OmniFold: Machine Learning-Assisted Unfolding

Andrew Cudd, Masaki Kawaue, Tatsuya Kikawa, Roger Huang, Ben Nachman, Callum Wilkinson

NuSTEC Cross Experiment Working Group (CEWG) Seminar

2024/12/05

University of Colorado Boulder BERKELEY LAB KYOTO UNIVERSITY

Measurements and Unfolding

Measure **selected number** of **events** in a **reconstructed variable** -- what the detector saw.

Efficiency | Background | Unfolding

Want the **total number** of **signal events** in a **true variable** -- what physically happened.

Assuming no background: $$R_j = \sum_{i}^{N} S_{ij} \, T_i \longleftrightarrow T_j = \sum_{j}^{N} U_{ij} \, R_i$$

Unfolding is finding the unsmearing matrix **U** given the smearing matrix **S** and removing the detector effects from the measured data (**R** in *j* bins) to get the "true" distribution (**T** in *i* bins)

Simplest method would be to invert **S**, but this is usually regarded as a bad move

Unfolding is hard

The process of unfolding is to estimate (or infer) the true distribution using smeared (reconstructed) observations

Unfolding is an ill-posed problem with many methodological challenges

The main challenge is that the smearing (response) matrix **S** is an ill-conditioned matrix

As a result, **very different true histograms can map to very similar smeared (reconstructed) distributions** – distinguishing between true predictions based on noisy data is quite difficult

Slide entirely adapted from M. Kuusela

“Traditional” Unfolding

Several common unfolding techniques currently used for neutrino physics are:

- iterative Bayesian unfolding (aka D’Agostini)
- SVD unfolding (including Wiener SVD)
- template likelihood unfolding (e.g. recent T2K analyses)

These methods (generally) **require that the distributions are binned**, and work best with a **small set of variables** (around 1 to 4)

However many of the **corrections (e.g. efficiency) can have high-dimensional dependence**, and this is difficult to capture with only a few variables

In all cases the reconstructed MC distribution is reweighted to better match the data, and this is propagated to the truth MC distribution

Machine learning (ML) assistance

Neural networks **learn to approximate the likelihood ratio** when trained to distinguish between two datasets

(or something monotonically related to it in some known way)

This transforms the problem from **density estimation** (which is hard) to **classification** (which is ~~easy~~ less hard)

Neural nets are naturally unbinned and are well suited to high-dimensional datasets

Note: that other classifiers could be used for this, such as a boosted decision tree

Explained in detail in this paper: A. Andreassen, B. Nachman, PRD RC 101 (2020) 091901, arxiv:1907.08209

OmniFold concept: ML reweighting

Train a fully connected neural network to classify between two datasets A & B:

Using the **weighted cross-entropy loss** where each event **x** has a weight **w** and a true label **p** gets a prediction **q**

$$L(p_i, q_i) = -w_i \left(p_i \log(q_i) + (1 - p_i) \log(1 - q_i)\right)$$

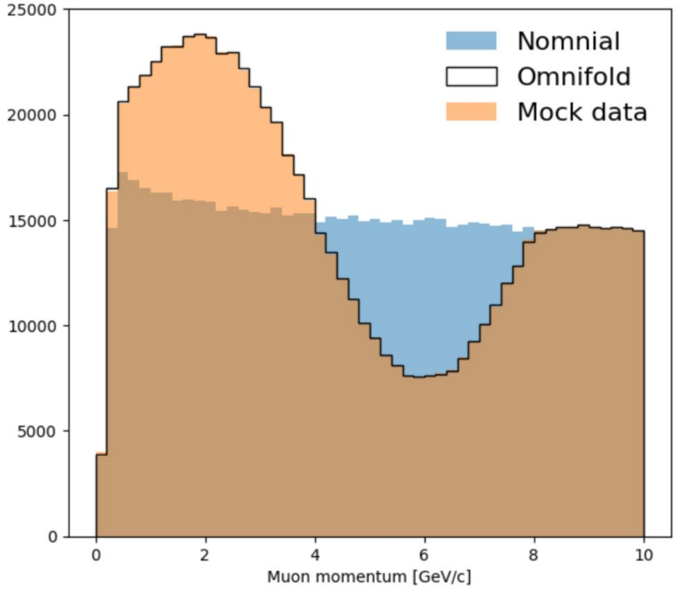
The predictions from the network **q** then approximate the ratio of the two datasets and can be used to **reweight from one to the other**:

