LPC EFT Workshop: Sept 6, 2023

Hands-On Tutorial: EFT-Aware Histograms

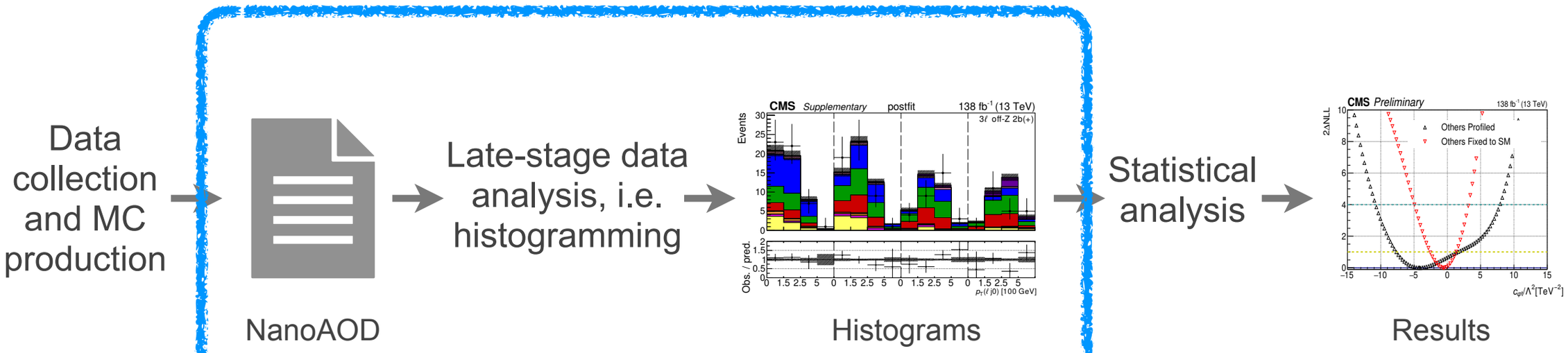
Kelci Mohrman (kelci.mohrman@cern.ch)

With thanks to the tutorial organizers, the workshop organizers, the TOP-22-006 team, and the ND CCL

Introduction

The big-picture goal is to compare EFT prediction to data in order to extract confidence intervals for the Wilson Coefficients (WCs), involves three main steps:

1. Generate MC that incorporates the EFT into the prediction (MC generation tutorial)
2. Perform selection to obtain the events of interest, summarized in histogram objects (this tutorial!)
3. Perform statistical analysis to compare the prediction to the observation and extract confidence intervals (statistical tools tutorial)



In this tutorial we'll focus on the histogramming step, discussing concepts and tools relevant for EFT analyses

Tutorial outline

- Review of EFT reweighting concepts
- Hands-on example of extracting quadratic from reweight points
- Discussion of EFT histograms concepts
- Hands-on example of using an EFT-aware histogram

Review: How do observables depend on EFT?
Let's start with the cross section, σ

EFT modeled linearly in amplitude:

$$\mathscr{M} = \mathscr{M}_{SM} + \sum_i c_i \mathscr{M}_i \quad \leftarrow \; c_i \text{ are the WCs}$$

So the σ depends as a quadratic in terms of the WCs:

$$\sigma(c_1) \propto |\mathscr{M}|^2 \propto \underset{\text{SM}}{s_0} + \underset{\text{Interference with SM}}{s_1 c_1} + \underset{\text{Pure NP}}{s_2 c_1^2} =$$

For *n* WCs, we will have an *n*-dimensional quadratic

A more general case

- With multiple WCs, there are more terms (but it's still quadratic)

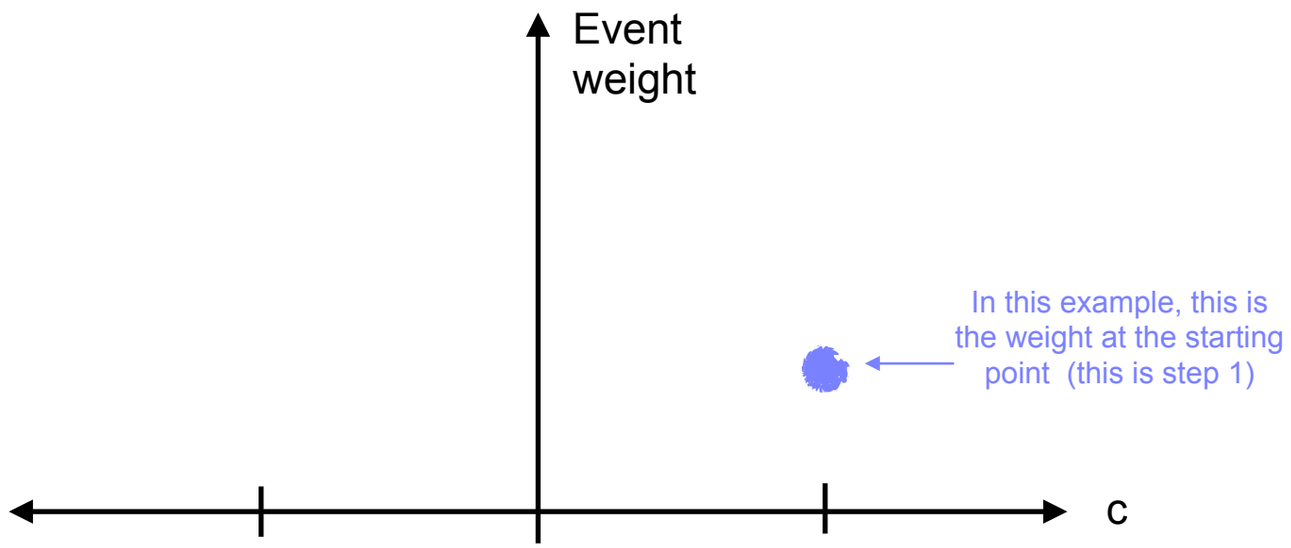
$$\sigma(\vec{c}) = s_0 + \sum_i s_{1i}\frac{c_i}{\Lambda^2} + \sum_i s_{2i}\frac{c_i^2}{\Lambda^4} + \sum_{i\neq j} s_{3ij}\frac{c_i}{\Lambda^2}\frac{c_j}{\Lambda^2}$$ ← ↻

