



# Unfolding *ATLAS* Collider Data with the Novel OmniFold Method

Adi Suresh, UC Berkeley

DPF 2021, July 12-14

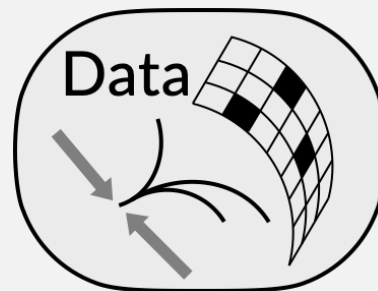
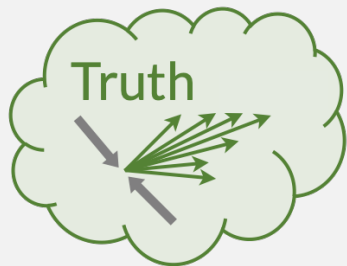
# Particle-level

# Detector-level

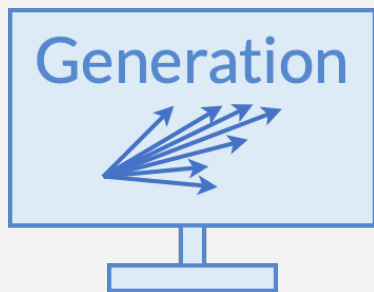
Andreassen,  
Komiske,  
Metodiev,  
Nachman, Suresh,  
Thaler  
arXiv:2105.04448

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Thaler  
PRL 124 (2020)  
182001

Natural



Synthetic



Nomenclature

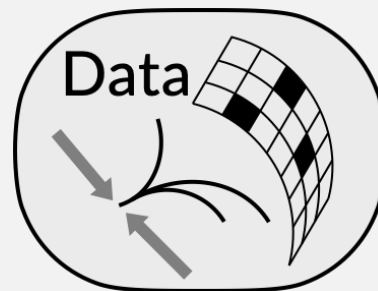
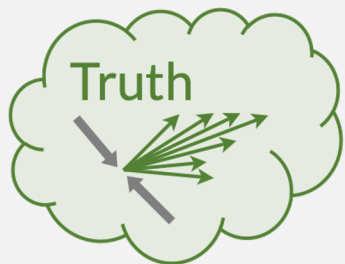
# Unfolding (AKA Deconvolution)

- ▶ Goal: take **data** (at detector-level) and **infer truth** (particle-level)
  - i.e. remove detector effects
- ▶ Key for comparing **theoretical predictions** to what we **observe** in the collider and for comparing between experiments

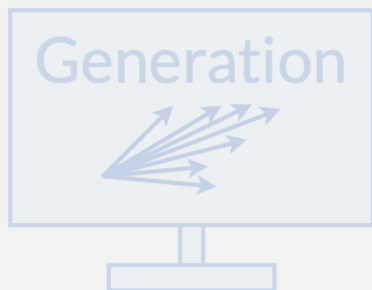
# Particle-level

# Detector-level

Natural



Synthetic



The Goal

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Suresh,  
Thaler  
arXiv:2105.04448

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Thaler  
PRL 124 (2020)  
182001

# Current Unfolding Drawbacks

- ▶ In HEP, the current standard unfolding methods
  - Take only **binned data**
  - Are unfeasible for unfolding **more than 1 dimension**
  - May not capture and remove **all salient detector effects**
- ▶ Output is a 1D histogram

# OmniFold

- OmniFold is a simulation-based, maximum likelihood procedure which uses deep learning to do unfolding
  - Specifically, iterative neural network reweighting (information in the backup slides)

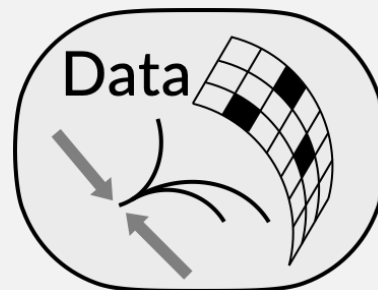
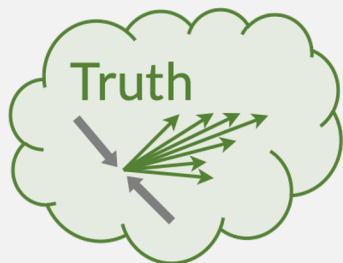
# Particle-level

# Detector-level

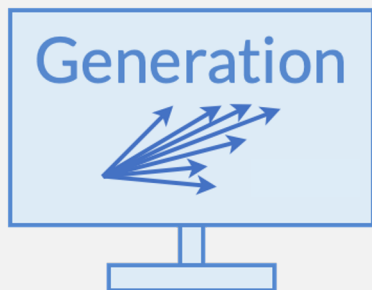
Andreassen,  
Komiske,  
Metodiev,  
Nachman, Suresh,  
Thaler  
arXiv:2105.04448

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Thaler  
PRL 124 (2020)  
182001

Natural



Synthetic



The Starting Point for OmniFold

# OmniFold

- ▶ In contrast to other methods, with NNs, OmniFold...
  - Can take **unbinned data**
  - Easily extendable to **multidimensional** observables or even the **full phase-space**
    - And thus can capture all detector effects
- ▶ The output of OmniFold is a per event **reweighting function** that reweights Generation to Truth

# Flavors of OmniFold

- ▶ UniFold: events represented by **one** observable
  - 1-D input to NN
  - Equivalent to unbinned version of standard methods
- ▶ MultiFold: events are represented by **multiple** observables
  - $n$ -D input to NN
- ▶ OmniFold: **full phase space** events
  - Events are represented as unordered lists of particles and their features
  - Variable dimensional input to NN

# *ATLAS* OmniFold

- ▷ We are working on the **first application of OmniFold** on physical collider data from the ATLAS experiment!
- ▷ Before we go into work in progress results, let's discuss the **methodology of OmniFold**

# Four Functions of a Complete Unfolding Algorithm

1. Background subtraction
2. Resolution effects
3. Fake factors
4. Efficiency factors

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Suresh,  
Thaler  
arXiv:2105.04448

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Thaler  
PRL 124 (2020)  
182001

# 1. Background Subtractions

**Using Neural Positive Reweighting**

(This isn't used in the ATLAS analysis, more information  
in the backup)

# 2. Resolution Effects

Using Iterative Neural Network Reweighting

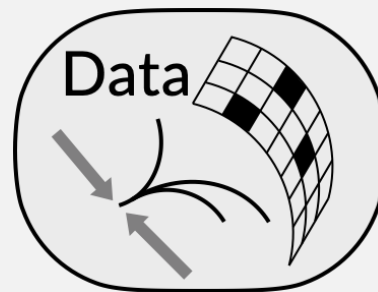
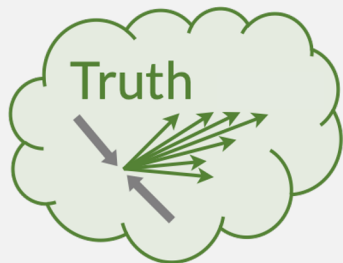
# Particle-level

# Detector-level

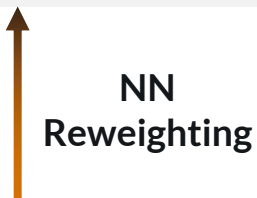
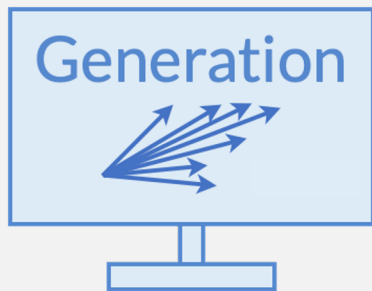
Andreassen,  
Komiske,  
Metodiev,  
Nachman, Suresh,  
Thaler  
arXiv:2105.04448

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Thaler  
PRL 124 (2020)  
182001

Natural



Synthetic

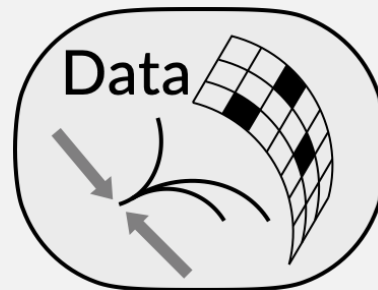
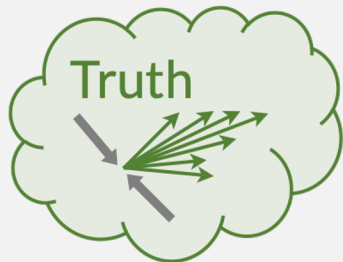


Step 1: Reweight **Sim.** to **Data**

# Particle-level

# Detector-level

Natural



Synthetic



Pull weights back to Gen. to get new Gen. sample: Gen.'