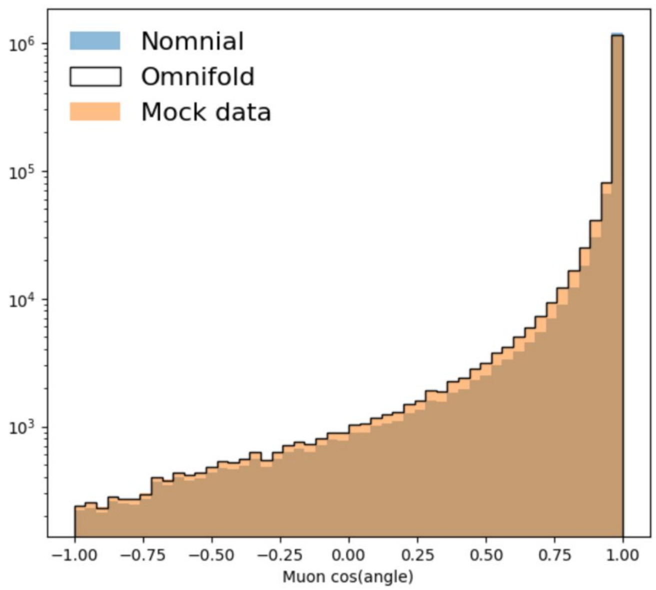
$$\mathcal{L} = p_A(x_i)/p_B(x_i) \approx q_i/(1 - q_i)$$

In practice this uses real (or fake) data and the MC prediction as inputs to the network ➔ this talk uses a public T2K MC dataset as input to OmniFold

OmniFold paper: A. Andreassen, P. Komiske, E. Metodiev, B. Nachman, J. Thaler, PRL 124 (2020) 182001 | Ben's NuXtract Talk

Reweighting example





Using NEUT events simulated with a flat flux and weighted with some arbitrary function of muon momentum |p| (ask if you want more details about this)

T2K CC0pi event selection

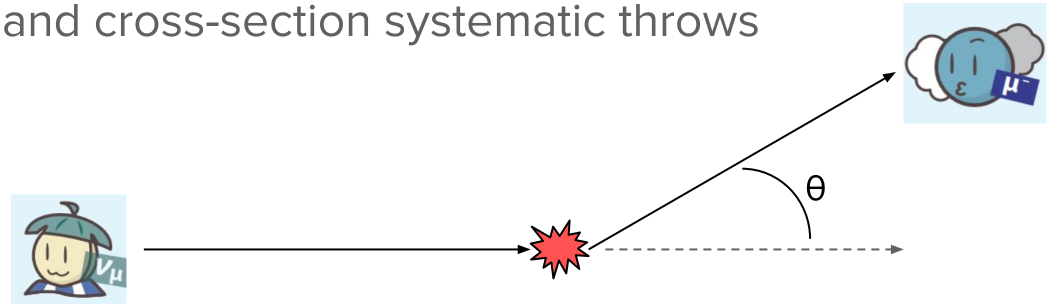
CC0pi signal definition (neutrino mode): **one negatively charged muon**, **zero pions**, and **any number of hadrons** detected in the final state where the vertex was reconstructed in the FGD1 (scintillator) fiducial volume

Signal samples are categorized by the (sub-)detectors used in the event, and the analysis includes several control samples to constrain background events

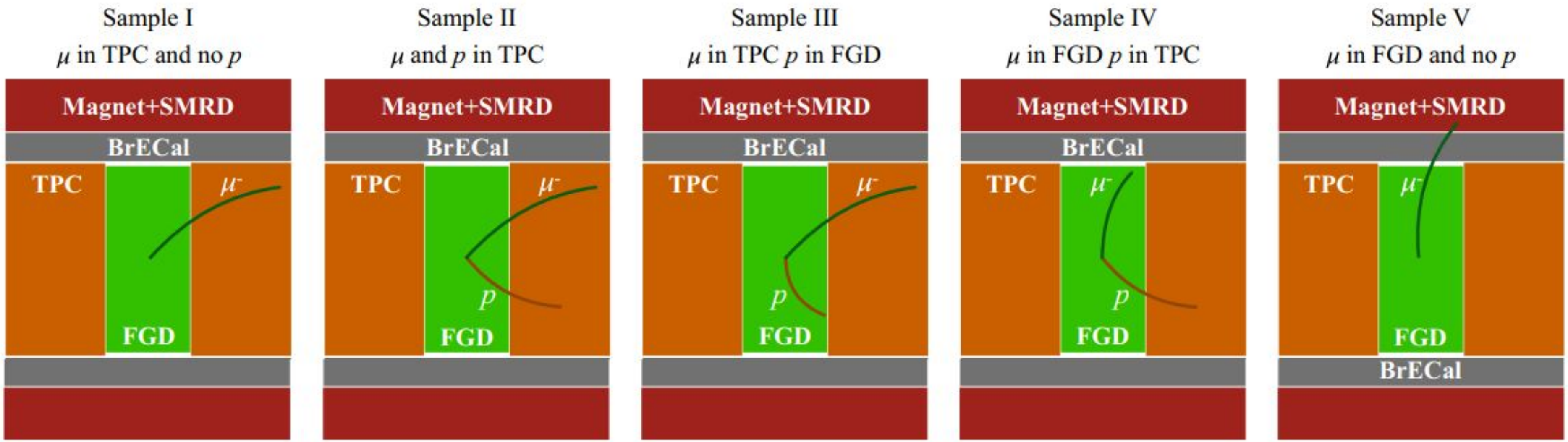
Events in the data set are characterized by muon (proton) kinematics, and also include weights for detector, flux, and cross-section systematic throws