- There are $((n+1)^2 - (n+1))/2 + n + 1$ terms in an *n*-dimensional quadratic function, e.g. 6 terms for 2 WCs, or 378 terms for 26 WCs
- This dependence is true for any xsec, inclusive or differential
- Since an event weight is essentially a small piece of the cross section, the weight of each generated event also depends as an n-dimensional quadratic in terms of the WCs

weight of each event = ↻

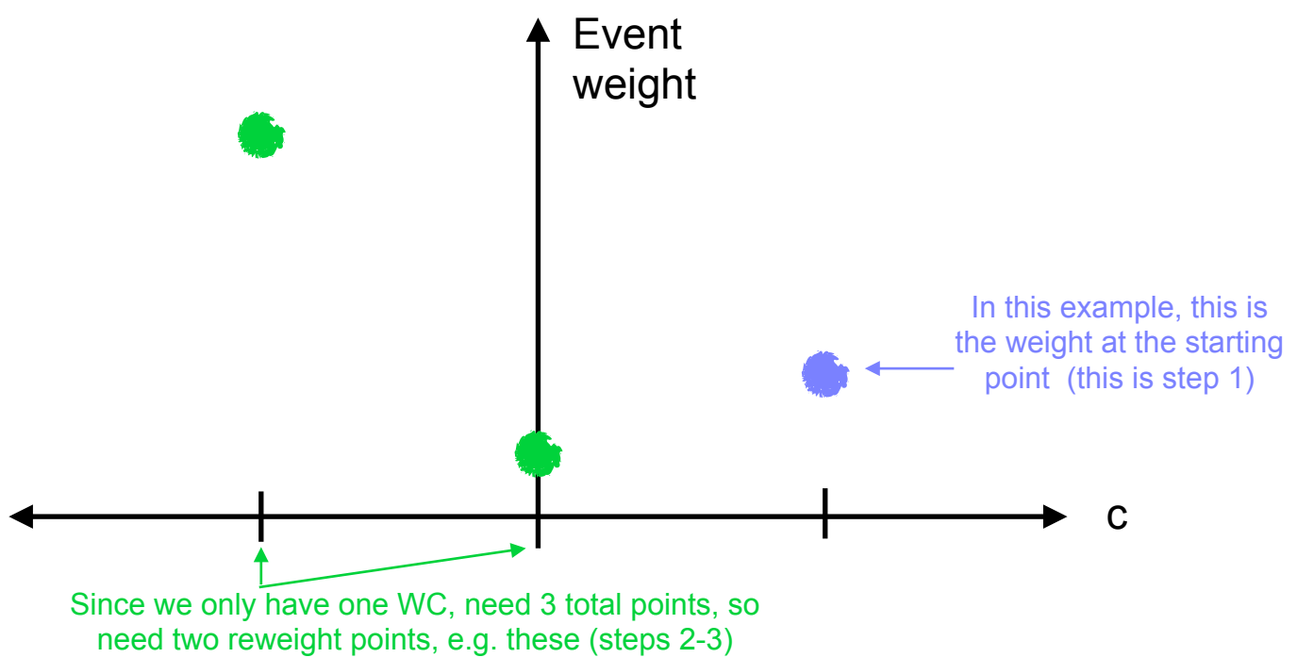
Extracting the quadratic dependence for a single event

- How do we find the quadratic dependence for each of the generated events?
- Use MG reweighting, as introduced in the previous tutorial, i.e.:
    1. Pick a "starting point" in the WC space, and MG generates an event (at some point in kinematic space) under the assumption of the given point in WC space (e.g. a "c=1" assumption)



