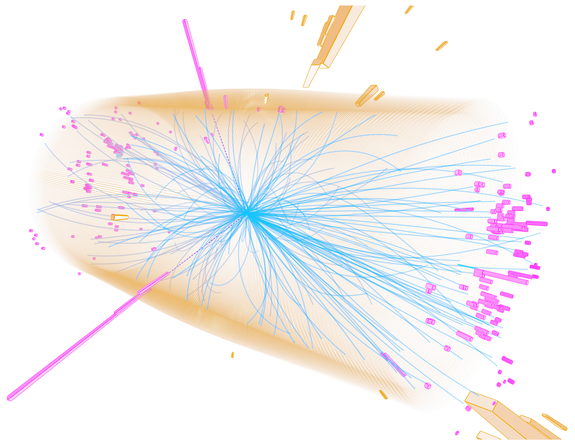
Raw data

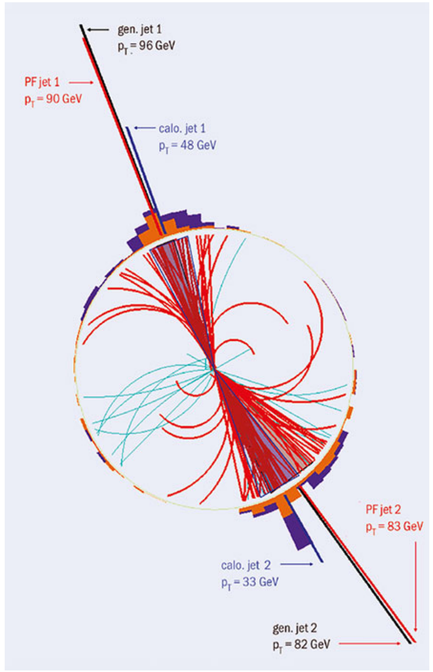
Object reconstruction

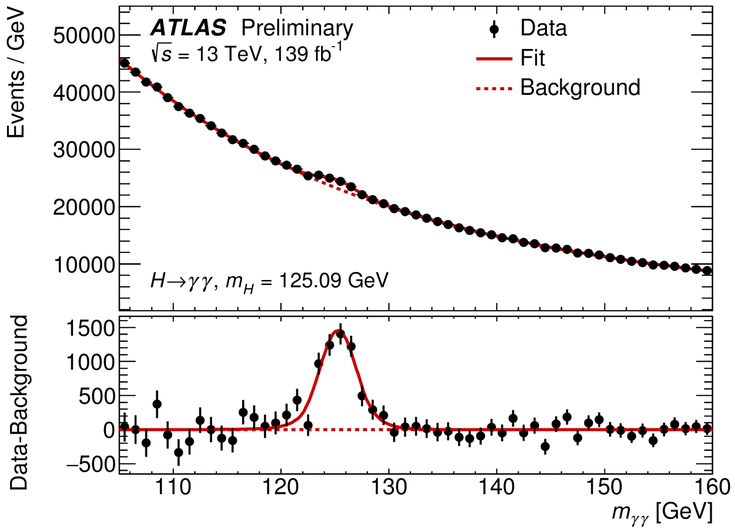
Statistical analysis

Focus of today

CMS event displays of Higgs to two photon candidate



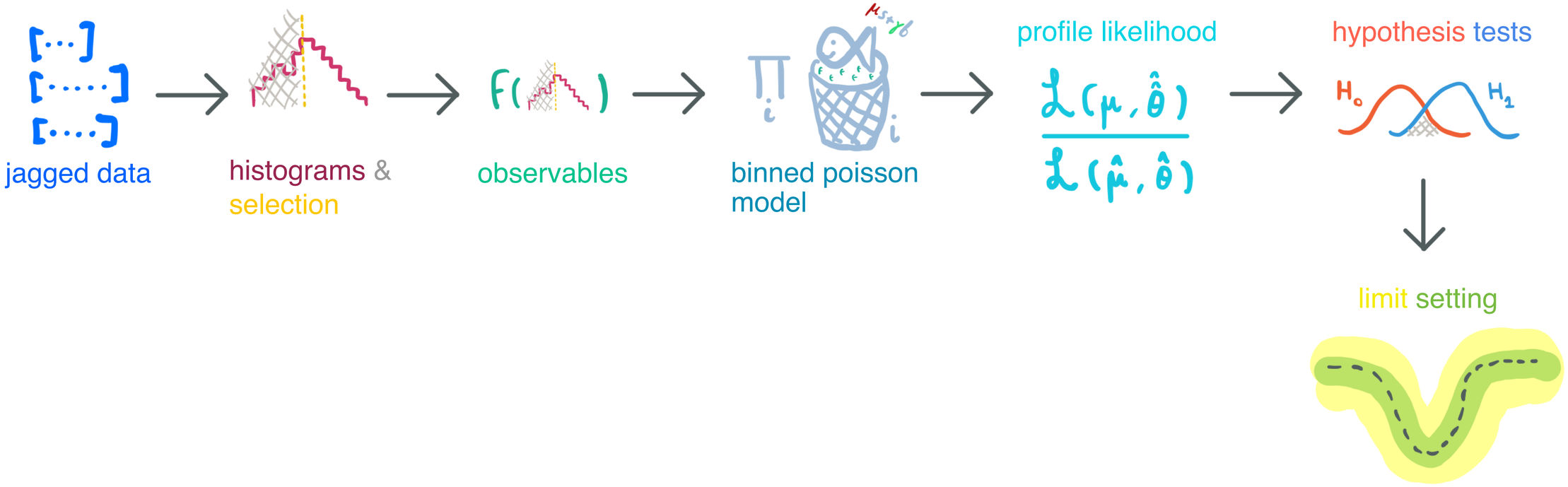




A HEP statistical analysis

From reconstructed particles to a physics objective

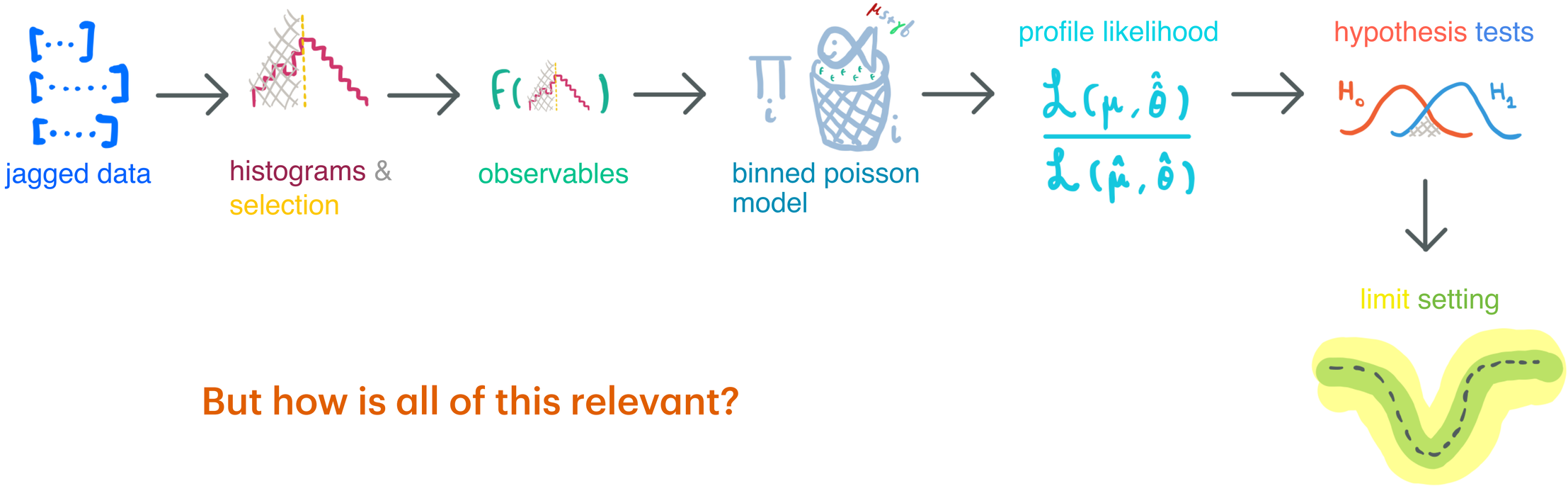
From Nathan Simpson's talk



A HEP statistical analysis

From reconstructed particles to a physics objective

From Nathan Simpson's talk

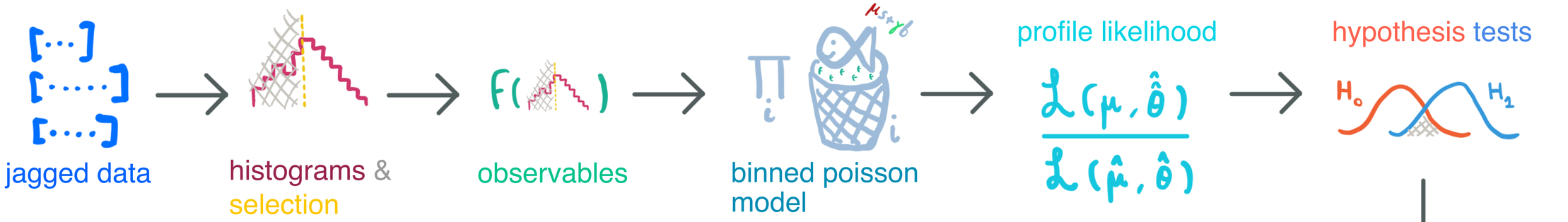


But how is all of this relevant?

A HEP statistical analysis

From reconstructed particles to a physics objective

From Nathan Simpson's talk



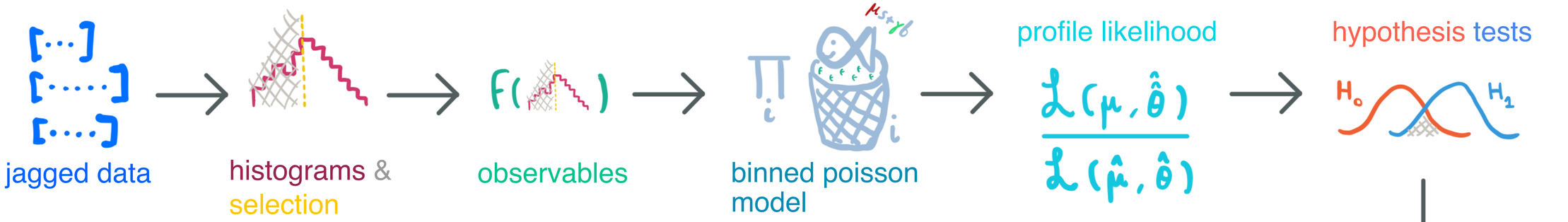
Steps with free parameters

Physics objective

A HEP statistical analysis

From reconstructed particles to a physics objective

From Nathan Simpson's talk



Steps with free parameters

Can we optimise the objective with respect to free parameters? [c.f. NNs]

Physics objective

The need for HEP gradients

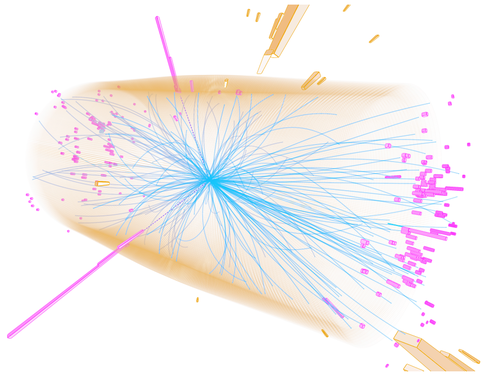
Why trace gradients?

Towards end-to-end learning?

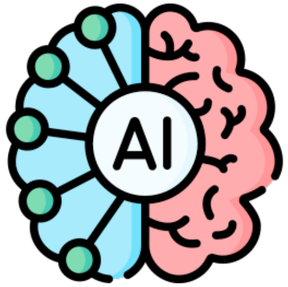
AI/ML entering more and more steps in the HEP pipeline → learnable end-to-end analyses?