Corresponding paper:
https://doi.org/10.1103/PhysRevD.101.112001

(see also:
https://doi.org/10.1103/PhysRevD.108.112009 or
https://doi.org/10.1103/PhysRevD.101.112004)



T2K ND280 Detector Samples



Reconstructed events split by which subdetectors the leading muon and proton are found in, due to different detector response and efficiencies

Also include control samples (not shown) for background constraints

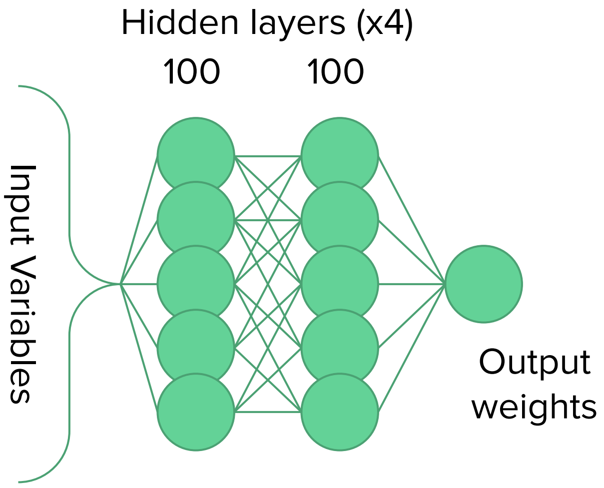
OmniFold Inputs & Network

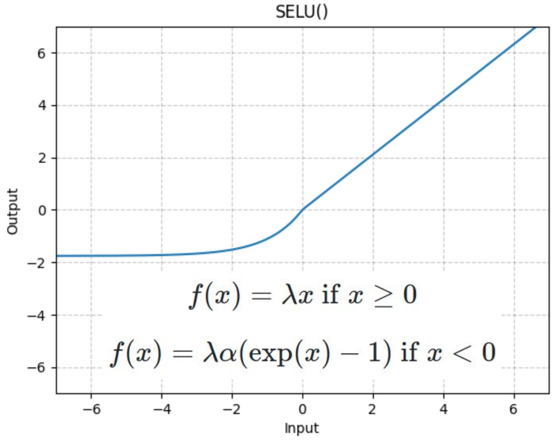
Network architecture is **four densely connected hidden layers** with **100 nodes each** (SeLU activation)

Input variables: **muon kinematics** and **leading proton kinematics** (if present) for each event parameterized as (log(|p|), cos(θ), φ), and **detector sample** (one-hot encoded)

Kinematic variables are normalized to have zero mean and unit variance

Not all background events are included due to missing information in the data release (e.g. NC events)





OmniFold procedure

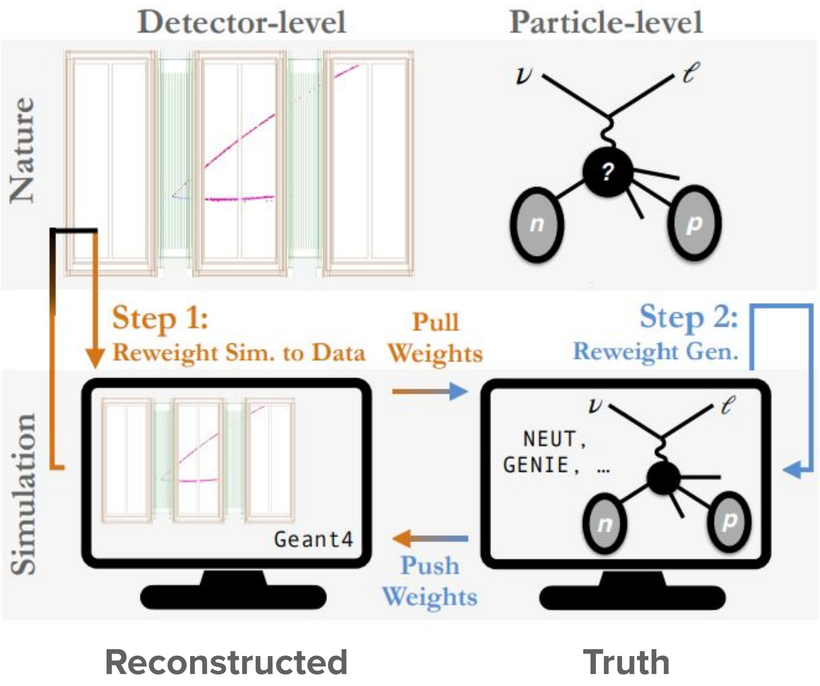
Running 15 iterations of OmniFold takes less than 30 minutes using a single NVidia A100 GPU on a NERSC Perlmutter node