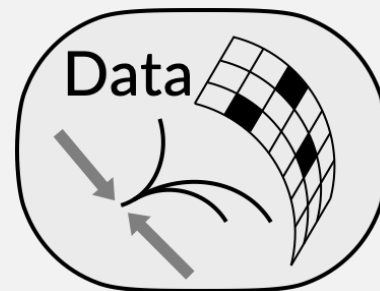
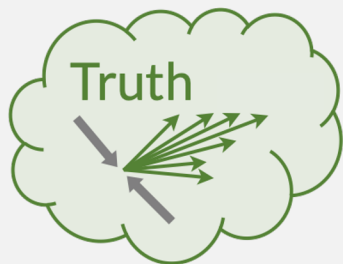
# Particle-level

# Detector-level

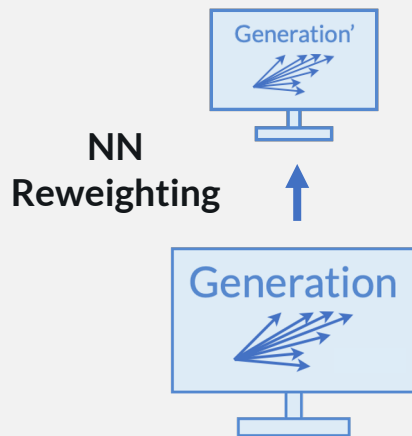
Andreassen,  
Komiske,  
Metodiev,  
Nachman, Suresh,  
Thaler  
arXiv:2105.04448

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Thaler  
PRL 124 (2020)  
182001

Natural



Synthetic

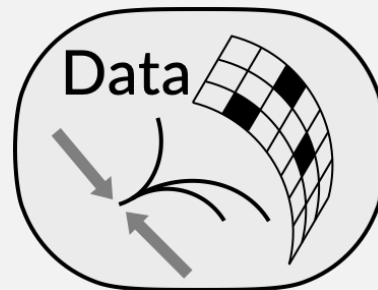
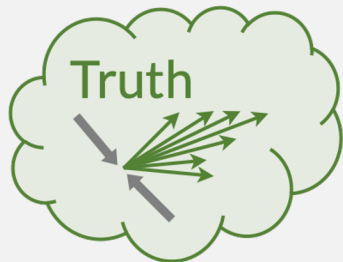


Step 2: Reweight Gen. to Gen.'

# Particle-level

# Detector-level

Natural



Synthetic



Push weights back to Sim. To get new Sim. Sample: Sim.'

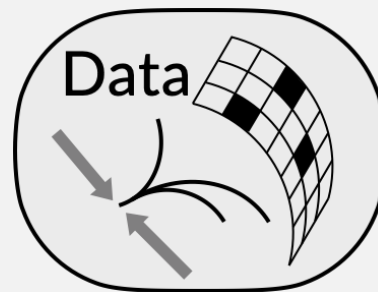
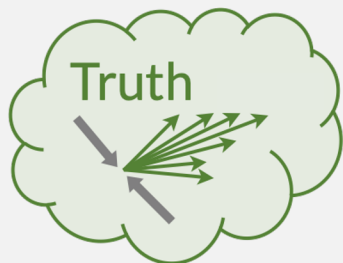
# Particle-level

# Detector-level

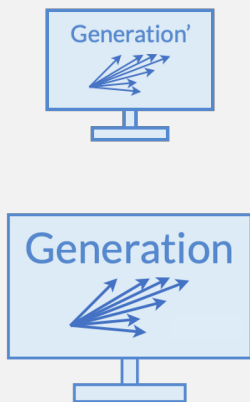
Andreassen,  
Komiske,  
Metodiev,  
Nachman, Suresh,  
Thaler  
arXiv:2105.04448

Andreassen,  
Komiske,  
Metodiev,  
Nachman, Thaler  
PRL 124 (2020)  
182001

Natural



Synthetic



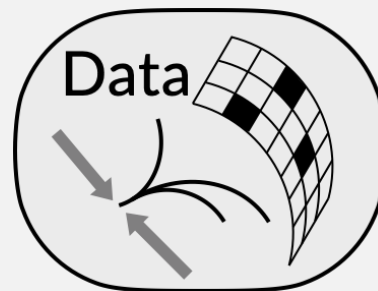
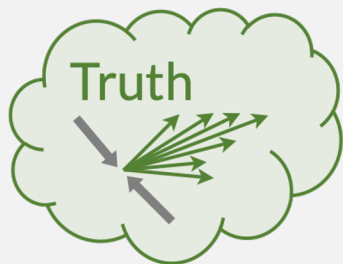
↑  
NN  
reweighting

Iterate... Step 1: Reweight Sim.' to Data

# Particle-level

# Detector-level

Natural



Unfolded!

Synthetic



After  $n$  iterations, take reweighted  $\text{Gen}^n$  as the unfolded distribution

# 3. Fake Factors & 4. Efficiency Factors

# What are fake and efficiency factors?

- ▷ Fake factors are events that pass the detector-level selection but not the particle-level selection
- ▷ Efficiency factors are events that pass the particle-level selection but not the detector-level selection

# How to deal with fake/efficiency factors

- ▶ Simple: assign a **dummy value** to either the particle-level information (for efficiency factors) or the detector-level information (for fake factors) to flag them