CMS event displays of Higgs to two photon candidate

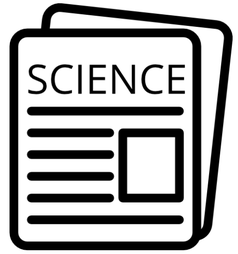
Raw data



Black box



Physics objective



Higgs found!

Do we trust this?

Why trace gradients?

Inserting physics knowledge

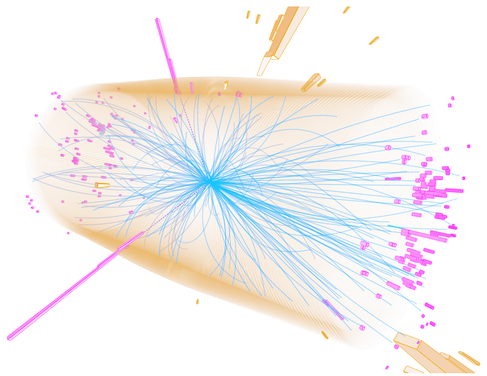
Inspired by Lukas Heinrich's talk

More realistically: a physics-informed hybrid approach → knowledge inserted via data flows between nodes

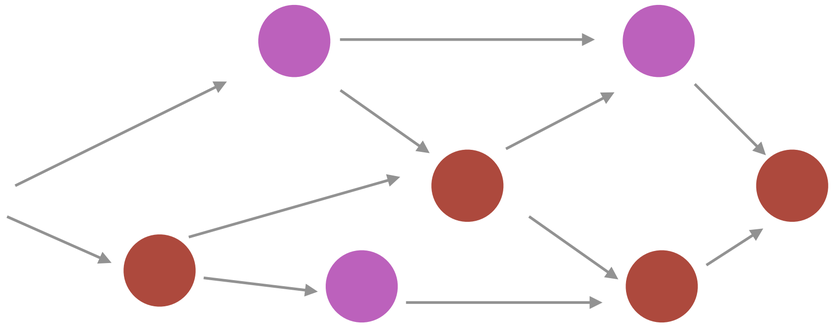
Raw data

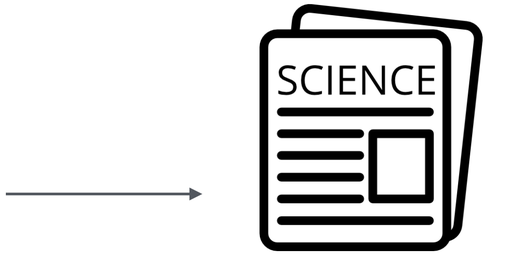
Hybrid pipeline

Physics objective



CMS event displays of Higgs to two photon candidate





Higgs found!

Physics Computations

ML Algorithms

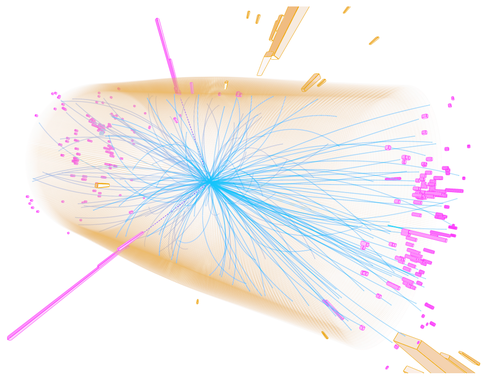
Why trace gradients?

Holistic optimisation of hybrid pipeline

Inspired by Lukas Heinrich's talk

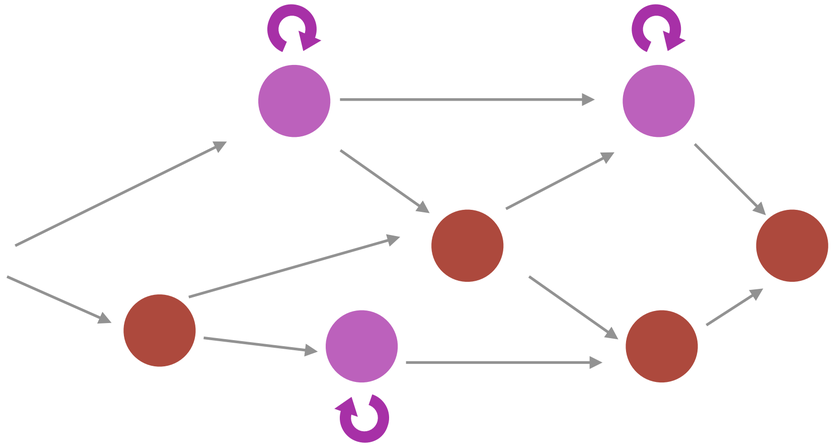
Each ML step is trained independently to optimise an objective

Raw data



CMS event displays of Higgs to two photon candidate

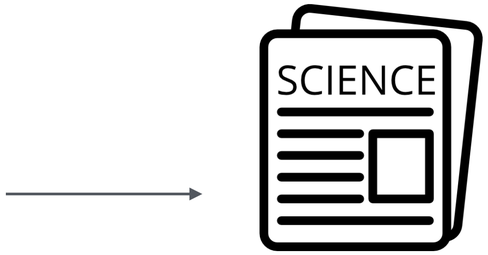
Hybrid pipeline



Physics Computations

ML Algorithms

Physics objective



Higgs found!

Why trace gradients?

Holistic optimisation of hybrid pipeline

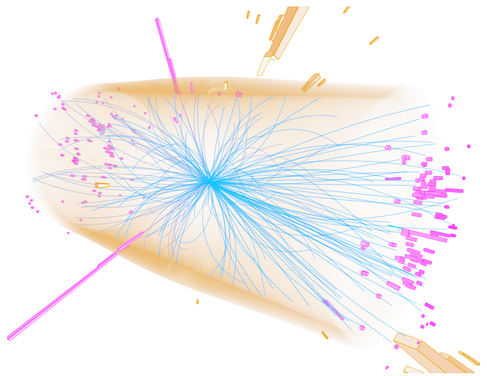
Inspired by Lukas Heinrich's talk

Ideally, we want to optimise everything for our physics objective
— i.e. back propagate through the pipeline

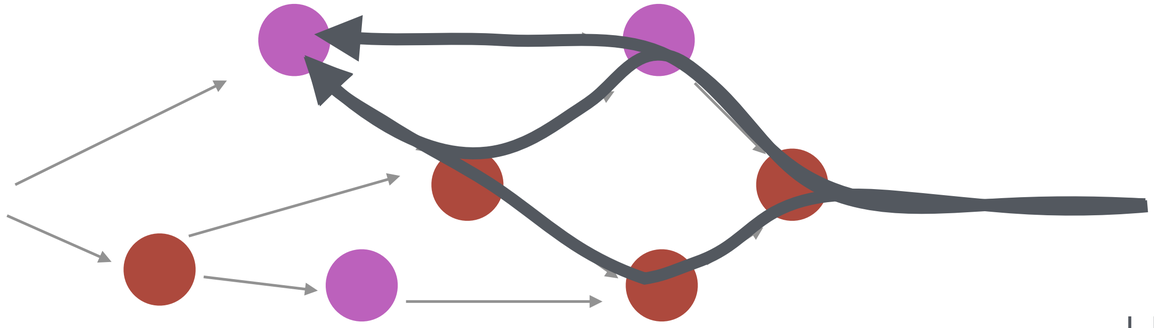
Raw data

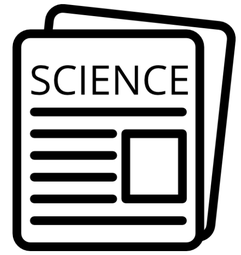
Hybrid pipeline

Physics objective



CMS event displays of Higgs to two photon candidate





Higgs found!

Physics Computations

ML Algorithms

Why trace gradients?

Holistic optimisation of hybrid pipeline

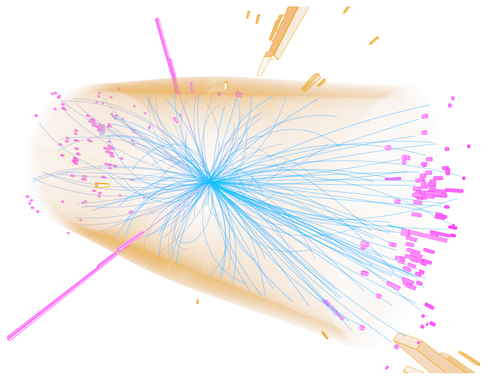
Inspired by Lukas Heinrich's talk

Need to track gradients for holistic optimisations!

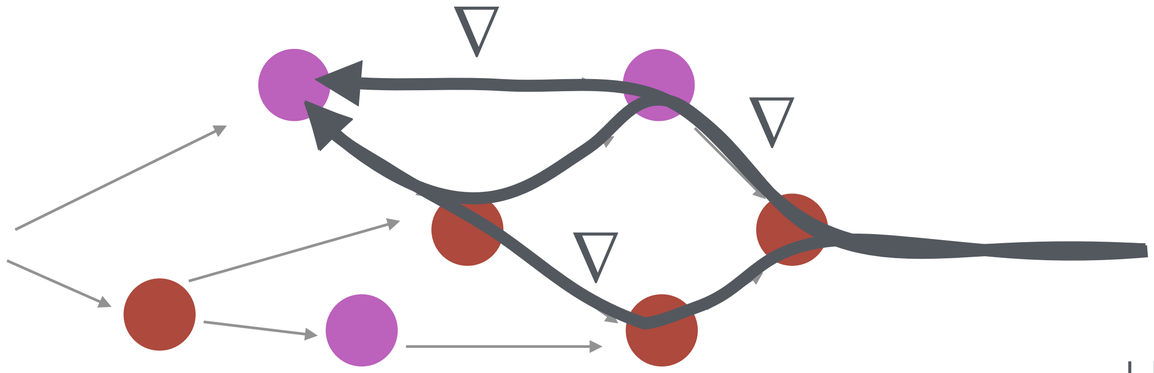
Raw data

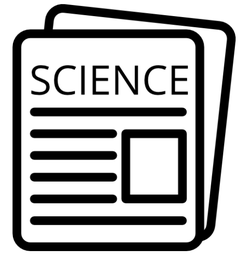
Hybrid pipeline

Physics objective



CMS event displays of Higgs to two photon candidate





Higgs found!