OmniFold is an **iterative unfolding procedure** performed in two steps:

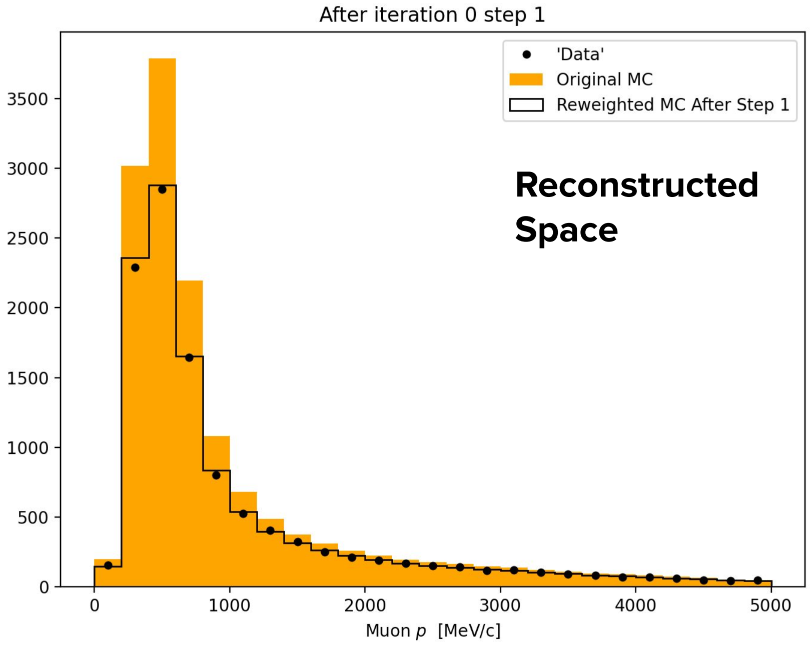
1. Reweight reconstructed MC distribution to (better) match data
2. Reweight nominal truth MC distribution to incorporate information from step 1

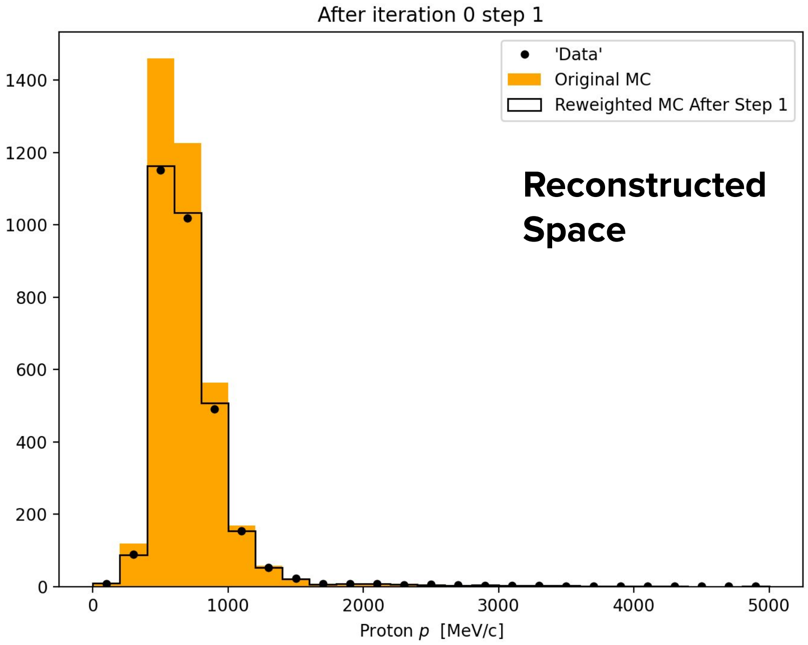
This is one iteration, and the method repeats until some convergence criteria is satisfied (or iteration limit is reached)