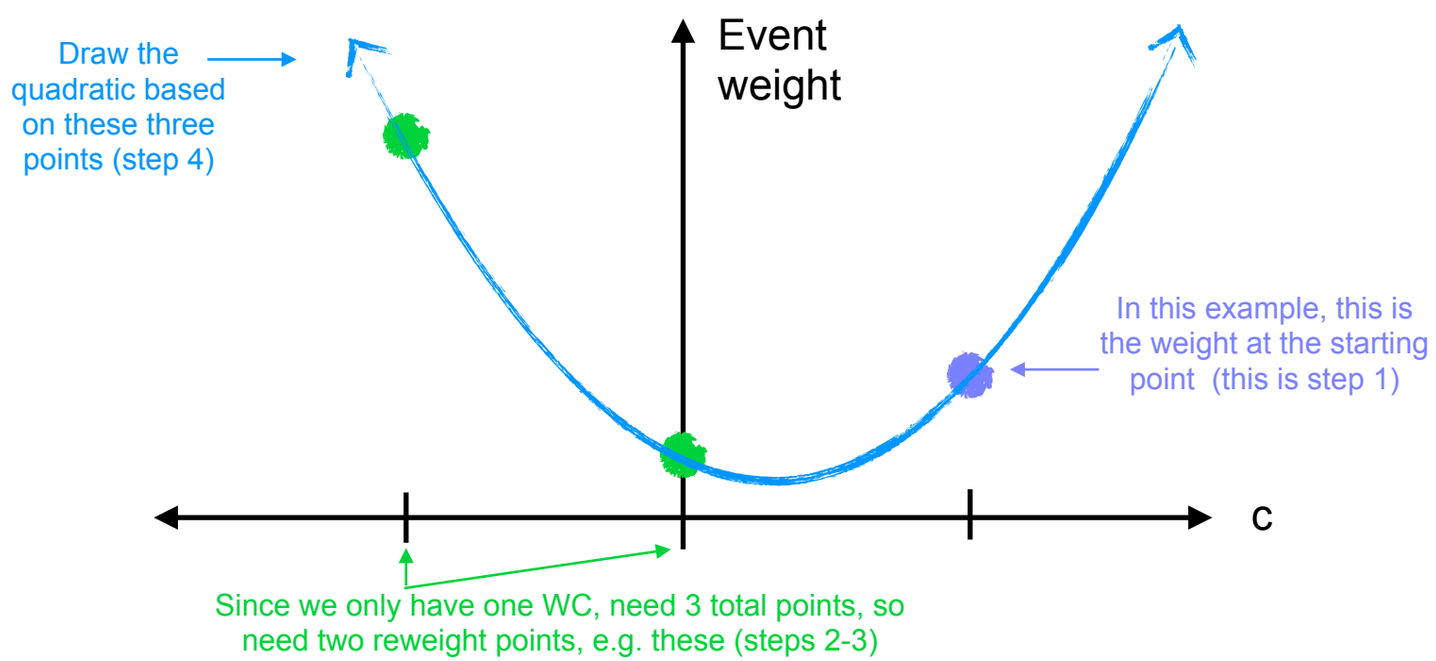
Extracting the quadratic dependence for a single event

- How do we find the quadratic dependence for each of the generated events?
- Use MG reweighting, as introduced in the previous tutorial, i.e.:
    1. Pick a "starting point" in the WC space, and MG generates an event (at some point in kinematic space) under the assumption of the given point in WC space (e.g. a "c=1" assumption)
    2. Ask MG "what would the weight of this event have been at a different point in the WC space?"
    3. Repeat step 2 for at least $((n+1)^2 - (n+1))/2 + n + 1$ points in the WC space



Extracting the quadratic dependence for a single event

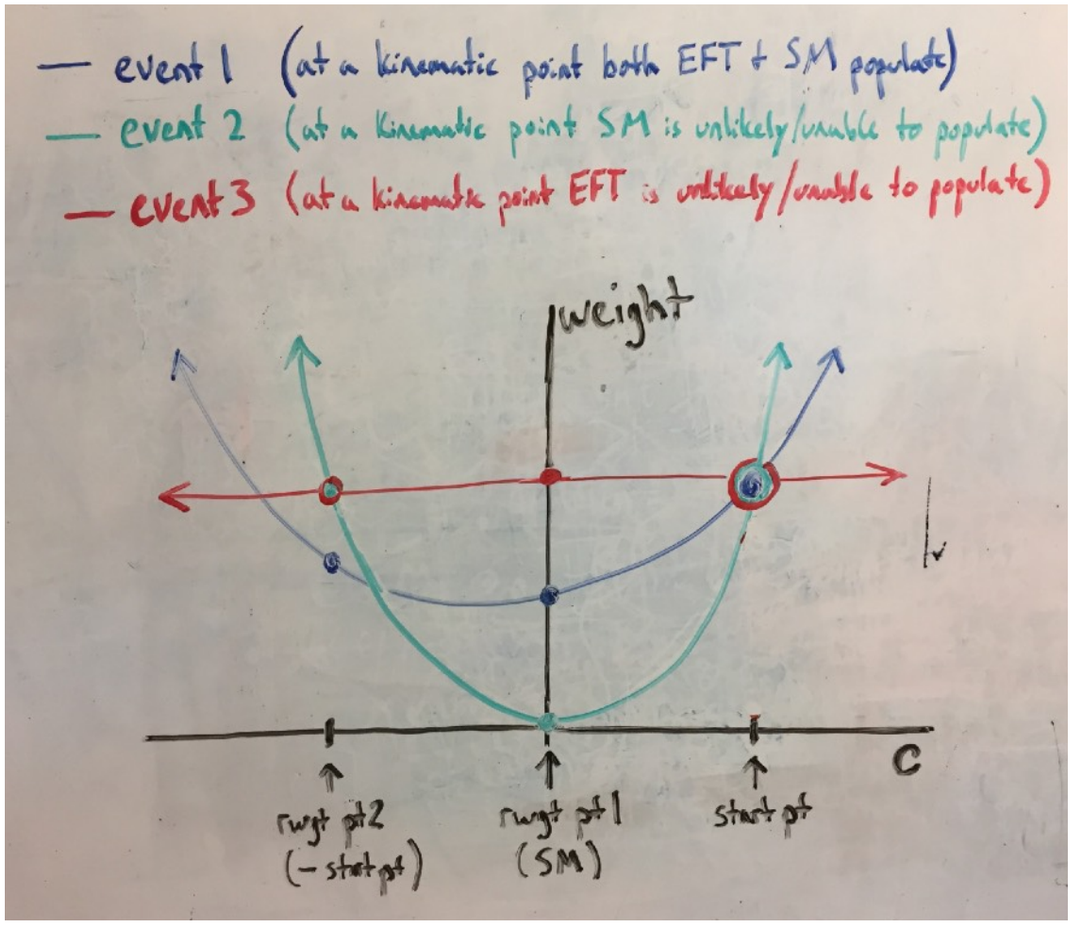
- How do we find the quadratic dependence for each of the generated events?
- Use MG reweighting, as introduced in the previous tutorial, i.e.:
    1. Pick a "starting point" in the WC space, and MG generates an event (at some point in kinematic space) under the assumption of the given point in WC space (e.g. a "c=1" assumption)
    2. Ask MG "what would the weight of this event have been at a different point in the WC space?"
    3. Repeat step 2 for at least $((n+1)^2 - (n+1))/2 + n + 1$ points in the WC space
    4. From the set of points in WC space and the associated weights, extract the quadratic parameterization