Physics Computations

ML Algorithms

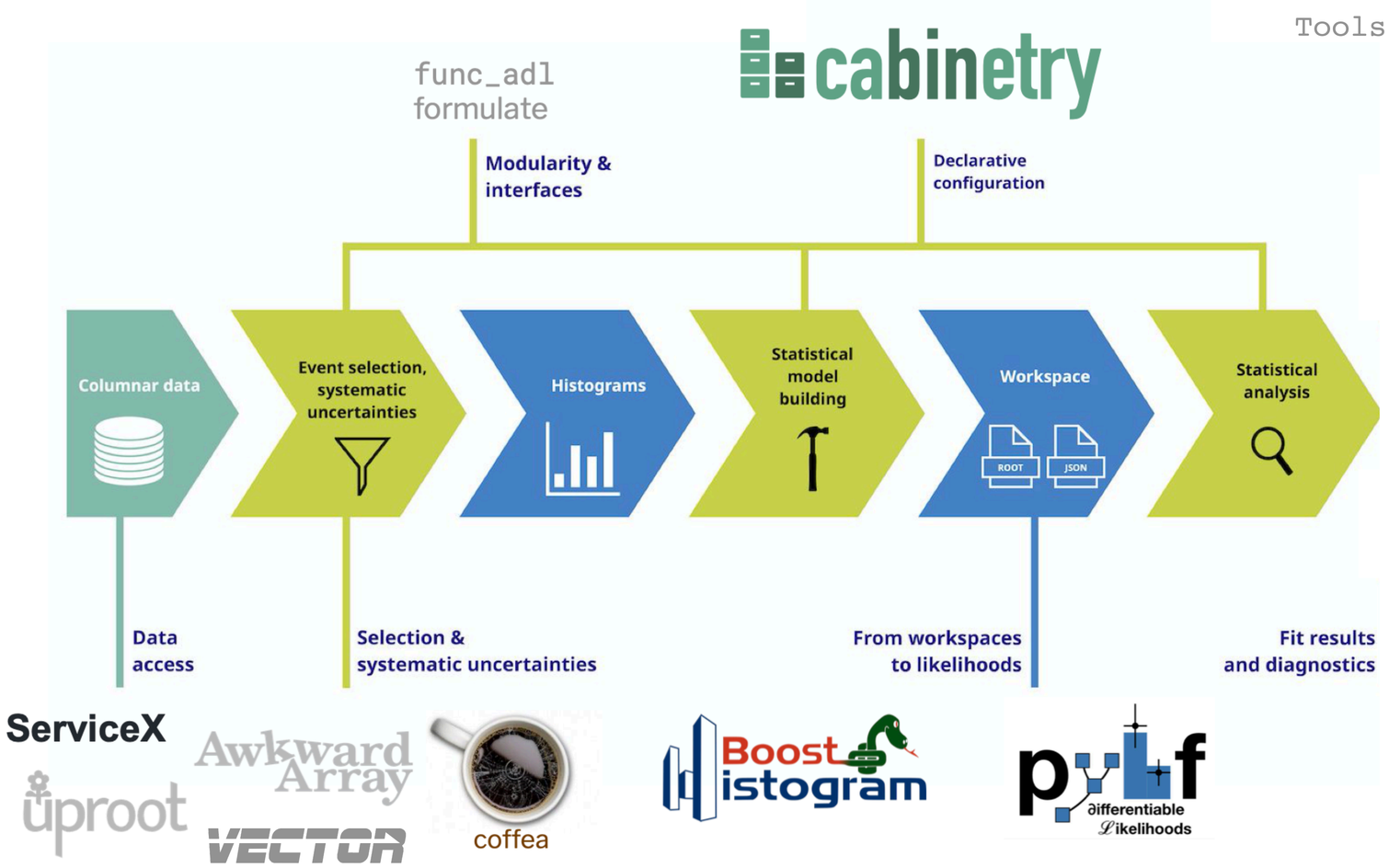
The path to HEP gradients

The scikit-HEP Ecosystem

Open-source pythonic data analysis tools

Modified from IRIS-HEP website

Tools under Scikit-HEP project



The Scikit-HEP Ecosystem

Particularly important

Awkward Array

Ragged arrays with numpy-like idioms

VECTOR

Support for Lorentz-vectors (4-vectors)

pyhf
differentiable
likelihoods

Constructing differentiable likelihoods

The Scikit-HEP Ecosystem

Particularly important

Awkward Array

Ragged arrays with numpy-like idioms

VECTOR

Support for Lorentz-vectors (4-vectors)

pyhf
differentiable
Likelihoods

Constructing differentiable likelihoods

We need to trace gradients through all the tools...

The Scikit-HEP Ecosystem

Particularly important

Awkward Array

Ragged arrays with numpy-like idioms

VECTOR

Support for Lorentz-vectors (4-vectors)

pyhf
differentiable Likelihoods

Constructing differentiable likelihoods

We need to trace gradients through all the tools...

Is this enough?

The Scikit-HEP Ecosystem

Particularly important

Awkward Array

Ragged arrays with numpy-like idioms

VECTOR

Support for Lorentz-vectors (4-vectors)

pyhf
differentiable Likelihoods

Constructing differentiable likelihoods

We need to trace gradients through all the tools...

Is this enough?

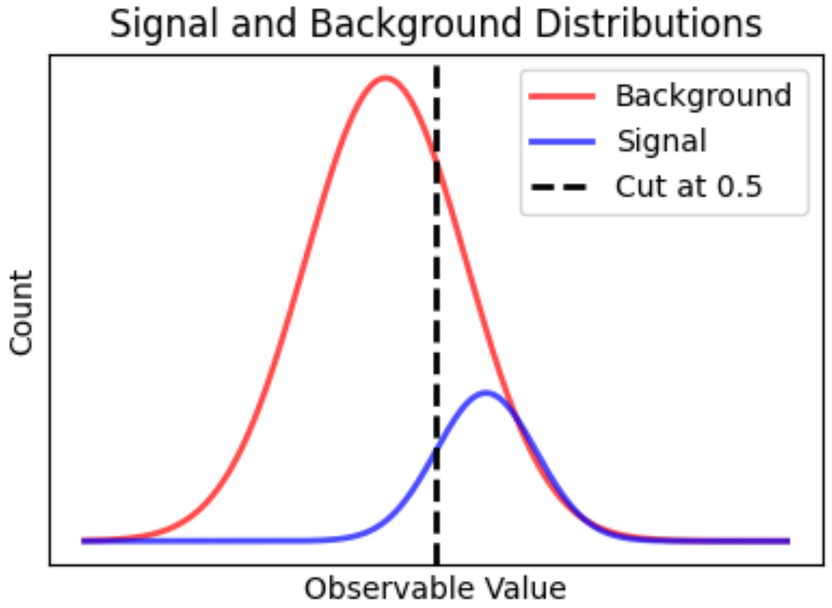
Not quite...

Differentiable Relaxations

Making discrete primitives differentiable

Typical operations applied on HEP data are non-differentiable

e.g. applying a threshold, sorting

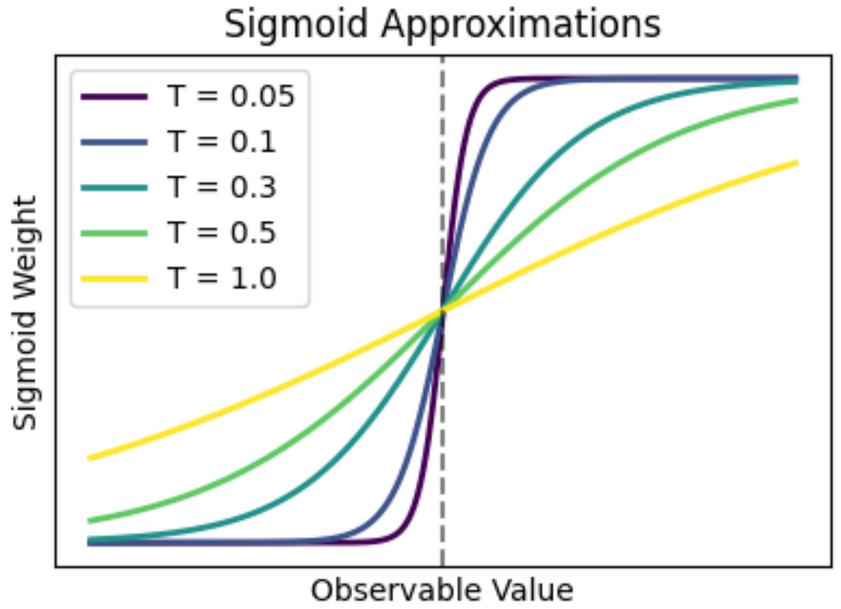


Differentiable Relaxations

Making discrete primitives differentiable

We can approximate (smoothen) the operations with differentiable functions → differentiable relaxations

Soft sorting demo for a mass reconstruction problem

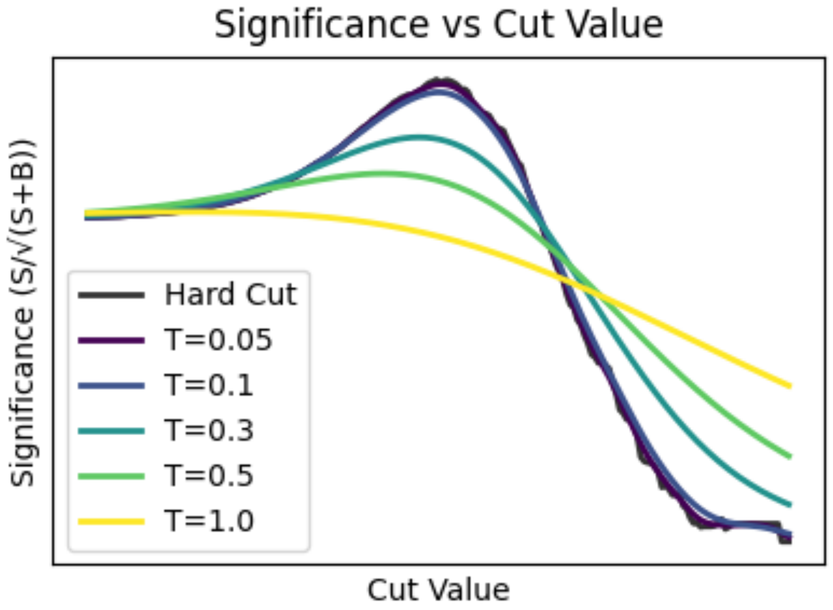


Smoothing controlled by the "Temperature" (T)

Differentiable Relaxations

Making discrete primitives differentiable

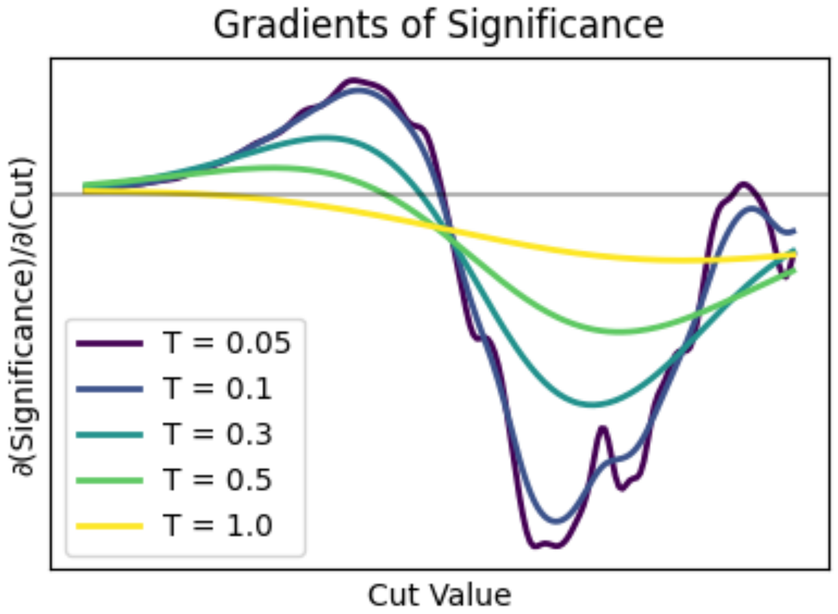
Optimal significance value is less robust against softening than its location (i.e. cut value)



Differentiable Relaxations

Making discrete primitives differentiable

A trade-off: faithfulness to a hard threshold and stability of in gradient evaluation