Each step uses its own network

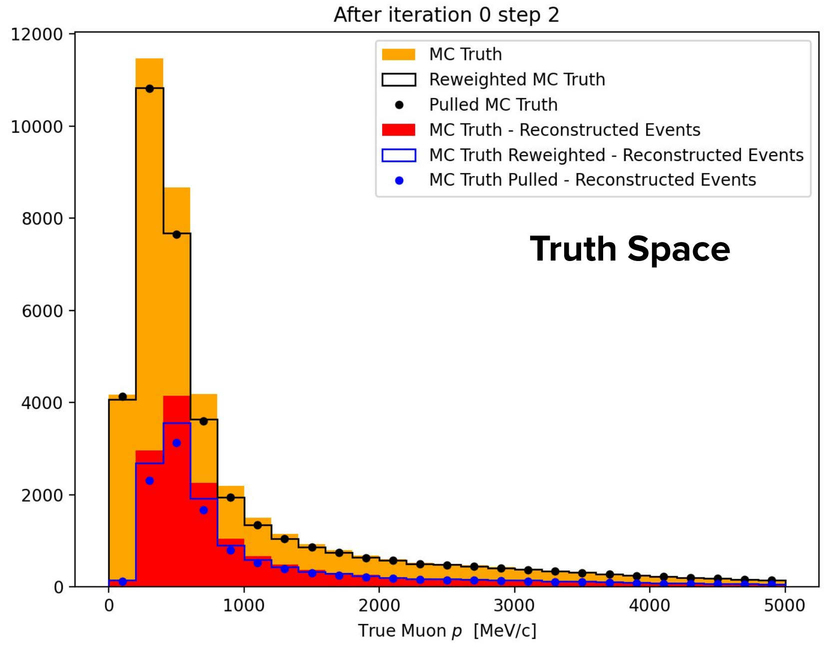


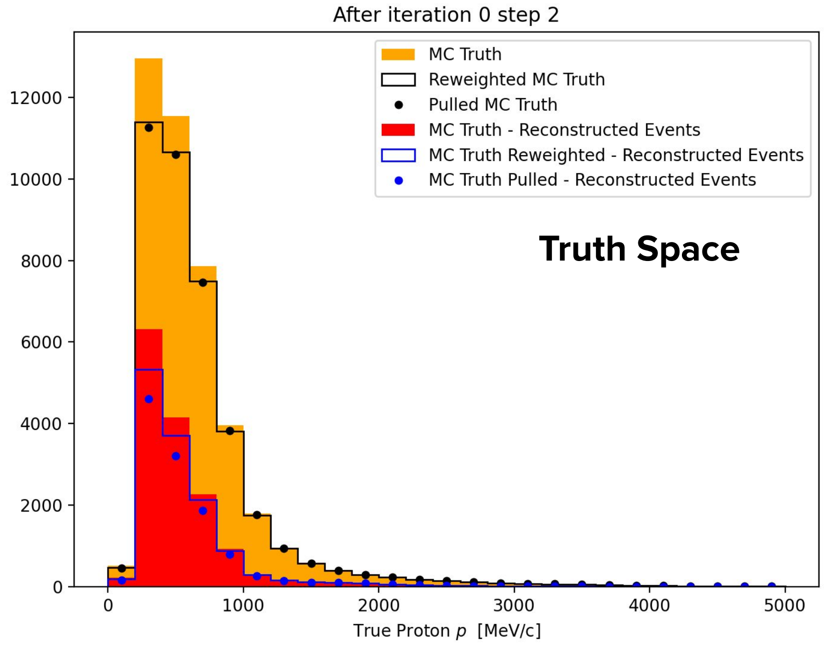
First Iteration – Step 1



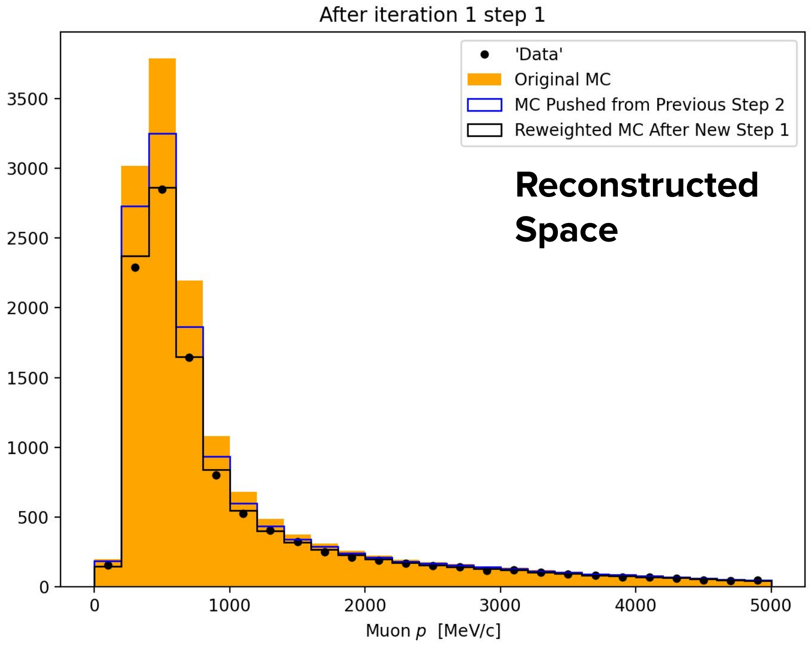


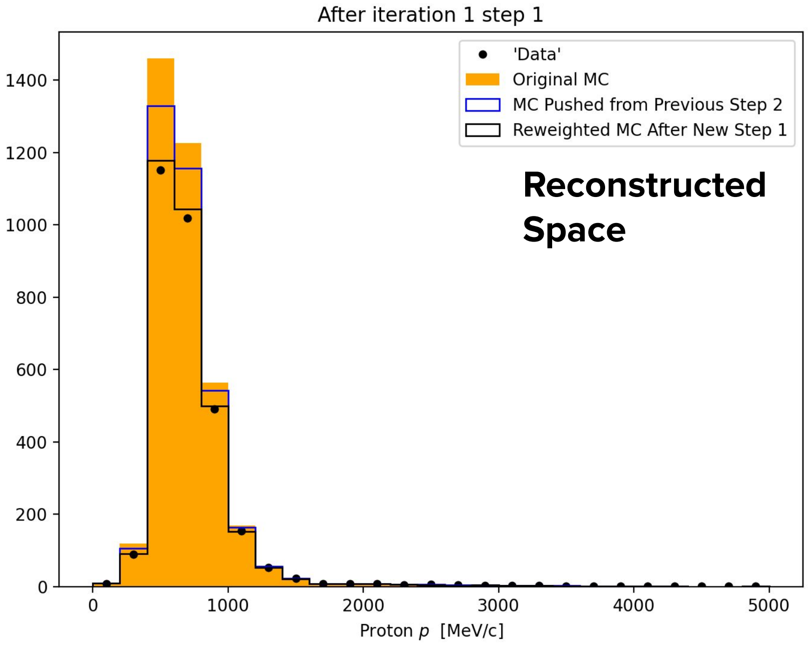
First Iteration – Step 2





Second Iteration – Step 1





Testing OmniFold

For a conventional benchmark, we compare the unfolding performance to what we get with iterative Bayesian unfolding (aka D'Agostini or Lucy–Richardson unfolding)

Using a series of mock data studies to assess the unfolding performance and network training / tuning

- "Easy": Reweighting with an arbitrary function changing event weights as a function of muon momentum