Why do different events have different quadratic shapes?

- Recall that MG will generate each event at a different kinematic point
- The kinematic point will be relatively less/more likely to be populated based on the theory assumption (i.e. at which point in WC space we are sitting)
- A complication to remember: Due to MG unweighting, the weight at the starting point will always be of the same magnitude (regardless of the differences in kinematics)

In this conceptual example, we're exploring different quadratic shapes we might see for three different simulated events →



This is somewhat difficult to conceptualize (at least for me) but remember that at the starting point, differences in probability due to different kinematics are conveyed by *how many events are generated at a given phase space point*, rather than by the *weight* of the given event at the given point in the space

Summary and some caveats

- Summary: If you have a sufficient number of reweighs points, you can extract the quadratic parametrization for each event's weight, which allows you to know the value of the event weight at any arbitrary point in the EFT space

- This can be a powerful approach for several reasons:
    - Allows essentially arbitrary regions in the EFT space to be probed with just a single sample
    - Allows the full effects of the EFT on kinematics to be accounted for
    - If the weights are carried through to detector level, allows EFT effects on acceptance/efficiency to be incorporated

- Caveats:
    - Vitally crucial to thoroughly validate the reweighted samples to ensure the sample can be consistently reweighted throughout the relevant EFT space
    - Important to explore the statistical power of the sample (highly non-uniform event weights degrade the statistical power)
    - Computationally challenging to produce samples with many WCs

Tutorial outline

- Review of EFT reweighting concepts
- Hands-on example of extracting quadratic from reweight points
- Discussion of EFT histograms concepts
- Hands-on example of using an EFT-aware histogram

Extracting the quadratic dependence, a toy example

- Let's say we have just two WCs, called c_1 and c_2
- We thus need 6 reweight points: $((n+1)^2 - (n+1))/2 + n + 1 \;|_{n=2} = 6$
- Let's say we run MG and get the following weights at the following points:

c1	c2	weight
0	0	1.000
0	1	0.909
5	0	1.403
5	10	0.721
-5	10	0.333
-10	10	0.418
10	10	1.194

(Notice that we have one more point than we need! This will let us make sure that this shape indeed looks quadratic)

- What we want to find are the structure constants (let's call them $\vec{s}$), given the set of reweight points and weights, i.e.: $\mathbf{A}\,\vec{s} = \vec{w}$

$$\mathrm{A} = \begin{bmatrix} 1 & (c_1)_0 & (c_2)_0 & (c_1^2)_0 & (c_2^2)_0 & (c_1c_2)_0 \\ 1 & (c_1)_1 & (c_2)_1 & (c_1^2)_1 & (c_2^2)_1 & (c_1c_2)_1 \\ 1 & (c_1)_2 & (c_2)_2 & (c_1^2)_2 & (c_2^2)_2 & (c_1c_2)_2 \\ 1 & (c_1)_3 & (c_2)_3 & (c_1^2)_3 & (c_2^2)_3 & (c_1c_2)_3 \\ 1 & (c_1)_4 & (c_2)_4 & (c_1^2)_4 & (c_2^2)_4 & (c_1c_2)_4 \\ 1 & (c_1)_5 & (c_2)_5 & (c_1^2)_5 & (c_2^2)_5 & (c_1c_2)_5 \\ 1 & (c_1)_6 & (c_2)_6 & (c_1^2)_6 & (c_2^2)_6 & (c_1c_2)_6 \end{bmatrix}, \quad \vec{s} = \begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{bmatrix}, \quad \vec{w} = \begin{bmatrix} w_0(c_1,c_2) \\ w_1(c_1,c_2) \\ w_2(c_1,c_2) \\ w_3(c_1,c_2) \\ w_4(c_1,c_2) \\ w_5(c_1,c_2) \\ w_6(c_1,c_2) \end{bmatrix}^{\mathrm{T}}$$