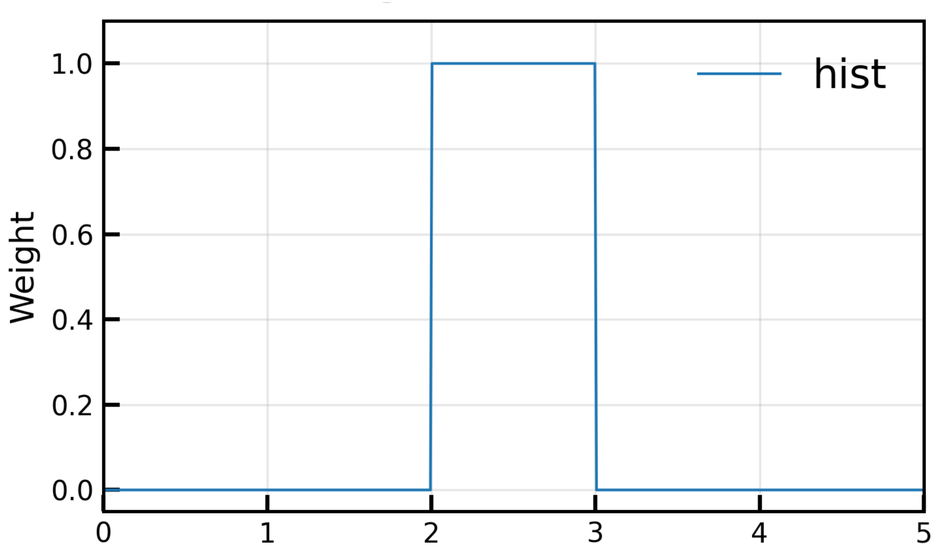


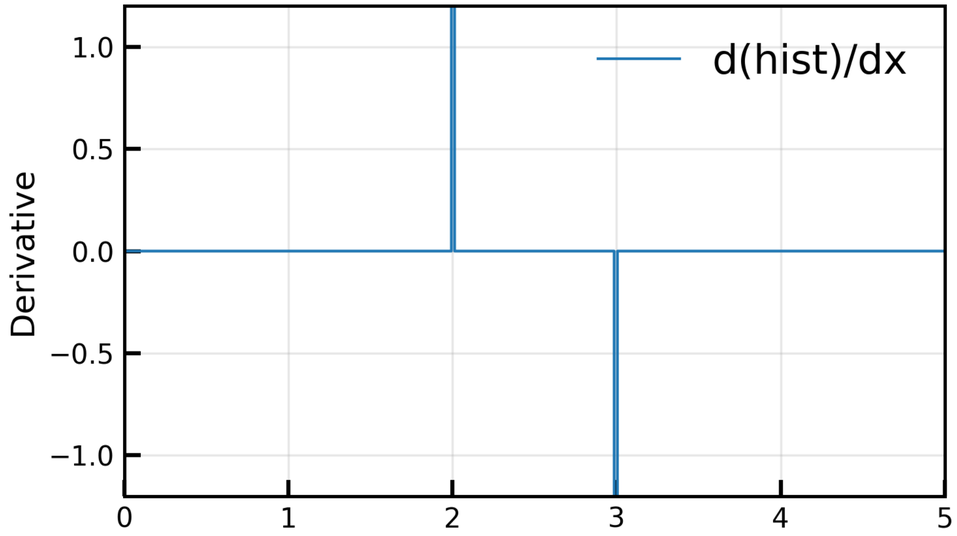
Differentiable Relaxations

Making histograms differentiable

Histograms are central (non-differentiable) data structures in HEP analyses

Vary data point: land in same bin, 0 gradient, land in another bin, infinite gradient



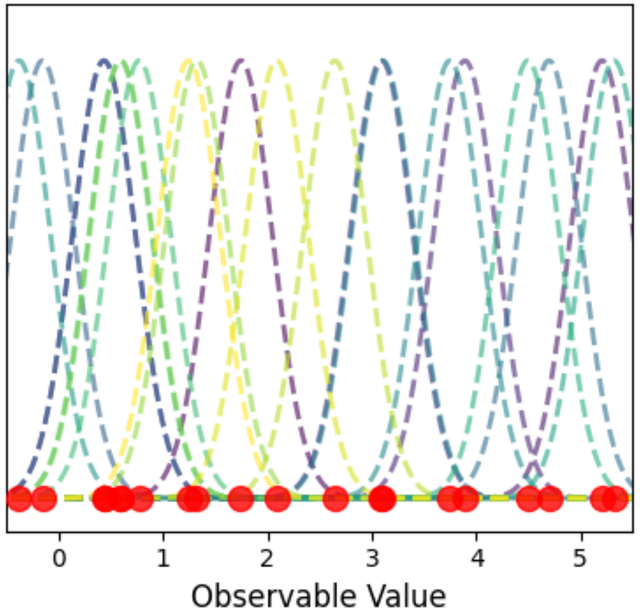


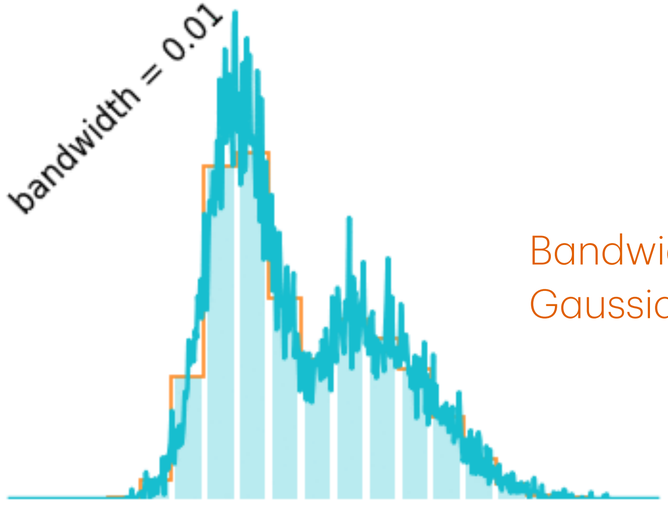
Differentiable Relaxations

Making histograms differentiable

Relaxation: binned kernel density estimate (bKDE)

Step 1: Define 1 Gaussian per-data point, and sum all Gaussians





Bandwidth controls Gaussian widths

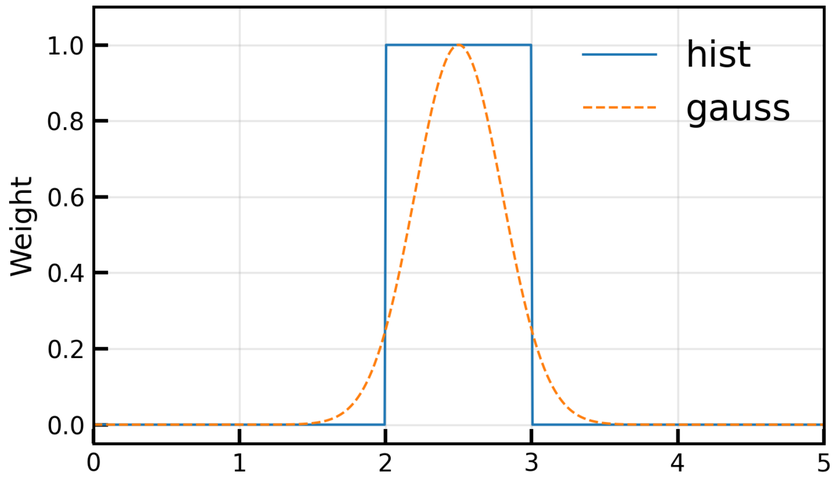
Animation Nathan Simpson's talk

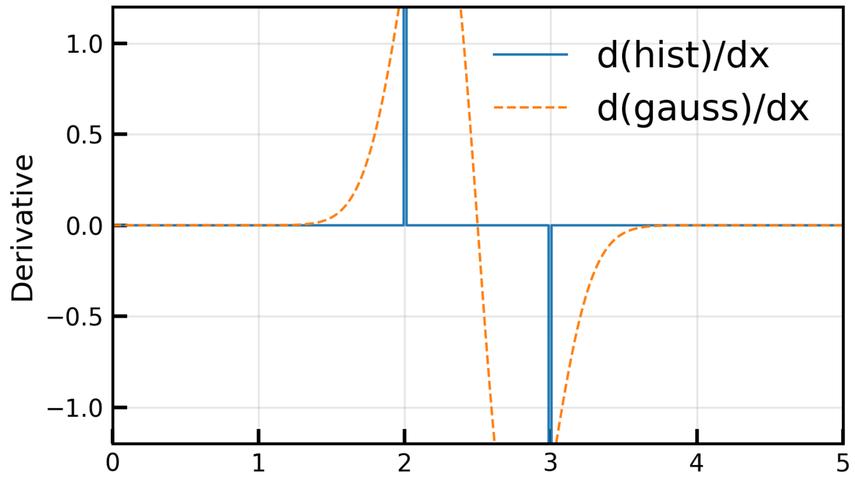
Differentiable Relaxations

Making histograms differentiable

Relaxation: binned kernel density estimate (bKDE)

Step 2: Integrate the Gaussian between bin-edges to get bin content





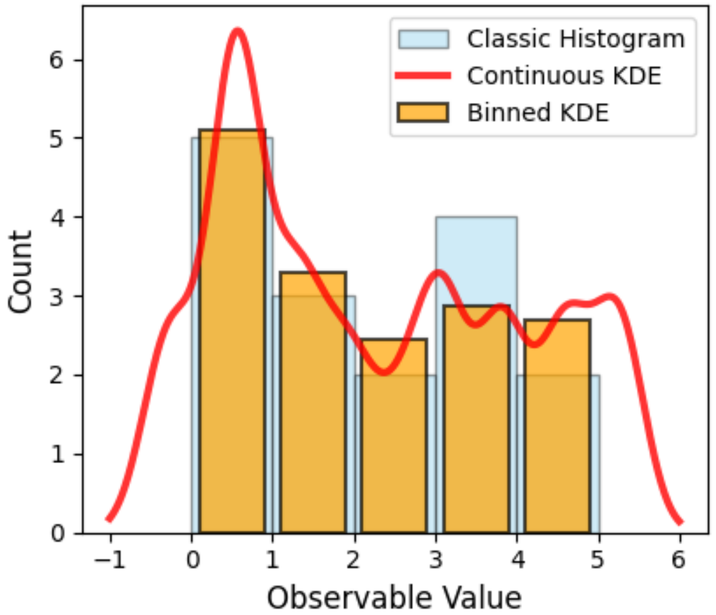
Differentiable Relaxations

Making histograms differentiable

Relaxation: binned kernel density estimate (bKDE)

Great tutorial on this by Nathan Simpson

Step 3: Differentiate each bin with respect to each data-point

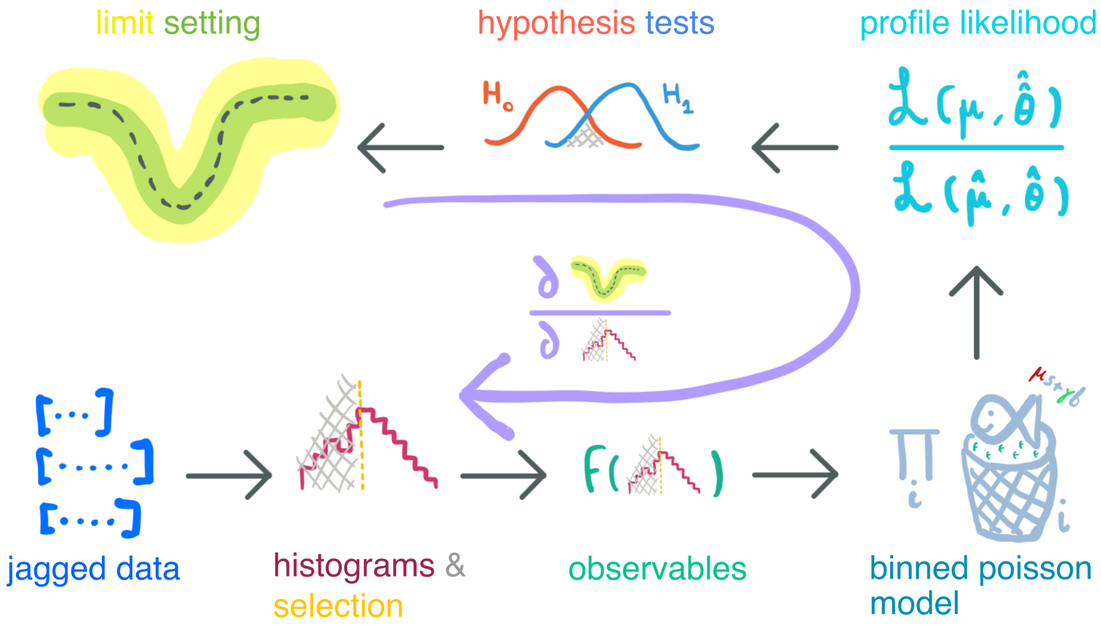


Fixed-Point Differentiation

Optimising the analysis while fitting to data

From Nathan Simpson's talk

With the gradient of the significance with respect to all parameters →
we can optimise all parameters to maximise significance



Fixed-Point Differentiation

Optimising the analysis while fitting to data

Caveat: the significance requires performing fits

Significance is a function of:

$$\frac{\mathcal{L}(\mu, \hat{\hat{\theta}})}{\mathcal{L}(\hat{\mu}, \hat{\theta})}$$

Run fit (nested minimiser) for each iteration in parameter optimisation?

Very expensive...