# How to deal with fake/efficiency factors

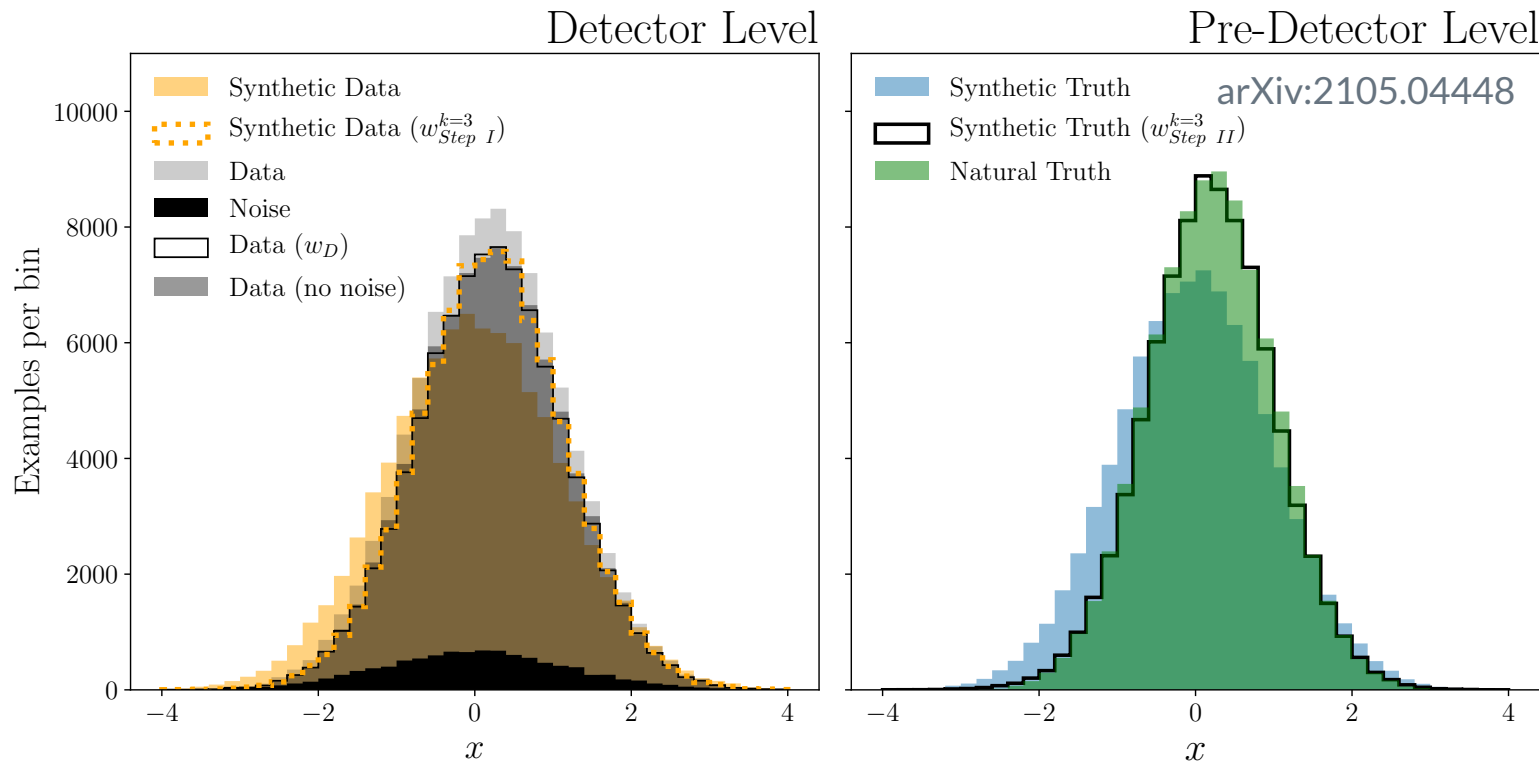
- ▷ Two options when we encounter a dummy value when doing reweighting:
  1. Take the weight of the prior ( $w = 1$ )
  2. Take the average weight
    - $\langle w | x_{\text{particle}} \rangle$  for efficiency factors
    - $\langle w | x_{\text{detector}} \rangle$  for fake factors

# Gaussian Toy Example

# Unfolding a Gaussian Distribution

(from arXiv:2105.04448)

- ▷  $X_T \sim \mathcal{N}(0.2, 0.8)$  ("Truth"/Target)
- ▷  $X_G \sim \mathcal{N}(0, 1)$  (Generation)
- ▷  $X_D \sim X_T + Z$  ("Data")
- ▷  $X_S \sim X_G + Z$  (Simulation)
- ▷  $Z \sim \mathcal{N}(0, 0.5)$  (Detector distortion)
- ▷ 10% of  $X_D$  are background noise processes  $\mathcal{N}(0, 1.1)$
- ▷ 10% of  $X_D$  and  $X_T$  have fake or efficiency factors



OmniFold on a Gaussian distribution after 3 iteration.

# Work In Progress

*ATLAS* OmniFold

## Currently...

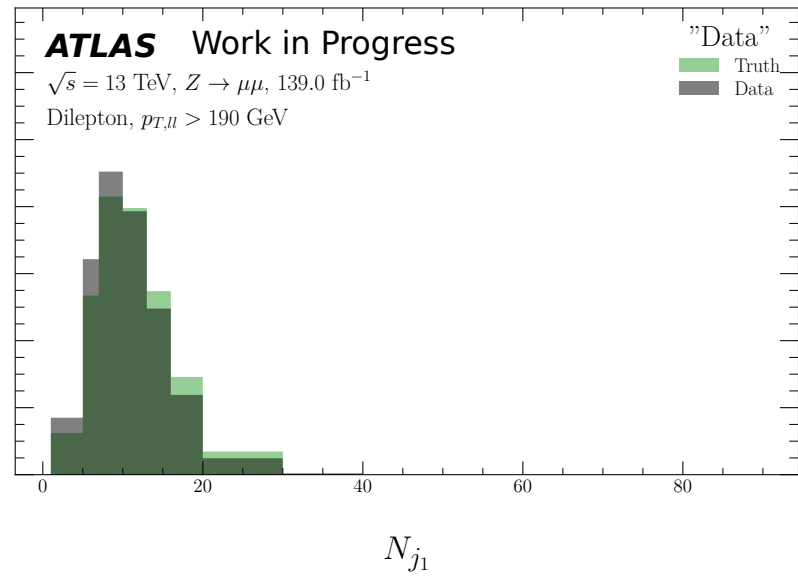
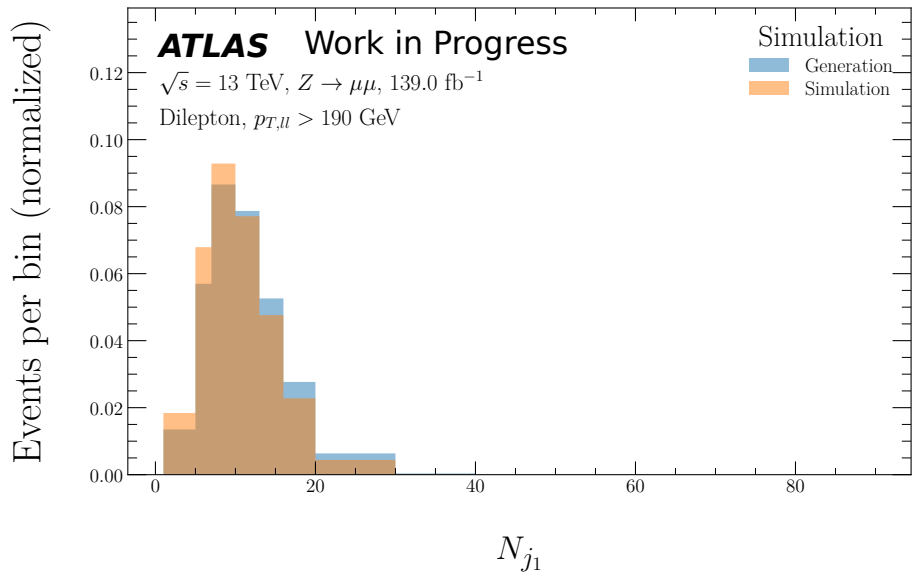
- ▶ Objective: use OmniFold on **physical collider data** from *ATLAS*
  - $Z \rightarrow \mu^+ \mu^- + \text{jets}$
- ▶ First: test on simulation
  - to ensure **validity** of implementation
  - **optimize** procedure in advance for data (e.g. ML hyperparameters, number of iterations)
  - estimate **uncertainty**