Extracting the quadratic dependence, a toy example

- Let's plug in the numbers from our seven reweight points and find the $\vec{s}$ that minimizes $||\vec{w} - \mathbf{A}\vec{s}||$ using numpy.linalg.lstsq

```
import numpy as np

w = [ 1.000, 0.909, 1.403, 0.721, 0.333, 0.418, 1.194]
A = [
    [1.0,   0.0,  0.0,   (0.0)**2,  (0.0)**2,   (0.0)*(0.0)  ],
    [1.0,   0.0,  1.0,   (0.0)**2,  (1.0)*2,    (0.0)*(1.0)  ],
    [1.0,   5.0,  0.0,   (5.0)**2,  (0.0)**2,   (5.0)*(0.0)  ],
    [1.0,   5.0, 10.0,   (5.0)**2, (10.0)**2,   (5.0)*(10.0) ],
    [1.0,  -5.0, 10.0,  (-5.0)**2, (10.0)**2,  (-5.0)*(10.0) ],
    [1.0, -10.0, 10.0, (-10.0)**2, (10.0)**2, (-10.0)*(10.0) ],
    [1.0,  10.0, 10.0,  (10.0)**2, (10.0)**2,  (10.0)*(10.0) ],
]

s, resid, _, _ = np.linalg.lstsq(A,w,rcond=None)
```

And the sum of the squared residuals is just 1.15168414e-30, not too big :)

- We find that: `s = [ 1.0, 0.062, -0.0996, 0.00372, 0.0043, -0.00232]`
- This means our quadratic dependence of the weight on WCs is thus:

$$w(c_1, c_2) = 1 + 0.062\,c_1 - 0.0996\,c_2 + 0.00372\,c_1^2 + 0.0043\,c_2^2 - 0.00232\,c_1 c_2$$

Just for fun ⟶

WolframAlpha computational intelligence

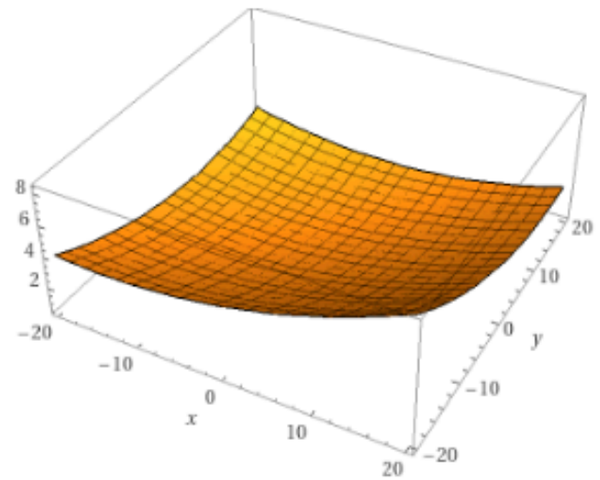
plot (1. + 0.062x -0.0996y + 0.00372x^2 + 0.0043y^2 -0.00232x*y) from x=(-20,20) and y=(-20,20)

NATURAL LANGUAGE | MATH INPUT | EXTENDED KEYBOARD | EXAMPLES | UPLOAD | RANDOM

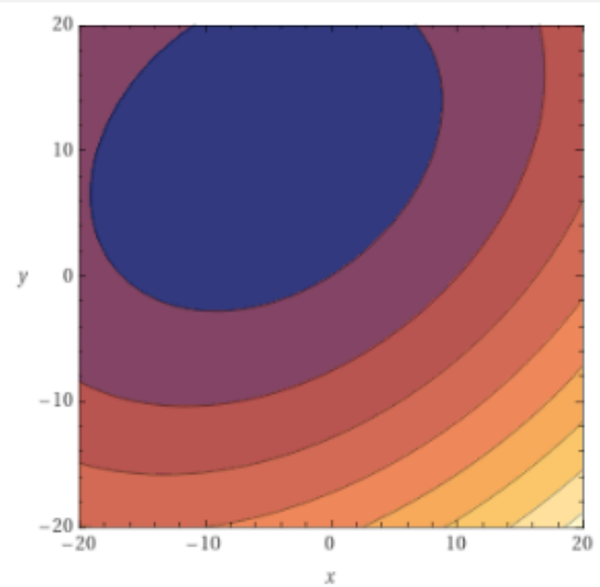
Input interpretation

plot	$1 + 0.062\,x - 0.0996\,y + 0.00372\,x^2 + 0.0043\,y^2 - 0.00232\,x\,y$	$x = -20$ to 20 $y = -20$ to 20

3D plot — Show contour lines



Contour plot



Tutorial outline

- Review of EFT reweighting concepts
- Hands-on example of extracting quadratic from reweight points
- Discussion of EFT histograms concepts
- Hands-on example of using an EFT-aware histogram