Fixed-Point Differentiation

Optimising the analysis while fitting a model to data

Solution from Implicit function theorem: → fixed-point differentiation

$F(x, y)$ ← A well-behaved function with a ROOT $F(x_0, y_0) = 0$

Implicit function theorem:

There exists an implicit function $\phi(x)$ such that $F(x, \phi(x)) = 0$ for $x \sim x_0$

with $$\frac{d\phi}{dx} = -\left(\frac{\partial F}{\partial y}\right)^{-1}\left(\frac{\partial F}{\partial x}\right)$$

Fixed-Point Differentiation

Optimising the analysis while fitting a model to data

Significance involves likelihood minimisation to find the best-fit model parameters $\theta^*(\alpha)$

$$\nabla_\theta \mathcal{L}(\alpha, \theta(\alpha)) = 0$$

Implicitly defines $\theta(\alpha)$ and hence $\frac{d\theta}{d\alpha}$

$$S(\alpha, \theta(\alpha)) \Rightarrow \frac{dS}{d\alpha} = \frac{\partial S}{\partial \alpha} + \frac{\partial S}{\partial \theta} \cdot \frac{d\theta}{d\alpha}$$

Plug in
More math in the back-up

Can compute gradient of the objective under the condition that we are at the likelihood minimum! Optimization in single loop!

Fixed-Point Differentiation

Optimising the analysis while fitting to data

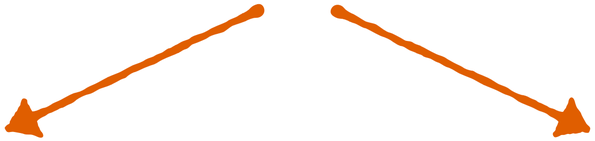
From Nathan Simpson's talk

The gradient of the objective is computed under the condition that we are at the likelihood minimum!

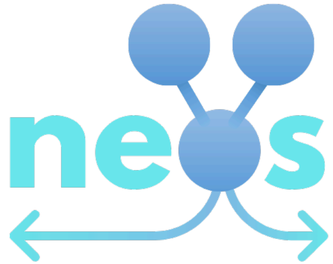
Where we stand

State of end-to-end differentiable HEP analysis

gradhep: collection of HEP differentiation tools



Differentiable relaxations



End-to-end differentiable analysis cookbook with relaxed and PyHF

Where we stand

State of end-to-end differentiable HEP analysis

Proof-of-concept for a differentiable analysis with simulated toy data → arXiv:2203.05570

neos: End-to-End-Optimised Summary Statistics for High Energy Physics

Nathan Simpson[1] **and Lukas Heinrich**[2]
[1]Lund University
[2]Technical University of Munich
E-mail: n.s@cern.ch, lukas.heinrich@cern.ch

"[...] a long term goal would be to scale neos in **application to open data from the major LHC experiments**, [...], allowing the probing of how the performance benefits scale to **realistic problems with many sources of uncertainty**."

Purpose of this work

What do we want to do

Goal #1

Scale concept of differentiable analysis to more realistic HEP data from major LHC experiment (CMS)

Goal #2

Review state + compatibility of existing tools and identify challenges to be addressed towards complete analysis

Applying gradients with CMS open-data

Case study: a $Z' \to t\bar{t}$ analysis

Differentiating through CMS open data

Diagram from CMS open-data workshop

- Searching for a hypothetical (heavy) BSM particle called Z' [arXiv:1810.05905]
  - Produced in proton-proton collisions, decays into a pair of top quarks

(resolved) (boosted)

Case study: a $Z' \to t\bar{t}$ analysis

Differentiating through CMS open data

Diagram from CMS open-data workshop

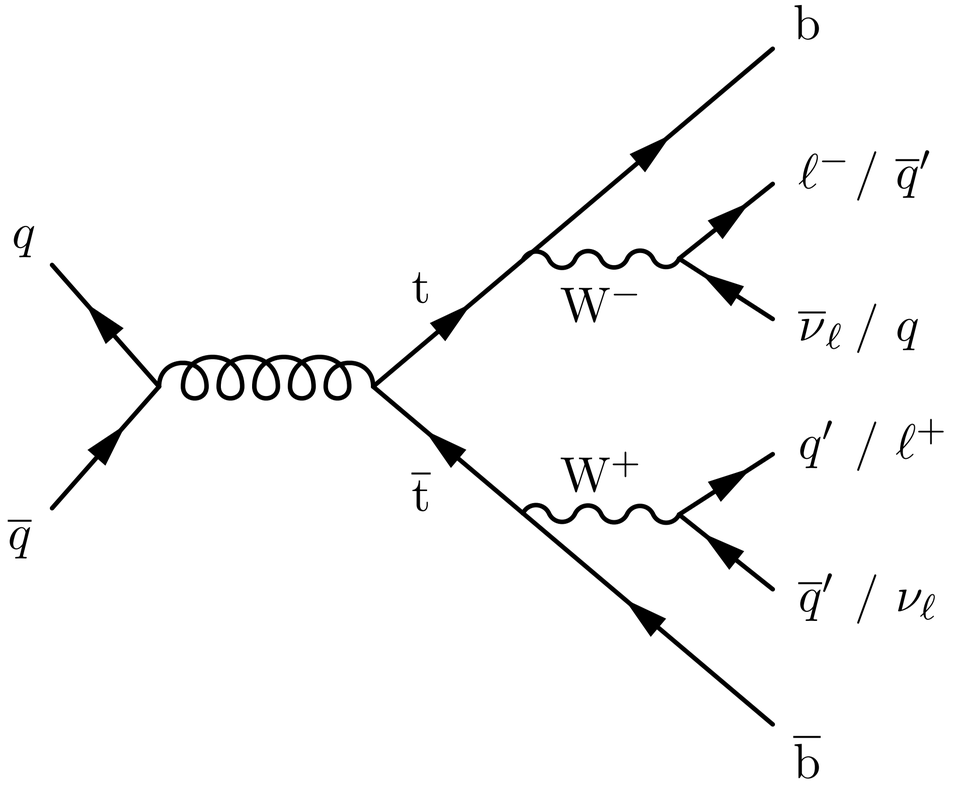
- Searching for a hypothetical (heavy) BSM particle called Z' [arXiv:1810.05905]
  - Produced in proton-proton collisions, decays into a pair of top quarks

(resolved) (boosted)

The simplified analysis

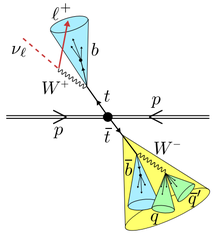
Based on CMS open-data workshop [2024]

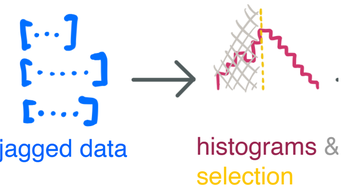
Main background: the production of top-quark pairs from the SM



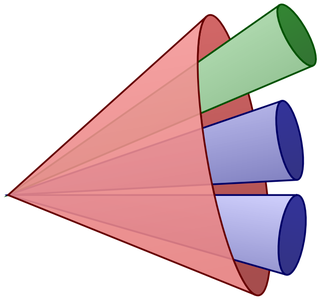
The simplified analysis

Based on CMS open-data workshop [2024]





Selecting the signal and dumping the background



Exactly 1 muon

MET > 50 GeV

> 1 b-tagged jet

Proxy to b-quark

Exactly 1 fat-jet

Proxy to top-quark

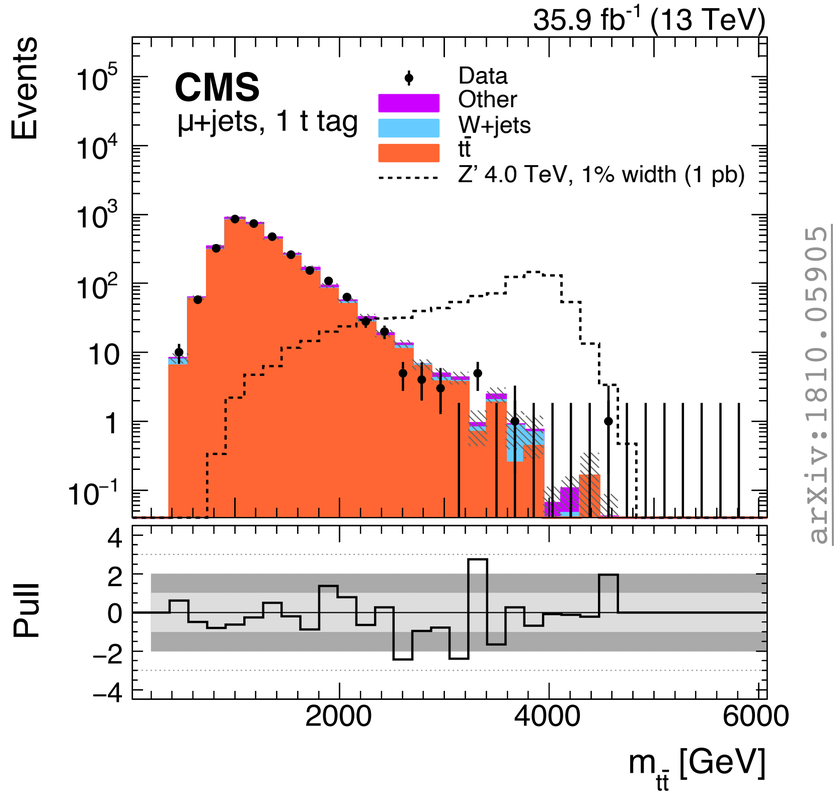
H_T = MET + muon momentum > 150 GeV

The simplified analysis

Based on CMS open-data workshop [2024]

observables

Building a discriminating observable: mass of the $t\bar{t}$ system



The simplified analysis

Based on CMS open-data workshop [2024]

profile likelihood

$$\frac{\mathcal{L}(\mu, \hat{\theta})}{\mathcal{L}(\hat{\mu}, \hat{\theta})}$$

Perform a binned profile likelihood fit to extract signal significance [p-value]

$$p(\vec{n}, \vec{a} \mid \vec{k}, \vec{\theta}) = \prod_{i \in N_{ch}} \text{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta})) \cdot \prod_{j \in N_\theta} c_j(a_j \mid \theta_j)$$

Bin-by-bin Poisson

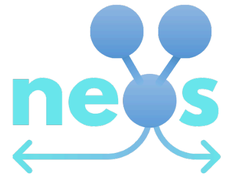
Priors of fit parameters

Differentiating the analysis

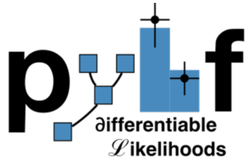
Tools we tried vs tools we used

Tried → Compatibility issues → **Used**

Awkward Array



relaxed



everlmore

relaxed

Awkward Array

VECTOR

Differentiating the analysis

Successes and compromises

Success #1: relaxing cuts and differentiable (KDE-based) histograms

```python
# Initialise a histogram
histogram = analysis.init_histogram()
# Read the data as an awkaward array
data = analysis.process_inputs(input_files)
# Define a cut on x > threshold
cut = data.x > threshold
# Filter the data
data_filtered = data[cut]
# Compute observable
observable = analysis.compute_observable(data_filtered)
# Extract histogram weights
event_weights = analysis.get_weights(data_filtered)
# Fill histogram
histogram.fill(observable, event_weights)
```

```python
import jax
import jax.numpy as jnp
# Read the data as an awkaward array
data = analysis.process_inputs(input_files)
# Define a relaxed version of the hard cut with tempreture T
soft_weight = jax.nn.sigmoid((data.x - threshold)/T)
# Compute observable
observable = analysis.compute_observable(data_filtered)
# Extract histogram weights
event_weights = analysis.get_weights(data_filtered)
# Compute a KDE-based histogram
cdf = jax.scipy.stats.norm.cdf(
    binning,
    observable,
    scale=bandwidth,
)
weighted_cdf = cdf * soft_weight * weights
bin_weights = weighted_cdf[1:, :] - weighted_cdf[:-1, :]
diff_histogram = jnp.sum(bin_weights, axis=1)
```