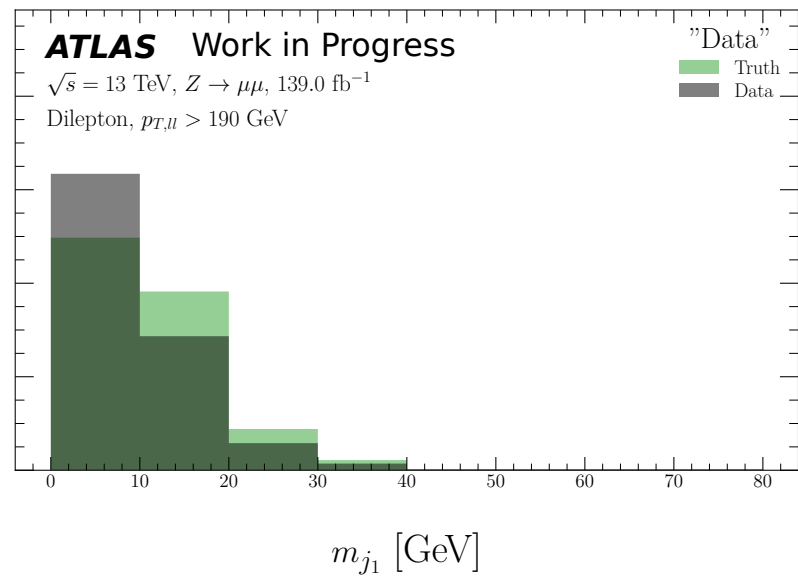
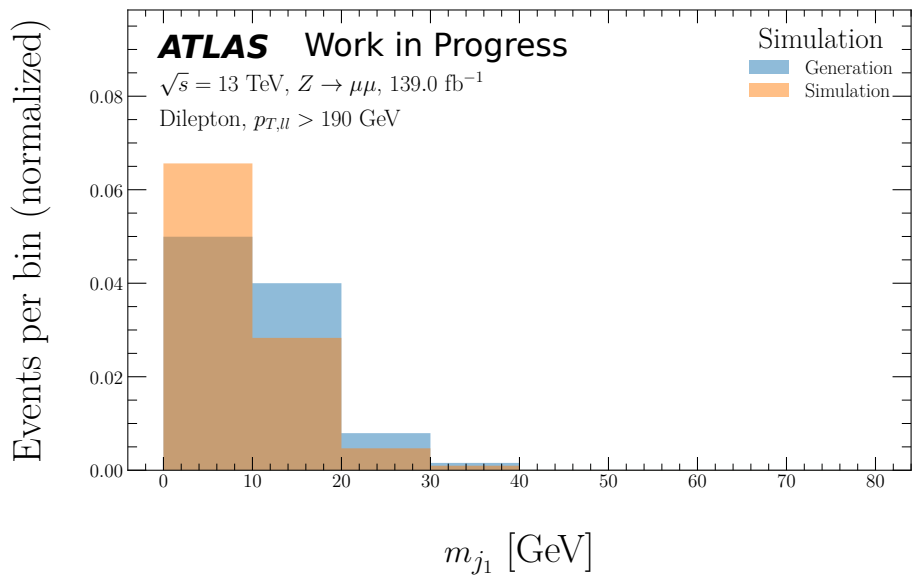
# Work in Progress

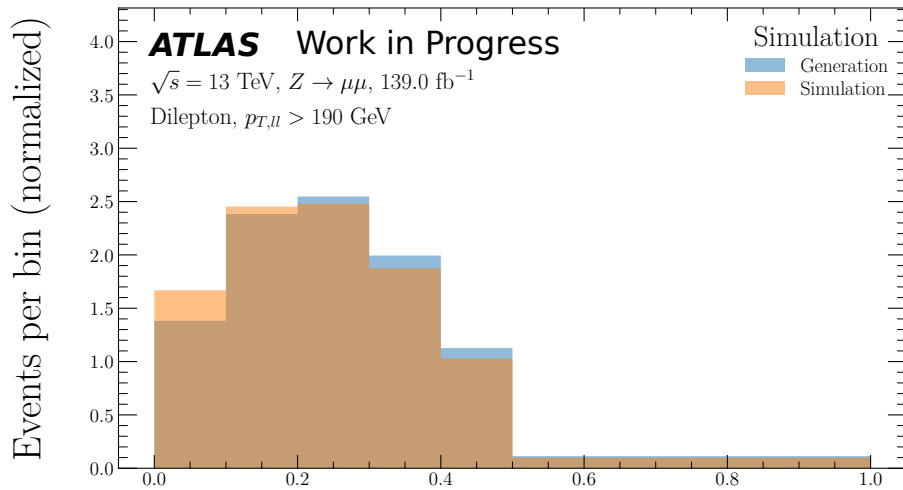
- ▷ Have a working implemented framework for UniFold and MultiFold
- ▷ On one sample, we have done:
  - **Technical closure test:** unfolding sample against itself
  - **Stress test:** induced artificial weights to get a 'new' sample and unfolded against original

# Work in Progress

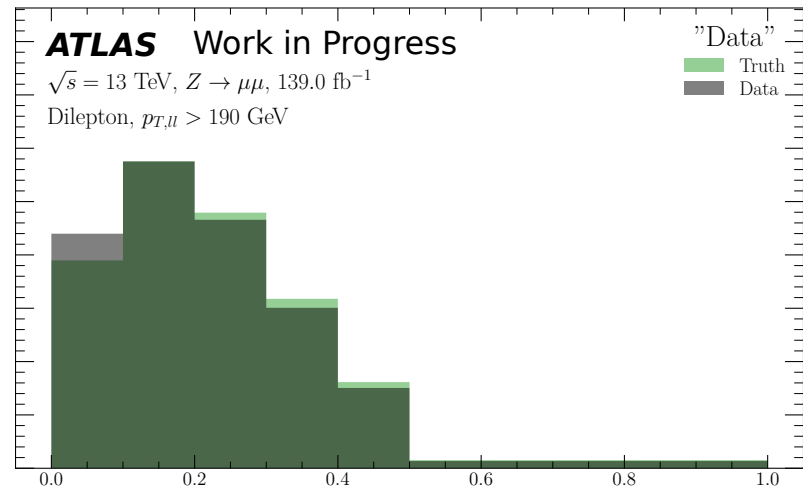
- ▷ Used two different samples (each with pre-detector generation and post-detector simulation events) and labeled one as “**data**” (pseudo-data)
- ▷ Used the other sample as simulation/generation to unfold “data”
- ▷ We have MultiFolded **24 observables simultaneously!**



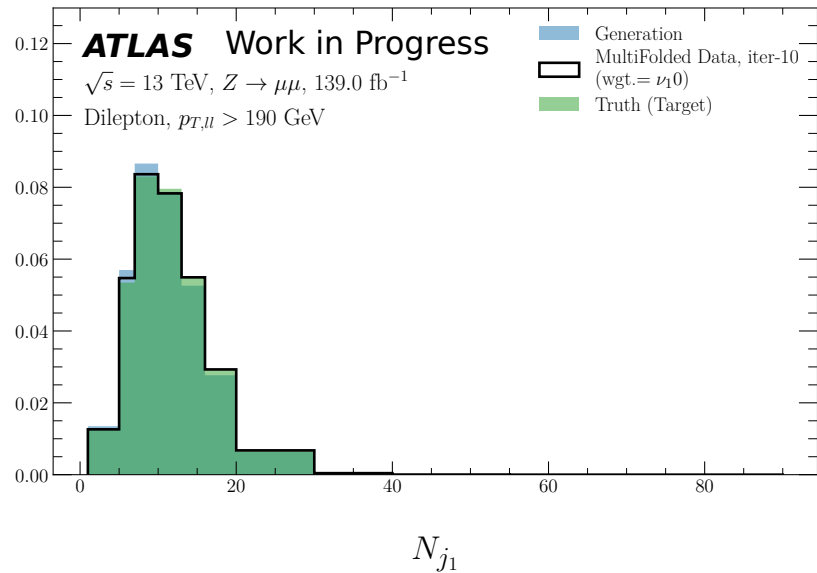
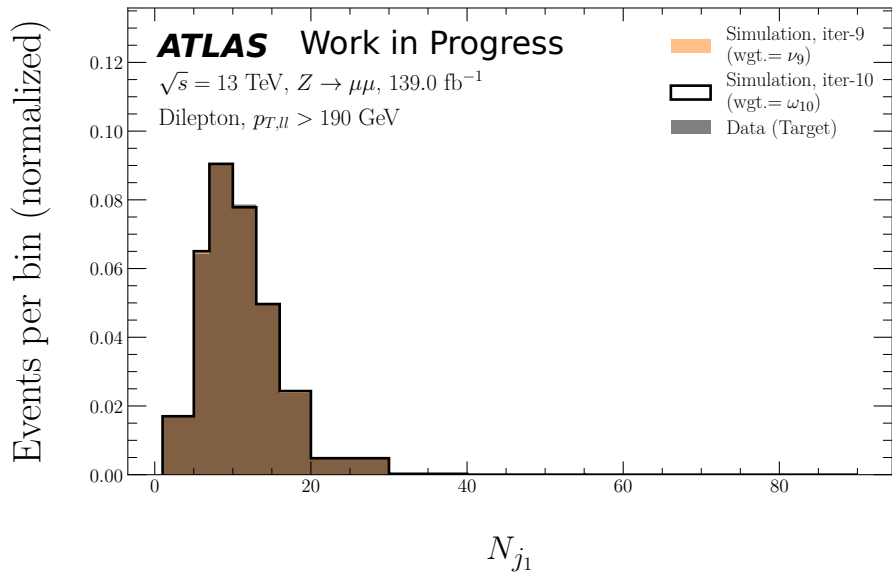