Recap of how the concepts fit into the workflow

Gridpack generation ← This is the step where we ask MG to find the weight at the set of reweight points

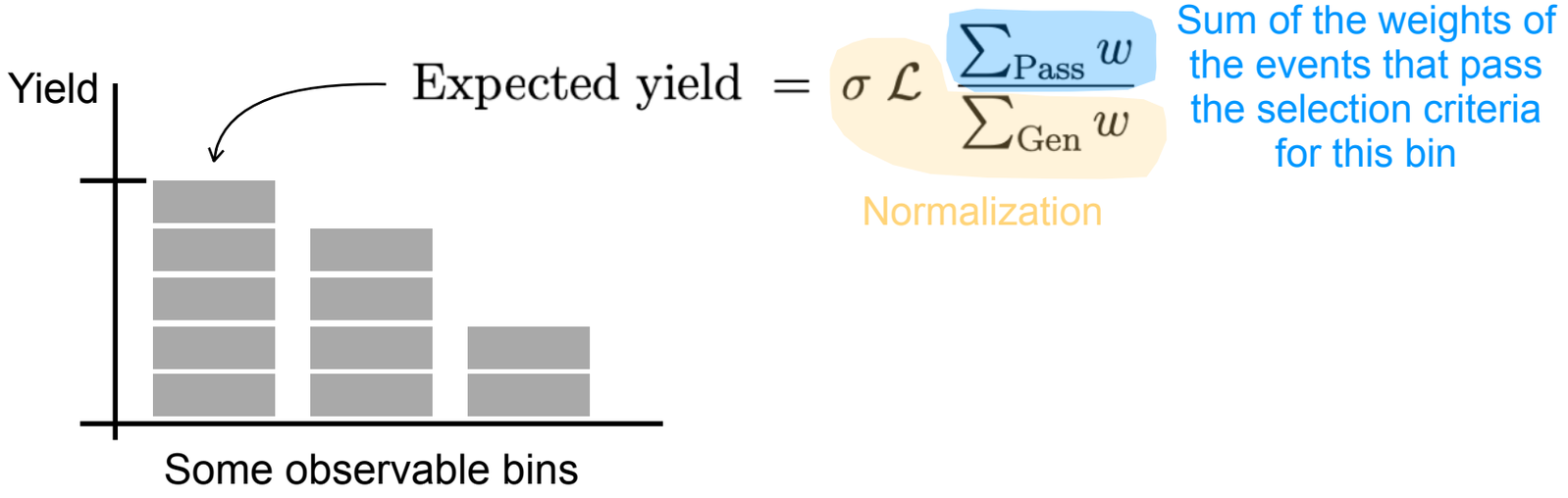
MC event simulation ← Throughout the simulation steps in a general workflow (LHE->GEN->SIM->DIGI->RECO->MAOD->NAOD), we can just carry along the set of weights until the point where we want to extract the quadratic fit
In principle this can be done at any point in between the gridpack generation and histogramming, but in this tutorial we do it in the NanoGen step

Histogramming ← Sum the quadratic parameterizations for all of the events passing the selection criteria for the given bins in order to find the parametrization for the predicted yield in the bin

Likelihood fitting ← Compare the parametrized yields to observed data and extract best fit values and confidence intervals for WCs

Concept: Histograms

- Before we jump into EFT-aware histograms, let's start by recalling some concepts about "regular" histograms
- A regular histogram is essentially a list of bin values and corresponding bin edges
  - The value in each bin is just the sum of the weights of all of the events that pass the selection criteria for the given bin
  - To get the yield, need to normalize properly



"Regular" histogram = [value in bin 1, value in bin 2, value in bin 3]

Concept: EFT-aware Histograms

“Regular” histogram = [Value for bin 1 , Value for bin 2 , Value for bin 3]

An “EFT-aware” histogram stores EFT parameterizations from which you can obtain a predicted yield at any point in WC space

“EFT-aware” histogram = [Quadratic parameterization for bin 1 , Quadratic parameterization for bin 2 , Quadratic parameterization for bin 3]

Concept: EFT-aware Histograms

“Regular” histogram = [Value for bin 1 , Value for bin 2 , Value for bin 3]

An “EFT-aware” histogram stores EFT parameterizations from which you can obtain a predicted yield at any point in WC space

“EFT-aware” histogram = [Quadratic parameterization for bin 1 , Quadratic parameterization for bin 2 , Quadratic parameterization for bin 3]

= []

Instead of storing the sum of weight values, EFT histograms store the *sum of the weight parameterizations* (in terms of the WCs), will in principle be different for each bin (since in principle the dependence is different for each event)

Some technical considerations: Normalization of EFT-aware histograms

- Usually you don't want to use the normalization straight from your generated sample (usually for EFT samples this is LO)
- Want to normalize to the best available theory cross section, as usual
- Usually achieve this normalization by dividing summing the parameterizations for all all generated events, then reweighting to the SM* (i.e. the SM prediction for the total cross section, denoted w(SM))
- After dividing by the w(SM), the constant term in your quadratic parameterization is 1, so after scaling by the lumi and the NLO xsec, the constant piece is the SM predicted yield

$$\text{Expected yield } (\vec{c}) \;=\; \sigma_{SM}\,\mathcal{L}\;\frac{\sum_{\text{Pass}} w(\vec{c})}{\sum_{\text{Gen}} w(SM)},$$