- "Hard": Reweightings that are functions of transverse kinematic imbalance variables, or based on neutrino interaction model parameter tuning

Uncertainty, efficiency, and ensembling

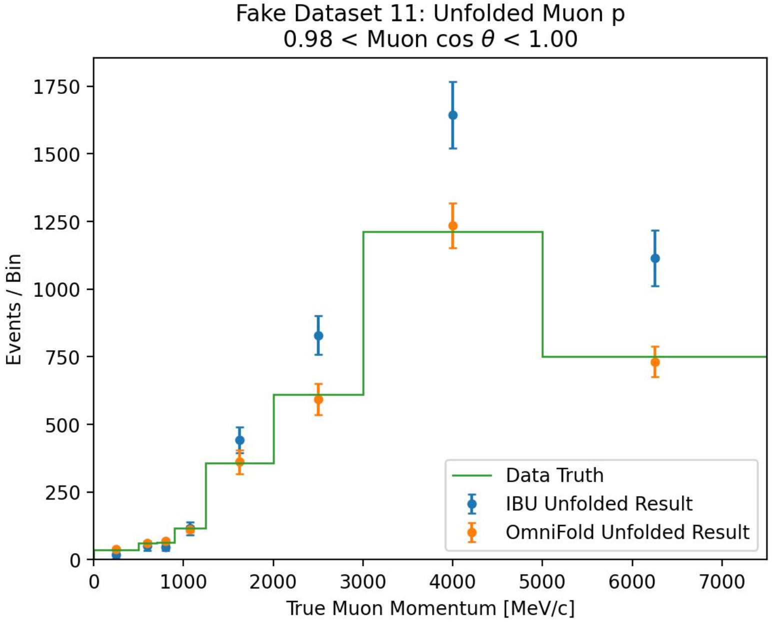
OmniFold naturally includes the efficiency correction when performing the unfolding (however this could be separated out and done as another step)

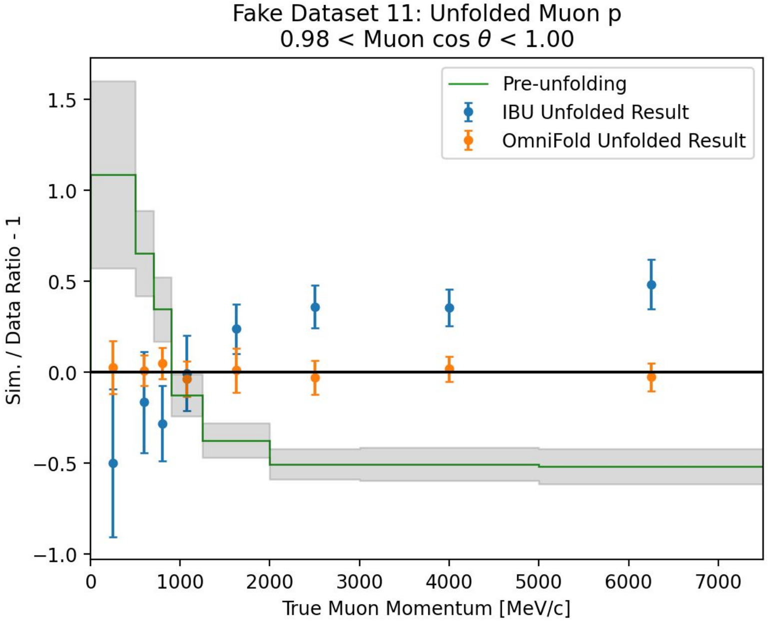
Systematic uncertainty is evaluated through a “universe” (toy throw) method where 500 different systematic variations are processed by OmniFold, and the spread in results gives an uncertainty band