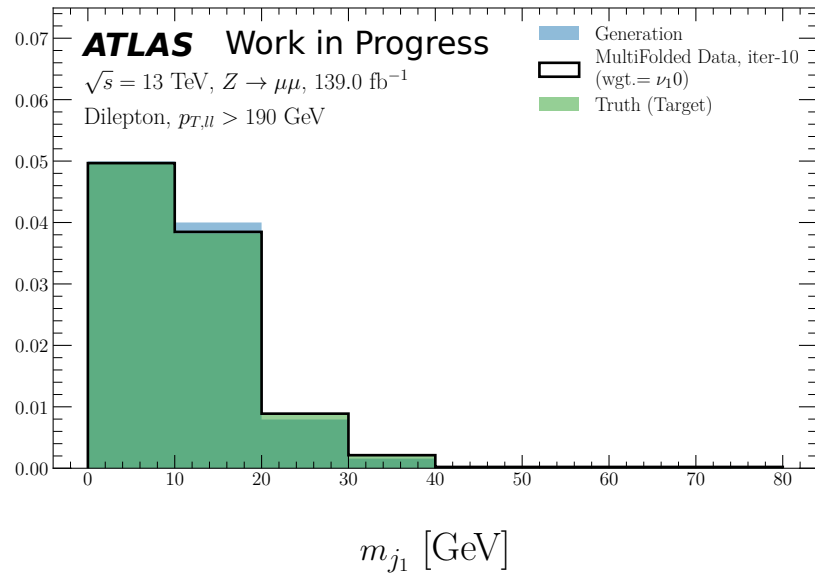
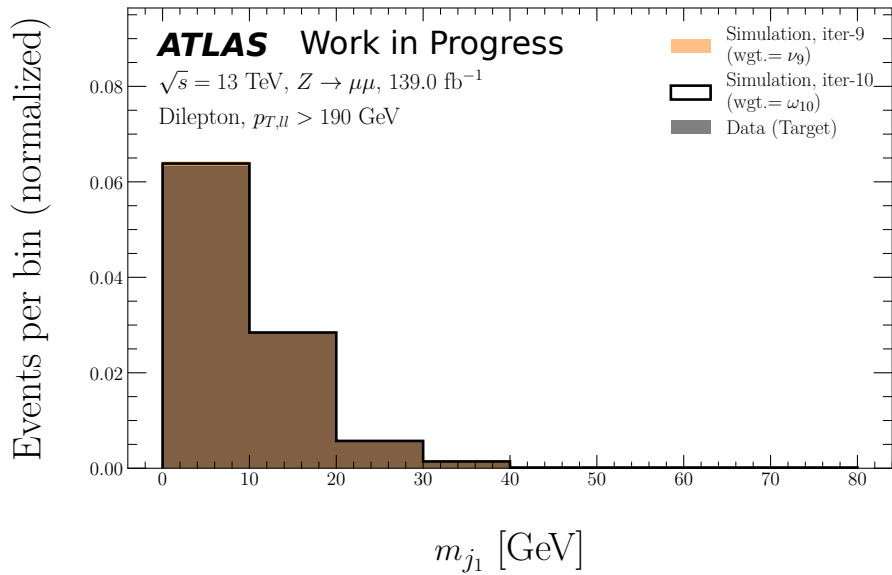


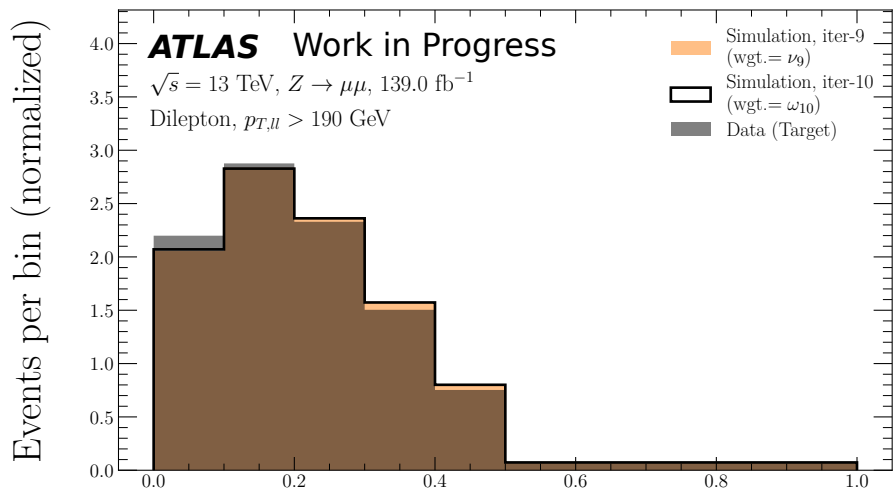
$\tau_{1,j_2}$  (Subleading Jet  $n$ -subjettiness)



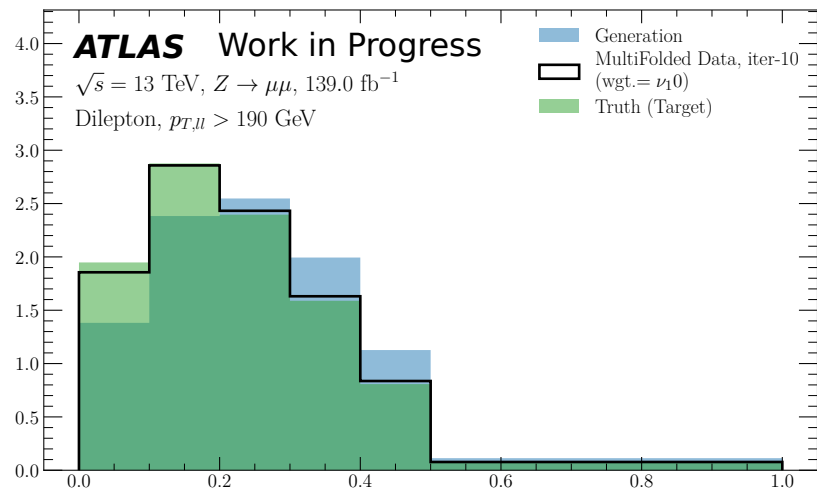
$\tau_{1,j_2}$  (Subleading Jet  $n$ -subjettiness)