* Note: This normalization approach is not possible in the case when the SM prediction for your sample is 0 (e.g. for FCNC samples, a different normalization approach is required)

Practical considerations:
Tools for EFT-aware histograms

- Now that we've talked about the concepts of EFT-aware histograms, let's discuss what this would look like in practice
- We know we need to store the quadratic parameterization for each bin
- But what really is the quadratic parameterization? Essentially it's just a list of terms, e.g. for two WCs:

$$\text{Quad parameterization} = s_0 + s_1 c_1 + s_2 c_1^2 + s_3 c_2 + s_4 c_1 c_2 + s_5 c_2^2$$

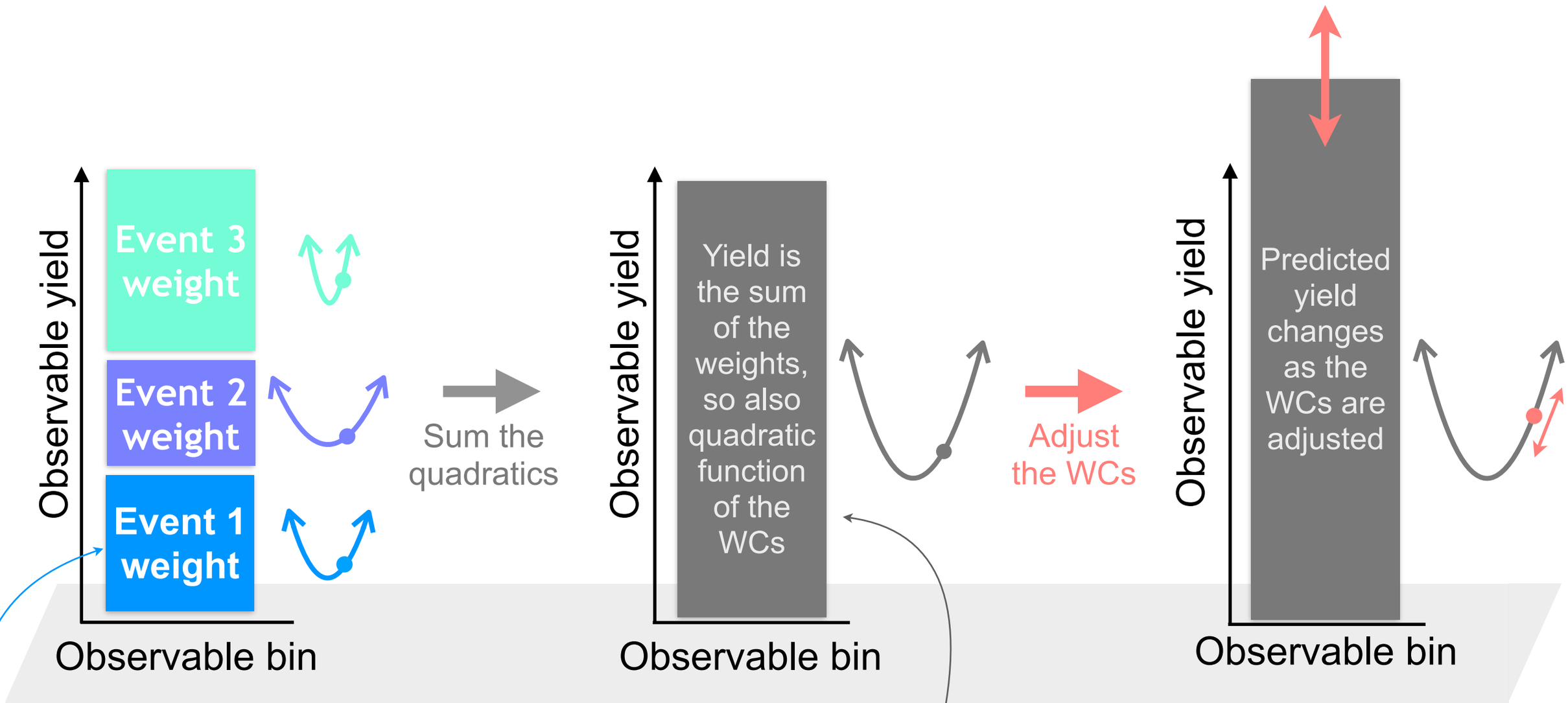
- The terms are essentially a structure constant (called "s" in the above) and the corresponding variables (i.e. the WCs denoted c_i)
- If we follow a convention for the order of the terms, we can just store the list of WCs $[c_1, c_2]$ and the structure constants $[s_0, s_1, s_2, s_3, s_4, s_5,]$ for each bin

See backup for discussion of ordering convention for structure constants

This is implemented in histEFT, which we will explore in the hands-on part coming next

Some history: TOP-19-001 developed EFT-aware "TH1EFT", then TOP-22-006 implemented new version on top of coffea hist and called it histEFT… but since coffee hist is now outdated, histEFT has recently been rewritten (by Ben Tovar of ND CCL) based on the scikit hep hist