Differentiating the analysis

Successes and compromises

Compromise:

Cannot use jagged data with pure JAX → limited cut and observable implementations

Differentiating the analysis

Successes and compromises

Compromise:

Cannot use jagged data with pure JAX → limited cut and observable implementations

Awkward arrays have a JAX backend
but the functionality is still evolving
[see Saransh's talk]

Differentiating the analysis

Successes and compromises

Success #2: writing a pure-JAX statistical model to pass to **relaxed**

```python
# Extraa# 1) JIT-compiled Poisson log-PDF
@jax.jit
def poisson_ll(n, lam):
    return n * jnp.log(lam + 1e-12) - lam - jsp.special.gammaln(n + 1.0)

# 2) A model compatible with relaxed with manual scaling
class AllBkg(eqx.Module):
    obs: jnp.ndarray
    bkg: jnp.ndarray
    sig: jnp.ndarray

    def expected_data(self, pars):
        μ = pars["mu"]
        θ = pars["tt"]
        return [μ * self.sig + θ * self.bkg], []

    def logpdf(self, data, pars):
        (obs_list, _ ) = data
        exp_list, _ = self.expected_data(pars)
        return jnp.sum(poisson_ll(obs_list[0], exp_list[0]))
```

```python
# 3) Dummy histograms
obs = jnp.array([12., 18., 25., 30.])
bkg = jnp.array([10., 10., 10., 10.])
sig = jnp.array([1., 2., 3., 4.])

# 4) Instantiate model and run hypotest in pure JAX
model = AllBkg(obs=obs, bkg=bkg, sig=sig)
data  = ([obs], [])
p0, mle = relaxed.infer.hypotest(
    test_mu      = 0.0,
    data         = data,
    model        = model,
    init_pars    = {"mu": 1.0, "tt": 1.0},
    return_mle_pars = True,
    test_stat    = "q0",
)
```

relaxed handles implicit differentiation!

Differentiating the analysis

Successes and compromises

Compromise:

Tedious to implement nuisance parameter modifiers from scratch

Differentiating the analysis

Successes and compromises

Compromise:

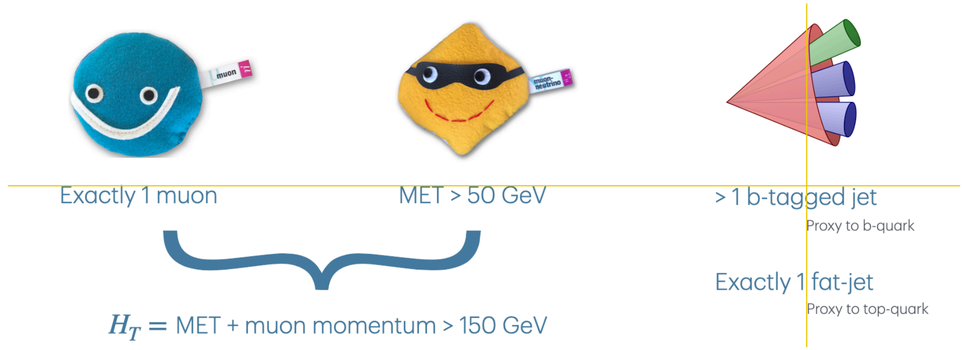
Tedious to implement nuisance parameter modifiers from scratch

hence we did not yet implement
systematic uncertainties

Optimisation

Physics objectives and parameters to optimise

- Compute the derivative of the significance with respect to:
  - The MET cut threshold
  - The b-tagging score threshold used to count b-jets
  - The H_T cut threshold



- Optimise the signal significance via the p-value
  - Smaller p-value, higher significance

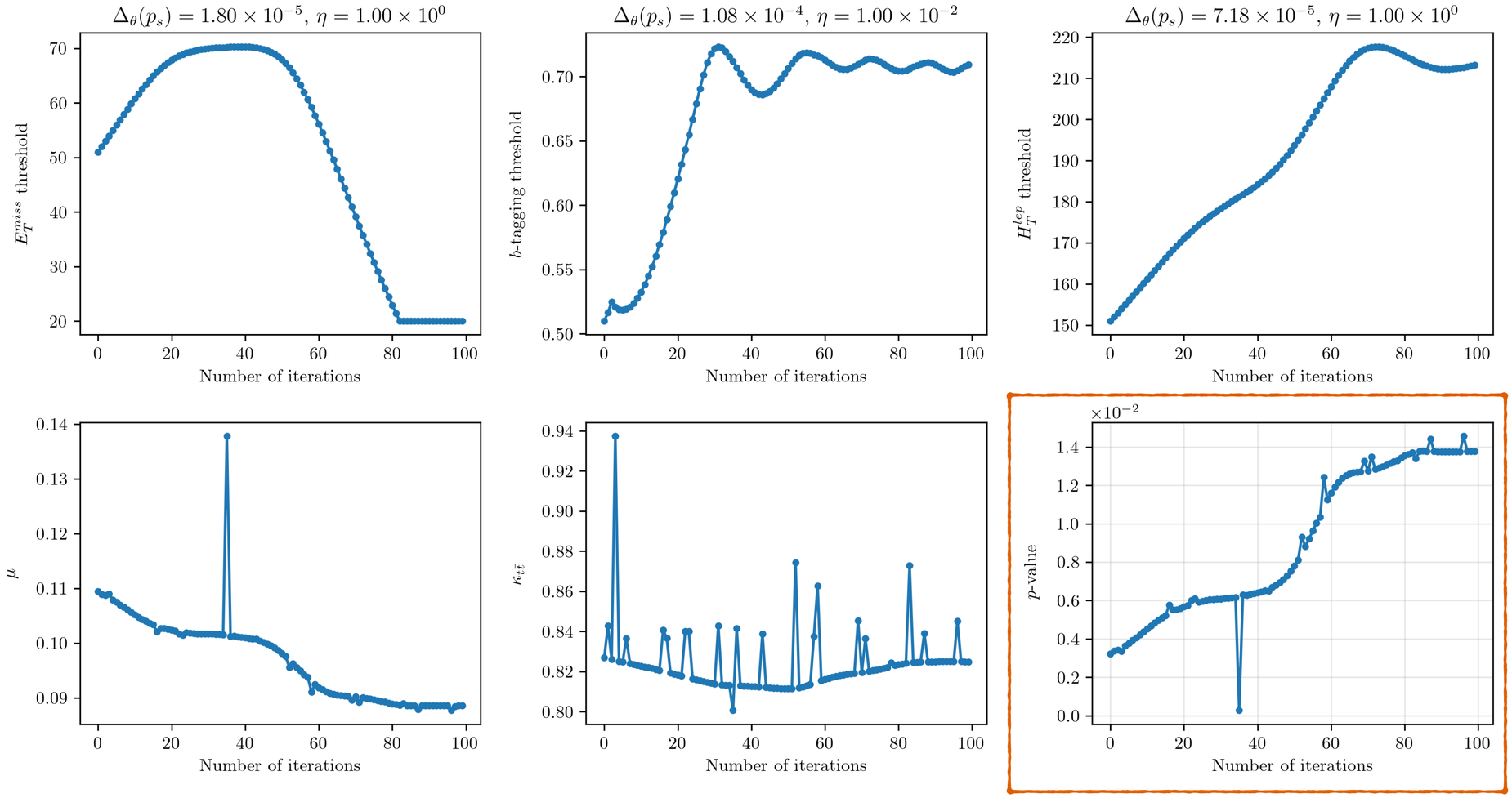
Optimisation
The gradient descent

- **jaxopt.OptaxSolver** for back-propagation
- Implement parameter-wise learning rates and clipping functions
- Implicit differentiation of fit parameters in **relaxed**

Optimisation

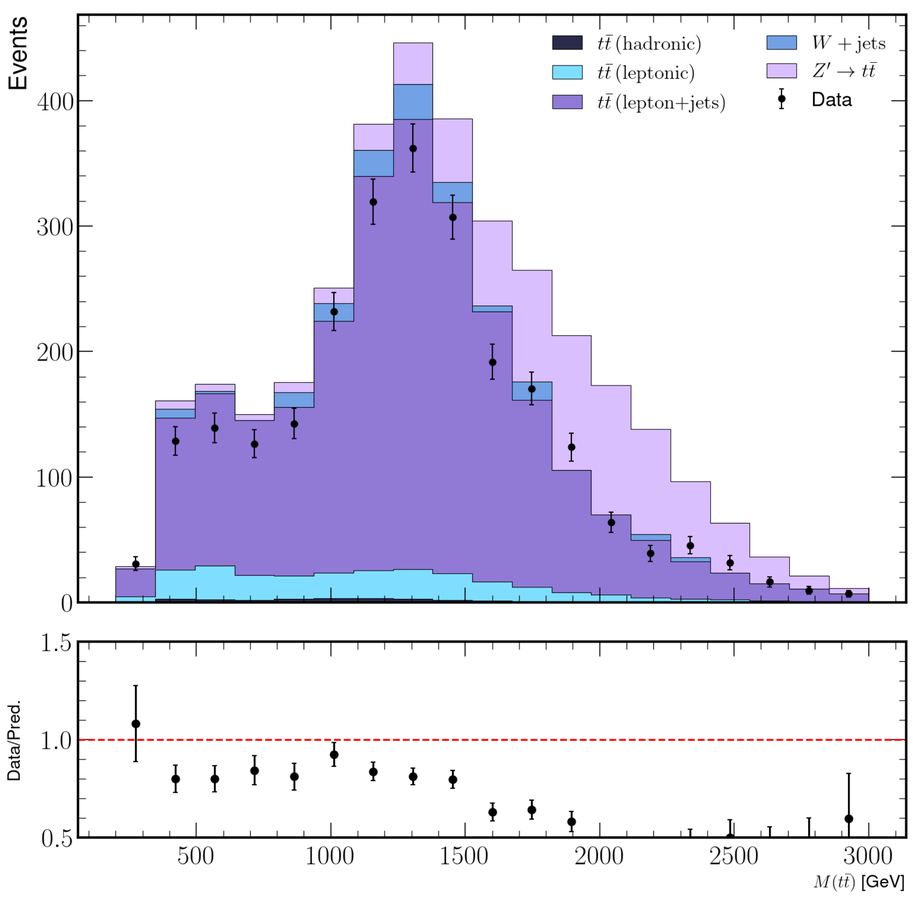
The gradient descent

- 100 iterations in ~3 minutes without JIT
- JIT optimisations were difficult to implement
- 320% improvement in significance