$\tau_{1,j_2}$  (Subleading Jet  $n$ -subjettiness)



$\tau_{1,j_2}$  (Subleading Jet  $n$ -subjettiness)

## Next steps

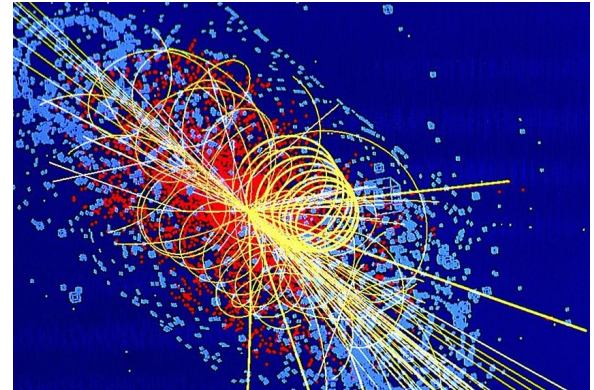
- ▷ We are working on statistical and systematic uncertainty estimates
- ▷ Also working on a comparison with IBU (Iterative Bayesian Unfolding, a commonly used method)
- ▷ Other challenge: constructing and training a flexible NN for OmniFold, which is more difficult

THANK YOU!

**Any questions?**

You can contact me at:

» [adisurtya@berkeley.edu](mailto:adisurtya@berkeley.edu)



# Backup:

# 1. Background Subtractions

## Using Neural Positive Reweighting

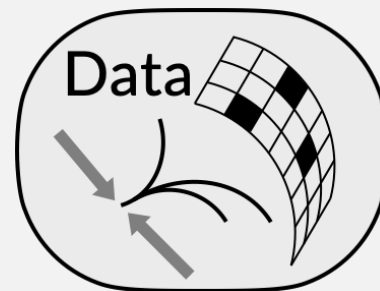
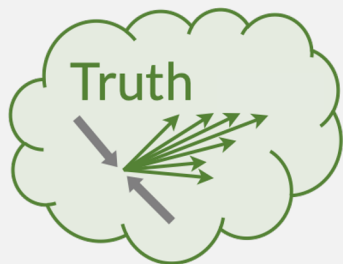
# Neural Positive Reweighting

- ▶ To remove background noise processes from the data, we employ neural positive reweighting **before** dealing with any detector resolution effects.
- ▶ This requires producing another detector simulation of **only the background noise processes**

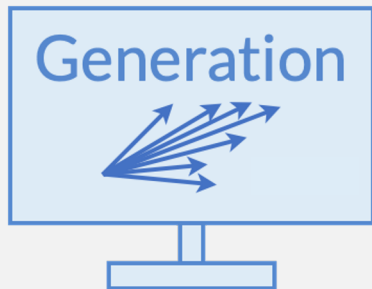
## Particle-level

## Detector-level

Natural



Synthetic



Additional detector simulation of noise only

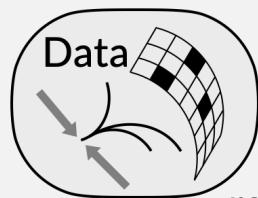
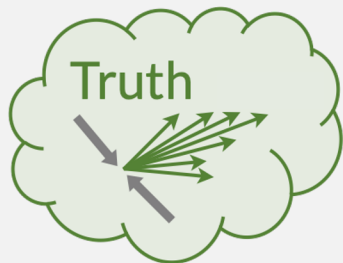
# Neural Positive Reweighting

- ▷ We assign **event weights of  $-1$**  to the background simulation, and then **concatenate** this sample with the data sample
  - This effectively **removes** the background noise from the data
- ▷ However, to avoid dealing with negative weights, we then reweight the nominal data sample to the new concatenated sample with **only positive event weights**

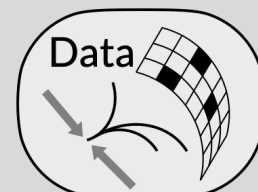
# Particle-level

# Detector-level

Natural



→  
NN  
reweighting



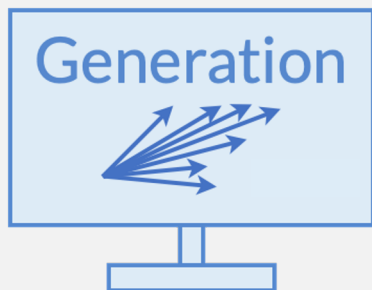
+

Noise Sim.



$w=-1$

Synthetic

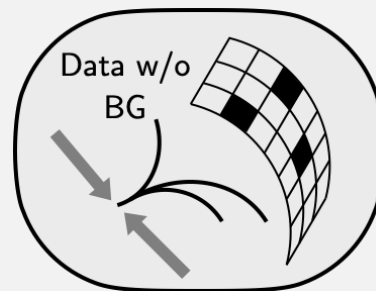
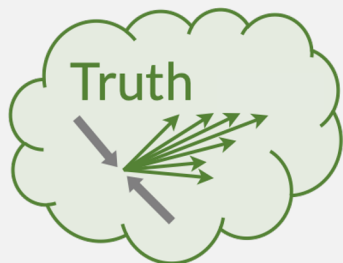


Neural positive reweighting to remove noise from data

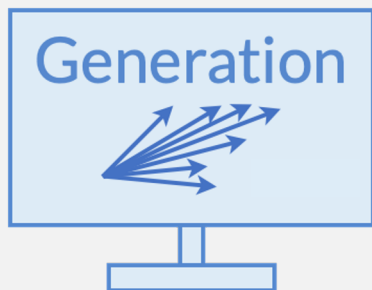
## Particle-level

## Detector-level

Natural



Synthetic



The New Starting Point for OmniFold

# Backup:

# How NN Reweighting Works

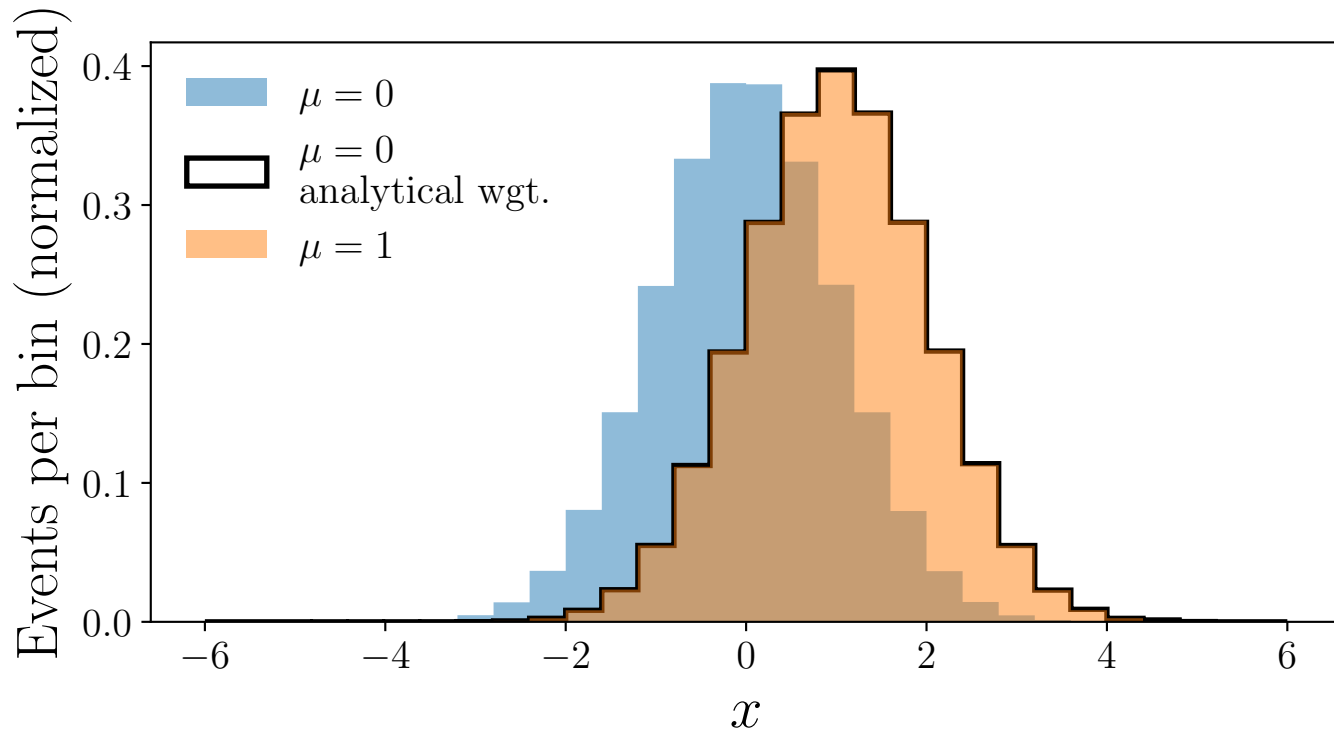
**DCTR – Deep Neural Networks using Classification for  
Tuning and Reweighting**