Statistical uncertainty is evaluated by a bootstrap resampling with replacement where the “data” and MC event weights are varied for 500 throws / universes

The inherent randomness of training neural networks is mitigated by ensembling using the average result of several trials

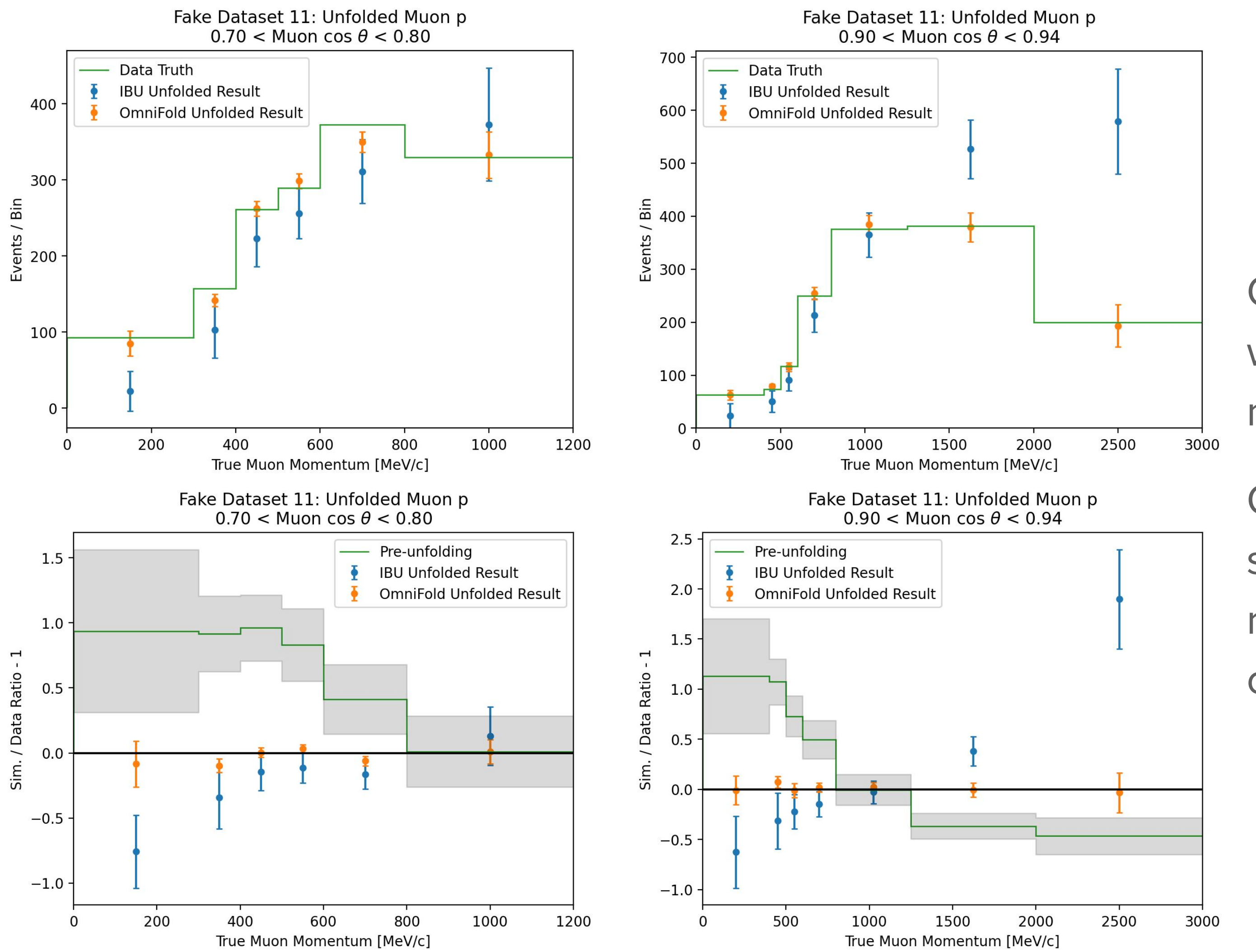
OmniFold results – muon momentum / angle





Slices of muon angle binned in momentum using T2K analysis binning (NB: OmniFold is unbinned)