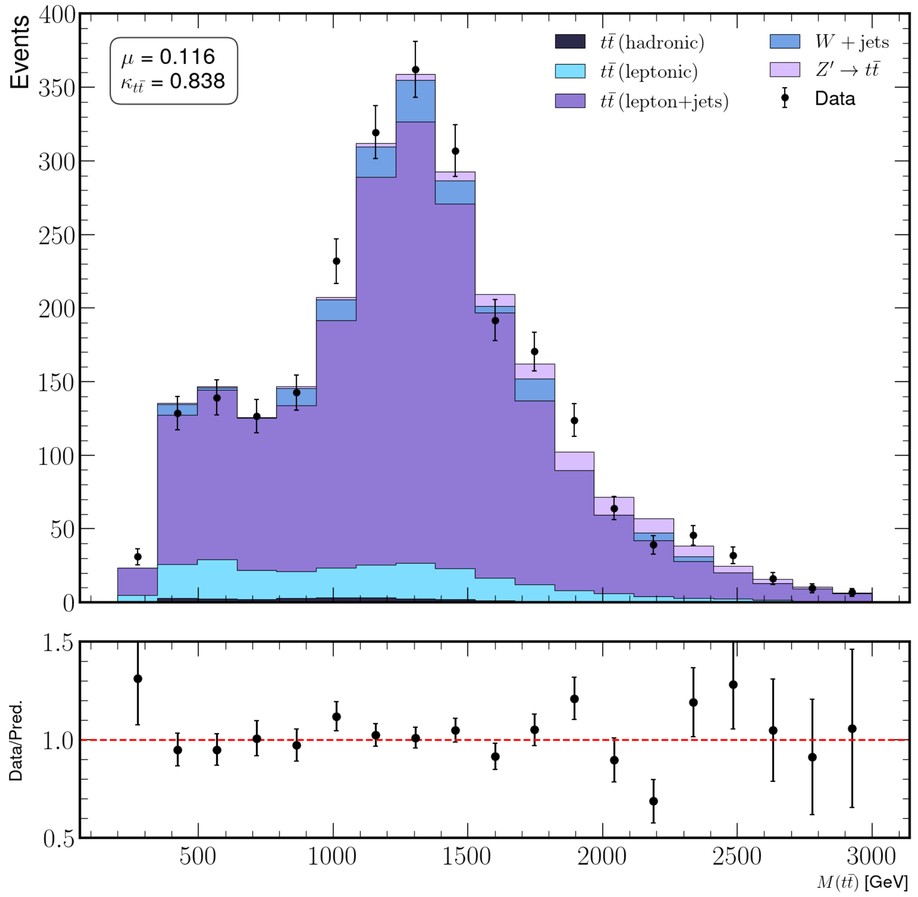


Optimisation

The fit results



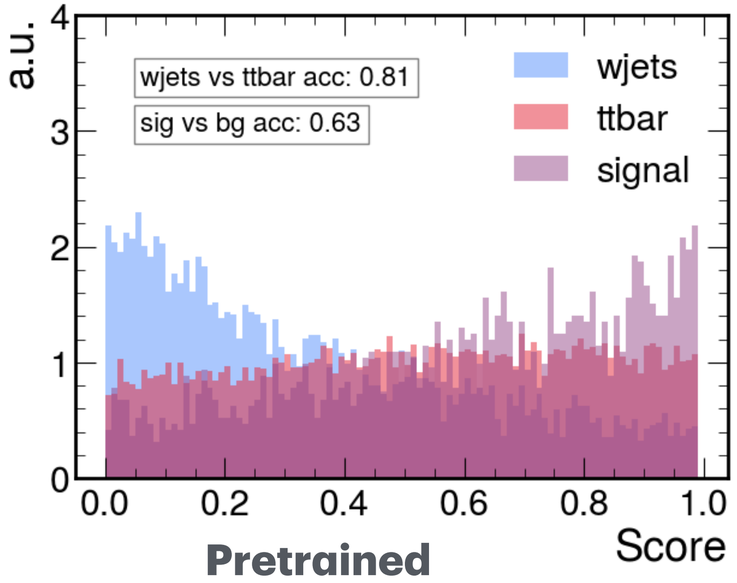
Holistic Optimisation

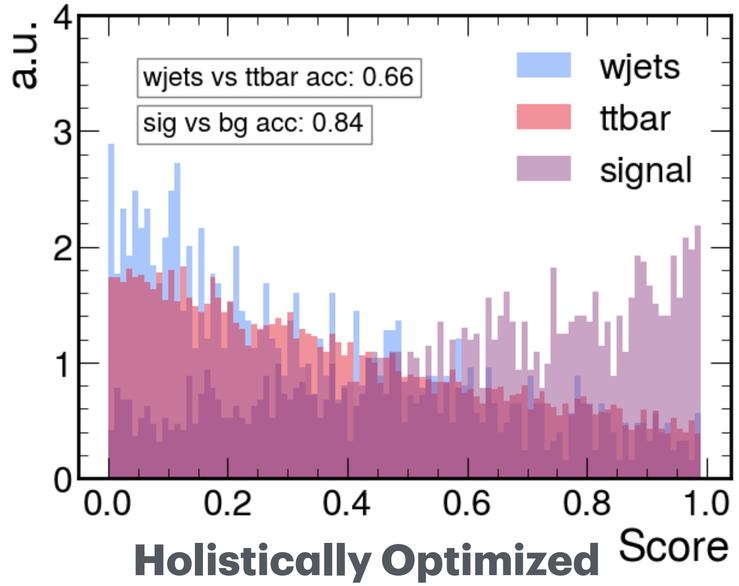


MVA Classifier

Holistically Optimizing ML steps

- Pre-train simple wjets / ttbar classifier from high-level kinematic variables
- Apply soft-cut on low value (0.05) to suppress wjets in signal region
  - Optimize NN's parameters as part of the entire pipeline (small learning rate)





- Re-training the wjets-vs-ttbar classifier in the holistic optimization turns it into a signal-vs-background classifier!
- No ground-breaking insight, but demonstration of feasibility to optimize O(1k) parameters!

Conclusions

Goal #1

Scale concept of differentiable analysis to more realistic HEP data from major LHC experiment (CMS)

Goal #2

Review state + compatibility of existing tools and identify challenges to be addressed towards complete analysis

Way forward

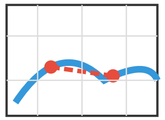
- Gradually make differentiable example more realistic
  - Also fit control regions, include systematic uncertainties
  - Optimize w.r.t more-and-more parameters
  - Include more realistic ML algorithms in the pipeline
- Eventually optimize (and publish) an actual physics measurement (w/ approval from collaboration)
- Document all missing features, bugs, and compatibility issues in GitHub issues
- Discuss with community how to proceed towards a differentiable Scikit-HEP ecosystem

Backup

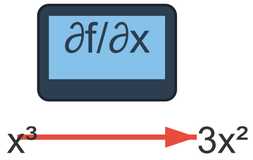
What is differentiable programming?

Four ways to calculate derivates in a computer program

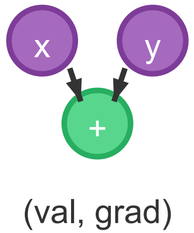
- Manual differentiation (calculate by hand and implement in code)



- Numerical differentiation (finite difference approximation)
  - Potential rounding errors, scales poorly w/ number of parameters



- Symbolic differentiation (program calculates analytical derivative)
  - Equations can become complex (“expression swell”)



- Automatic differentiation (store gradients, apply chain rule)
  - Key idea behind backpropagation

Fixed-Point Differentiation

Fitting the model and optimising the analysis - two loops?

- Physics objective S depends on fit result:

$$S\left(\theta^*(\alpha), \alpha\right) \Rightarrow \frac{dS}{d\alpha} = \frac{\partial S}{\partial \alpha} + \frac{\partial S}{\partial \theta^*} \cdot \frac{d\theta^*}{d\alpha}$$

Best fit result

- Two optimisation loops? No, there is a trick!

Implicit function theorem

$$f(x, y) = 0 \Rightarrow \exists \phi : f(x, \phi(x)) = 0 \text{ and } \frac{d\phi}{dx} = -\left(\frac{\partial f}{\partial y}\right)^{-1}\left(\frac{\partial f}{\partial x}\right)$$

Best fit Result:

$$\nabla_\theta \mathscr{L}(\theta^*(\alpha), \alpha) = 0 \Rightarrow \frac{d\theta^*}{d\alpha} = -\left(\frac{\partial^2 \mathscr{L}}{\partial \alpha \partial \theta}\right)^{-1}\left(\frac{\partial^2 \mathscr{L}}{\partial \theta^2}\right)$$

Combined Result

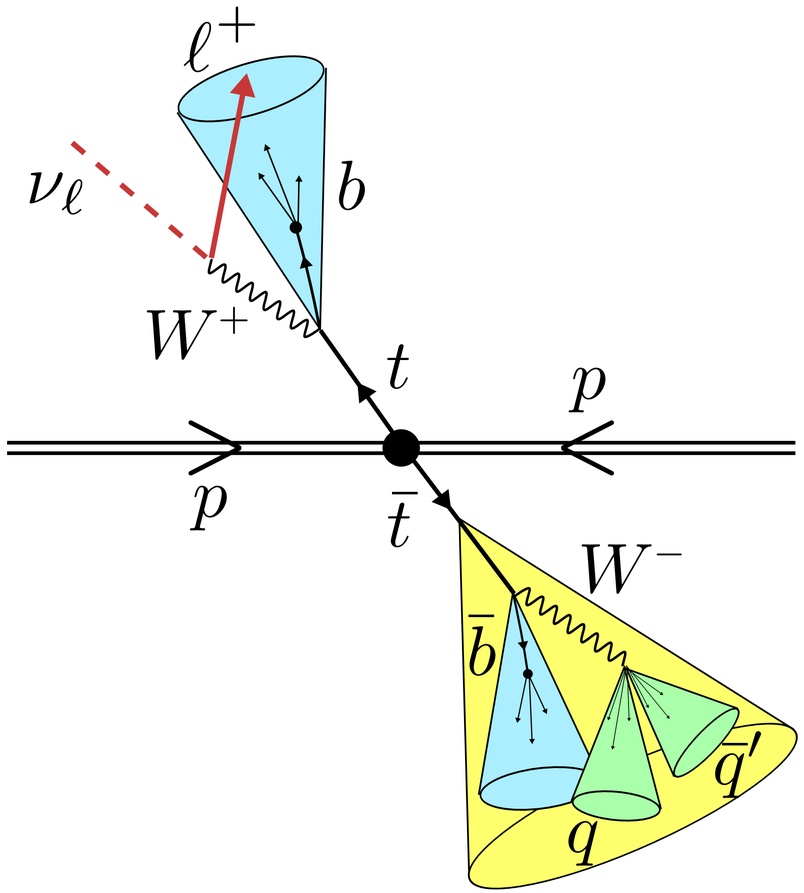
$$\frac{dS}{d\alpha} = -\frac{\partial S}{\partial \alpha} - \frac{\partial S}{\partial \theta^*} \cdot \left(\frac{\partial^2 \mathscr{L}}{\partial \theta^2}\right)^{-1} \cdot \left(\frac{\partial^2 \mathscr{L}}{\partial \theta \partial \alpha}\right)$$

- Gradient of Objective under condition that the likelihood is stationary
- Can run gradient descent along the best-fit manifold. Very neat!