# Reweighting from One Sample to Another

- ▷ Suppose Sample A is drawn from  $p_0(x)$
- ▷ Suppose Sample B is drawn from  $p_1(x)$
- ▷ Reweighting function
  - Takes an event  $x$  from Sample A and reweights it to an event from Sample B
  - $w_{A \rightarrow B}(x) = \frac{p_1(x)}{p_0(x)}$

## Example: Gaussian Distribution

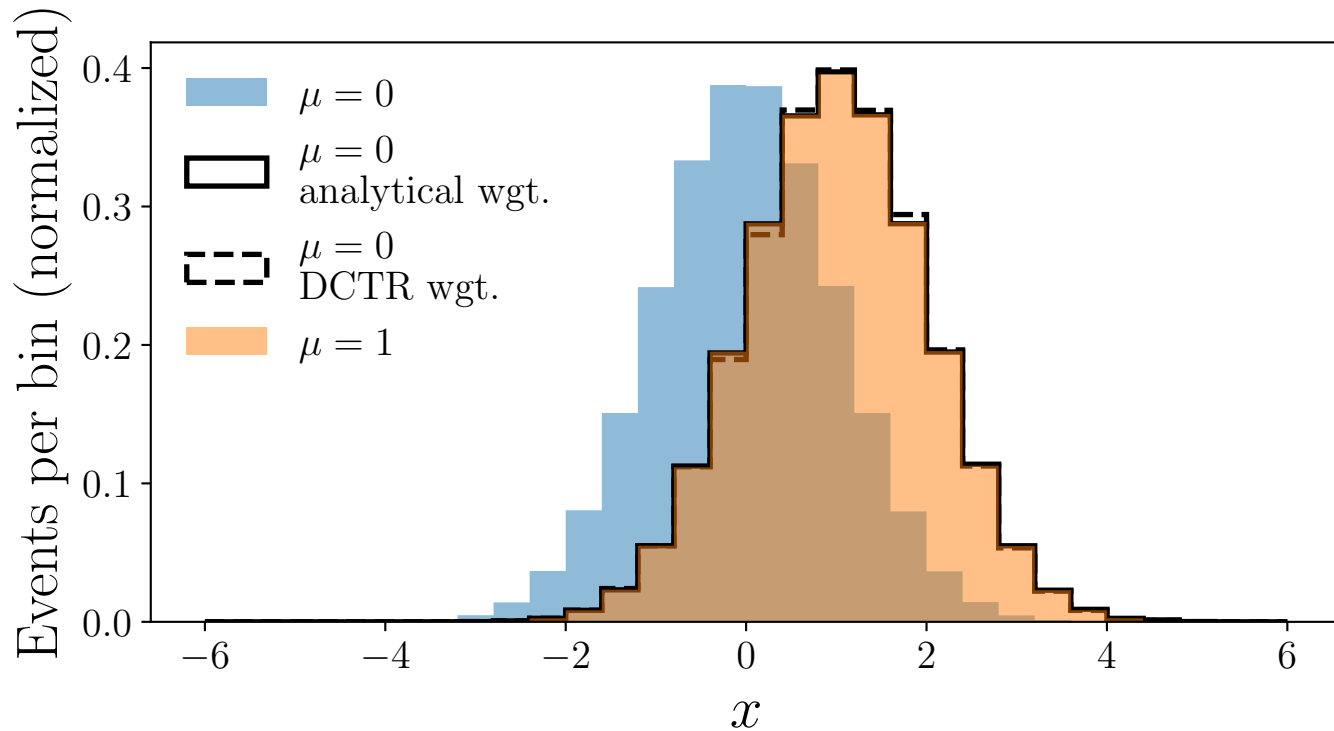
- ▷ Suppose Sample A is drawn from  $\mathcal{N}(0, 1)$ 
  - $p_0(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$
- ▷ Suppose Sample B is drawn from  $\mathcal{N}(1, 1)$ 
  - $p_1(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-1)^2}$
- ▷ Then the reweighting function is:
  - $w_{A \rightarrow B}(x) = e^{-\frac{1}{2}((x-1)^2 - x^2)}$



Analytical Reweighting from a Gaussian centered at 0 to a Gaussian centered at 1

# Neural Networks and Reweighting

- ▷ Reweighting is useful because neural networks can easily approximate the likelihood ratio:
  - Let  $f(x) \in [0,1]$  be a binary classifier trained to distinguish Sample A from Sample B
  - $w_{A \rightarrow B}(x) = \frac{p_1(x)}{p_0(x)} \approx \frac{f(x)}{1-f(x)}$ 
    - **(DCTR Reweighting)**



Different reweightings from a Gaussian centered at 0 to a Gaussian centered at 1

# Parameterized Reweighting

- ▷ Suppose some key generator parameters  $\theta$  inform  $p(x)$ 
  - E.g. the center  $\mu$  and the width  $\sigma$  of a Gaussian
- ▷ Then we can add  $\theta$  as an input to  $f$  and get a parametrized neural network  $f(x, \theta)$

# Simultaneous and Full Phase-space Reweighting

- ▷ An event  $x$  may be multidimensional and/or of variable dimension
  - Could contain multiple observables
  - Could contain the full phase-space of information
- ▷ HEP full phase-space

- $x = \begin{pmatrix} m_1 & p_{T,1} & \eta_1 & \phi_1 \\ m_2 & p_{T,1} & \eta_2 & \phi_2 \\ m_3 & p_{T,3} & \eta_3 & \phi_3 \end{pmatrix}$