Visualization of putting it all together (example with just one bin)



$$w_i = s_{0i} + \sum_j s_{1ij}\, c_j + \sum_j s_{2ij}\, c_j^2 + \sum_{j \neq k} s_{3ijk}\, c_j c_k$$

$$= \cup$$

Obtain the fit coefficients for each event from MG reweighting

$$\text{Yield} = \Sigma w_i$$

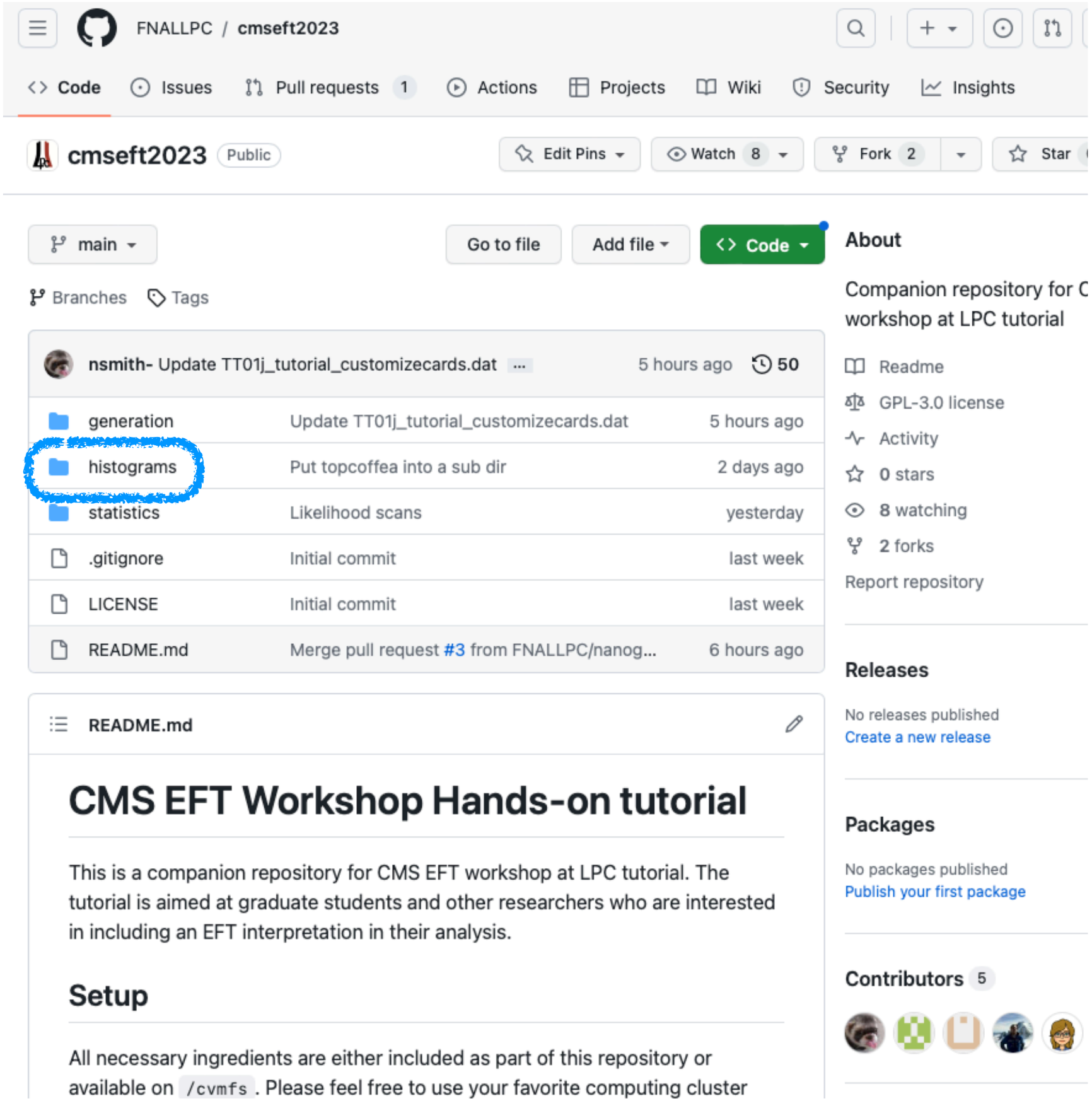
$$= \cup + \cup + \cup = \cup$$

Tutorial outline

- Review of EFT reweighting concepts
- Hands-on example of extracting quadratic from reweight points
- Discussion of EFT histograms concepts
- Hands-on example of using an EFT-aware histogram

Hands-on example

- The histograms tutorial will process the EFT NanoAOD file that you produced in the first tutorial (this NanoAOD contains the quadratic parametrization for each event)
- We will fill some histograms (using the histEFT tool from topcoffea)
- The resulting “EFT-aware” histos will be the input for the final step of the tutorial (the likelihood fits)



Backup

Term ordering convention for histEFT

- For histEFT, the term ordering convention follows the order of the lower triangle of an (n+1)x(n+1) matrix, where n is the number of WCs, and the order of the WCs is assumed to be [sm, c1, c2, …, cn]
- Thus, if you know the WC order, you can reconstruct the quadratic parametrization from the list of terms