Dissecting the process

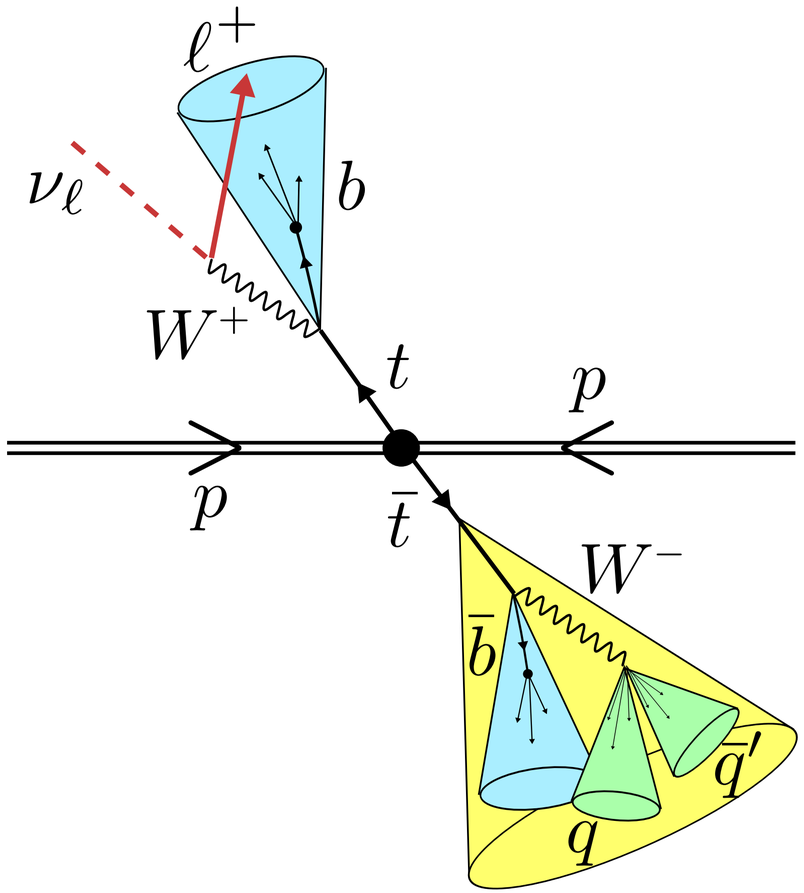
Getting the terminology right

Diagram from CMS open-data workshop



Dissecting the process

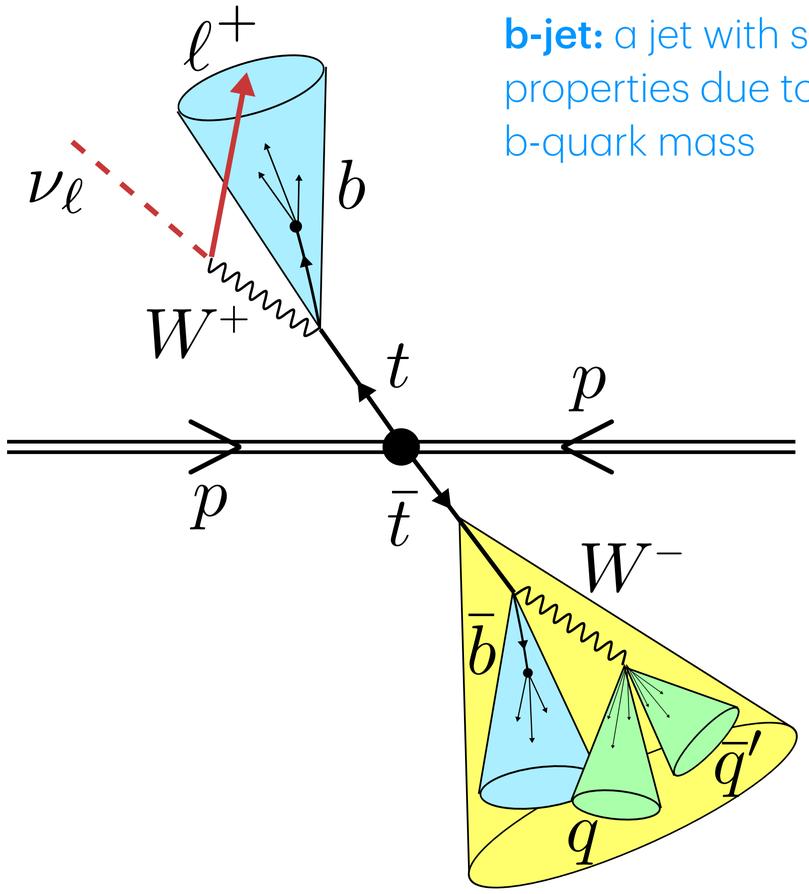
Getting the terminology right



jets: due to quarks *"fragmenting"* then *"hadronising"* into bound states (*hadrons*)

Dissecting the process

Getting the terminology right

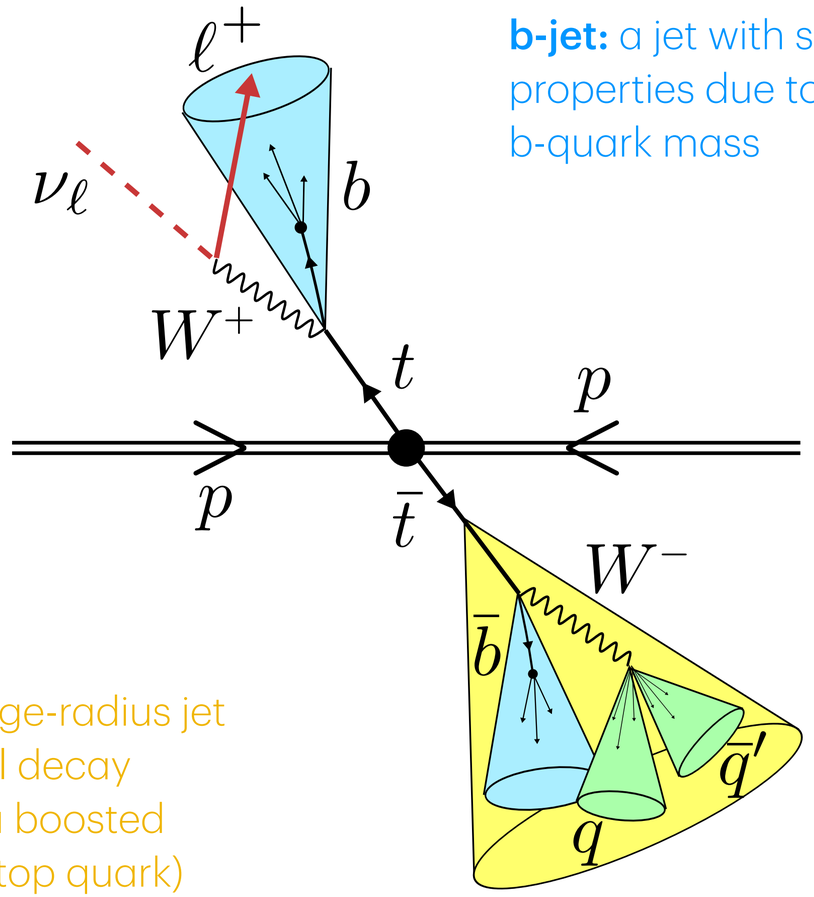


b-jet: a jet with special properties due to the heavy b-quark mass

jets: due to quarks *"fragmenting"* then *"hadronising"* into bound states (*hadrons*)

Dissecting the process

Getting the terminology right



b-jet: a jet with special properties due to the heavy b-quark mass

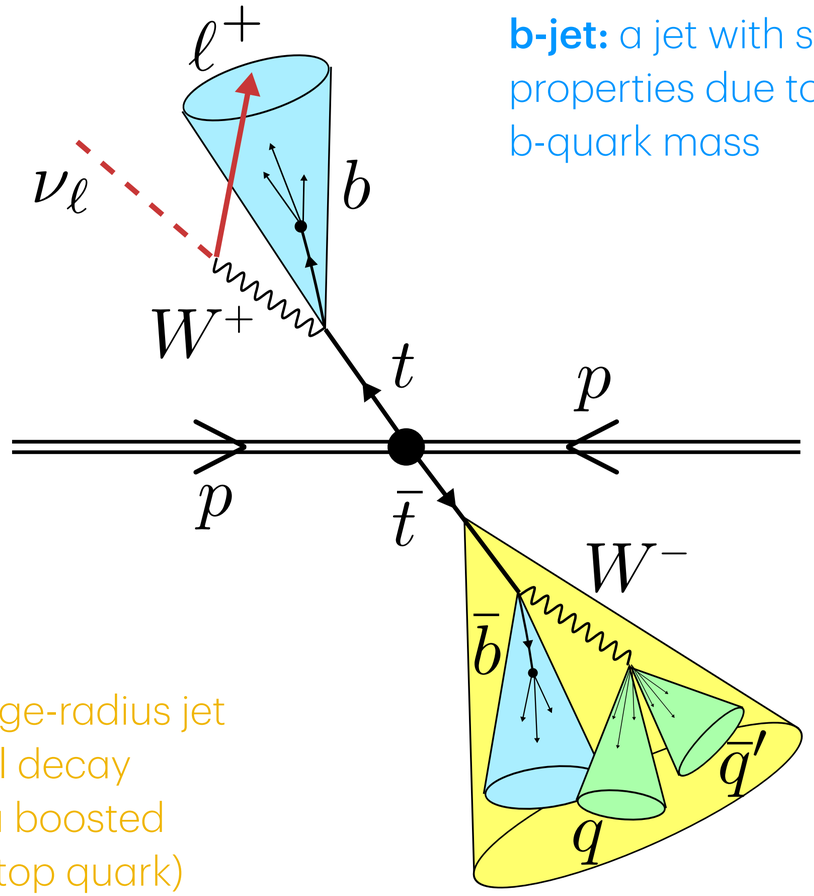
fat-jets: a large-radius jet containing all decay products of a boosted particle (e.g. top quark)

jets: due to quarks *"fragmenting"* then *"hadronising"* into bound states (*hadrons*)

Dissecting the process

Getting the terminology right

neutrinos: undetectable in CMS, their presence inferred via momentum conservation as missing energy (MET)



b-jet: a jet with special properties due to the heavy b-quark mass

fat-jets: a large-radius jet containing all decay products of a boosted particle (e.g. top quark)

jets: due to quarks *"fragmenting"* then *"hadronising"* into bound states (*hadrons*)

The simplified analysis

Based on CMS open-data workshop [2024]

profile likelihood

$$\frac{\mathcal{L}(\mu, \hat{\theta})}{\mathcal{L}(\hat{\mu}, \hat{\theta})}$$

hypothesis tests

Likelihood computed from observed data and parameterised model

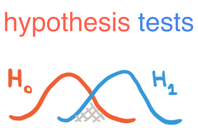
$$\prod_{i \in N_{ch}} \text{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta}))$$

The simplified analysis

Based on CMS open-data workshop [2024]

profile likelihood

$$\frac{\mathcal{L}(\mu, \hat{\theta})}{\mathcal{L}(\hat{\mu}, \hat{\theta})}$$



Likelihood computed from observed data and parameterised model

$$\prod_{i \in N_{ch}} \text{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta}))$$

Observed data

The simplified analysis

Based on CMS open-data workshop [2024]

profile likelihood

$$\frac{\mathcal{L}(\mu, \hat{\theta})}{\mathcal{L}(\hat{\mu}, \hat{\theta})}$$

hypothesis tests

Likelihood computed from observed data and parameterised model

$$\prod_{i \in N_{ch}} \text{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta}))$$

Observed data

Prediction which depends on parameters of interest $\vec{k}$ and nuisance parameters $\vec{\theta}$

The simplified analysis

Based on CMS open-data workshop [2024]

profile likelihood

$$\frac{\mathcal{L}(\mu, \hat{\hat{\theta}})}{\mathcal{L}(\hat{\mu}, \hat{\theta})}$$

hypothesis tests

Likelihood computed from observed data and parameterised model

$$\prod_{i \in N_{ch}} \text{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta}))$$

Observed data

Prediction which depends on parameters of interest $\vec{k}$ and nuisance parameters $\vec{\theta}$

The signal prediction is parameterised via a signal strength μ

The simplified analysis

Based on CMS open-data workshop [2024]

profile likelihood

$$\frac{\mathcal{L}(\mu, \hat{\theta})}{\mathcal{L}(\hat{\mu}, \hat{\theta})}$$

Perform a binned profile likelihood fit to extract signal significance [p-value]

$$p(\vec{n}, \vec{a} \mid \vec{k}, \vec{\theta}) = \prod_{i \in N_{ch}} \text{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta})) \cdot \underbrace{\prod_{j \in N_\theta} c_j(a_j \mid \theta_j)}$$

Constraint terms for systematic uncertainties from auxiliary measurement

Differentiating the analysis

Tools that did not work very well together

Evermore **model is incompatible with what** relaxed **expects, and tracing issues arise when a model wrapper is attempted**

Differentiating the analysis

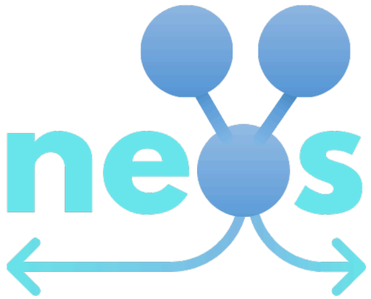
Tools that did not work very well together

Awkward Array

relaxed **is unable to handle awkward arrays during data processing**

Differentiating the analysis

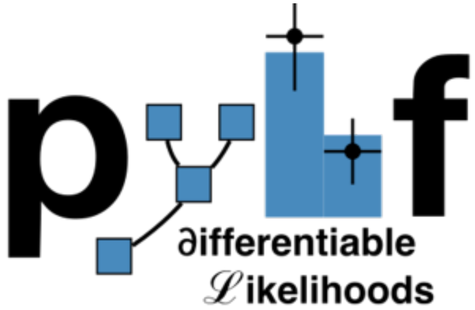
Tools that did not work very well together



Difficult to navigate version clashes between old dependencies

Differentiating the analysis

Tools that did not work very well together



JAX backend support is limited in PyHF — difficulties in model building

Differentiating the analysis

Tools that did not work very well together

Awkward Array

JAX **backend support is limited in** awkward array **— many missing operations**