# EnergyFlow: Particle Flow Networks

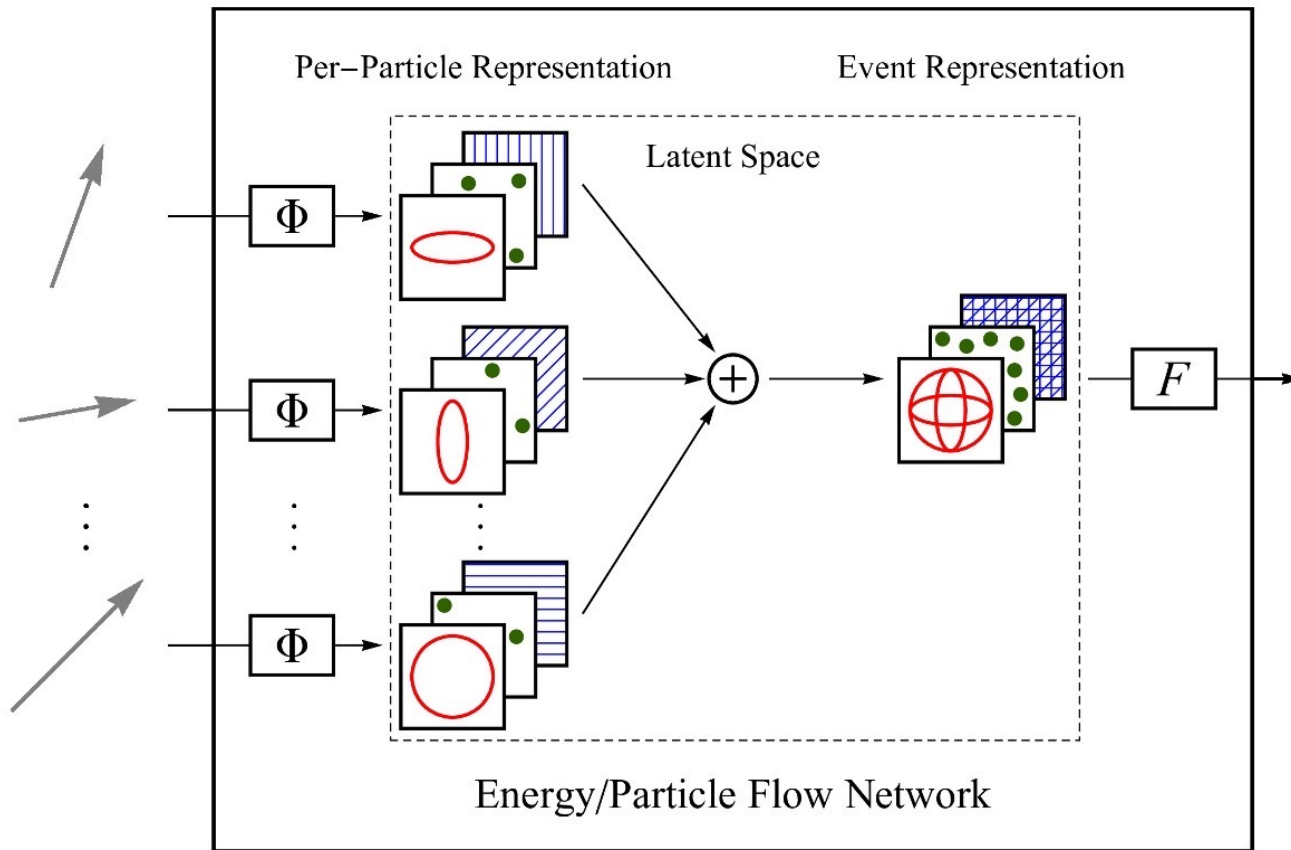
- ▶ Collider data is of **variable dimension**; there is no fixed number of particles per event
  - So how do we structure our neural networks if the input is a different size every time?
- ▶ EnergyFlow's **Particle Flow Networks**
  - Python package with a TensorFlow/Keras backend

# Particle Flow Networks and Deep Sets

- ▷ PFN =  $F(\sum_{i=1}^M \Phi(p_i))$  (from **Deep Sets**)
  - $p_i \in \mathbb{R}^d$  – relevant per particle info (e.g. four-vector)
  - $\Phi: \mathbb{R}^d \rightarrow \mathbb{R}^l$  – per particle mapping
    - $\mathbb{R}^l$  is a **latent** space
  - $F: \mathbb{R}^l \rightarrow \mathbb{R}$  – continuous function
    - $F: \mathbb{R}^l \rightarrow [0, 1]$  (binary classifier output)
- ▷ Because of the **sum**, also respects the **permutation invariance** of collider data!

Particles

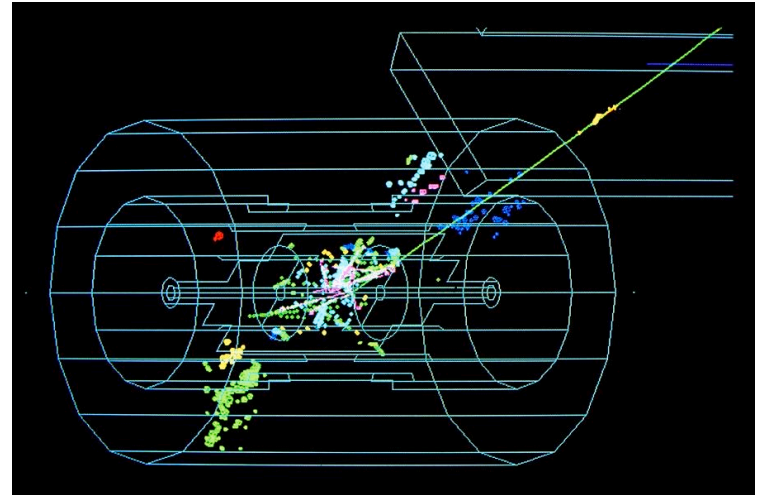
Observable

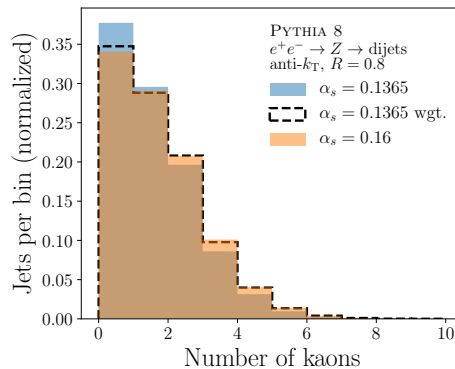
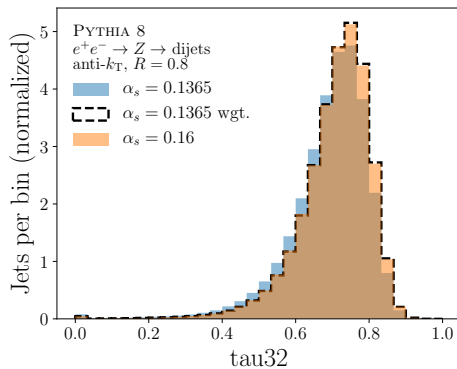
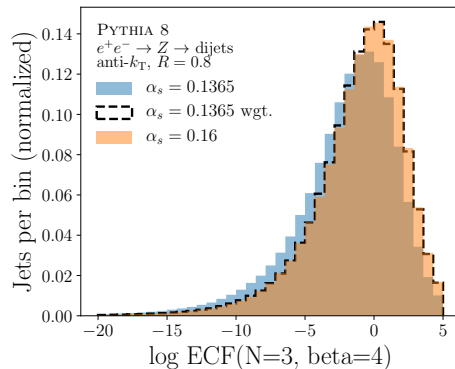
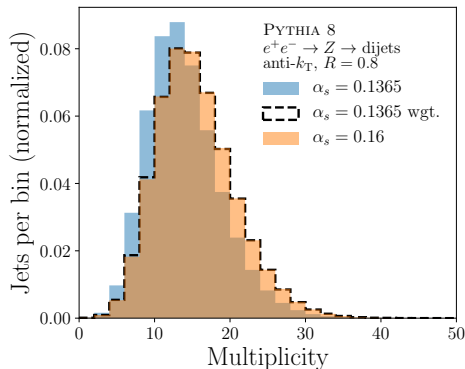


Particle Flow Network Schematic Diagram

# HEP Example: electron-positron collisions

- ▷  $e^+e^- \rightarrow Z \rightarrow$  dijets
- ▷ It is difficult to visualize such high-dimensional information, so we **construct several observables to visualize and probe the phase space!**





Full phase-space reweighting from  $\alpha_s = 0.1365 \rightarrow 0.1600$  for particle level  
 $e^+e^- \rightarrow Z \rightarrow$  dijets