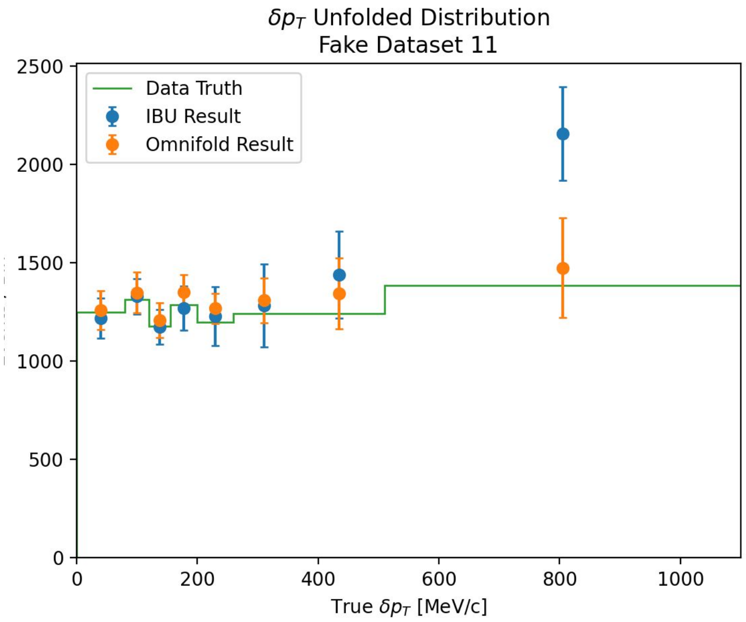
Error bars are from 500 stat+syst varied throws for both IBU and OmniFold

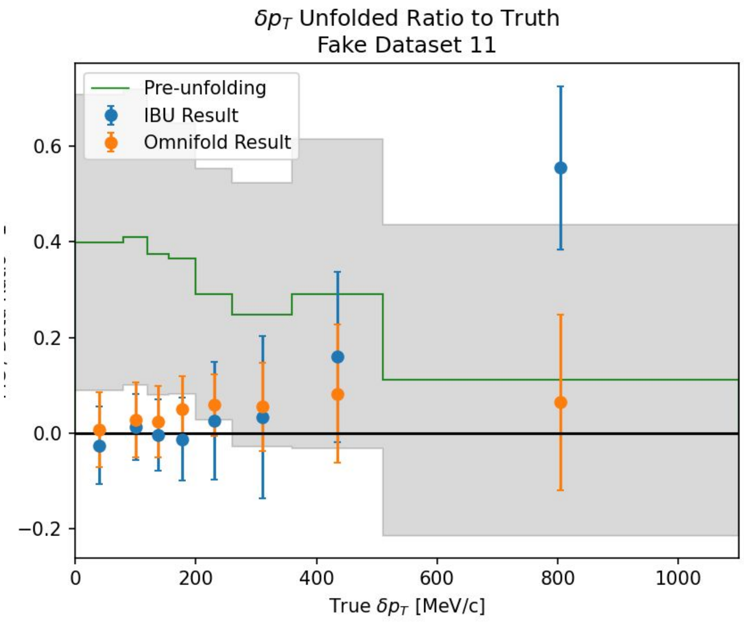


OmniFold does quite well recovering the mock data truth

OmniFold achieves smaller bias and reduced uncertainties compared to IBU

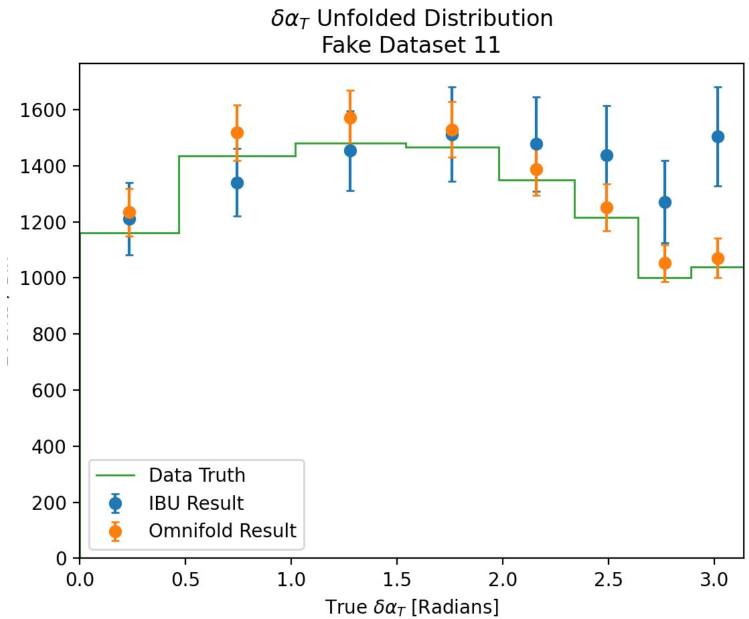
OmniFold results – transverse variables (pT)

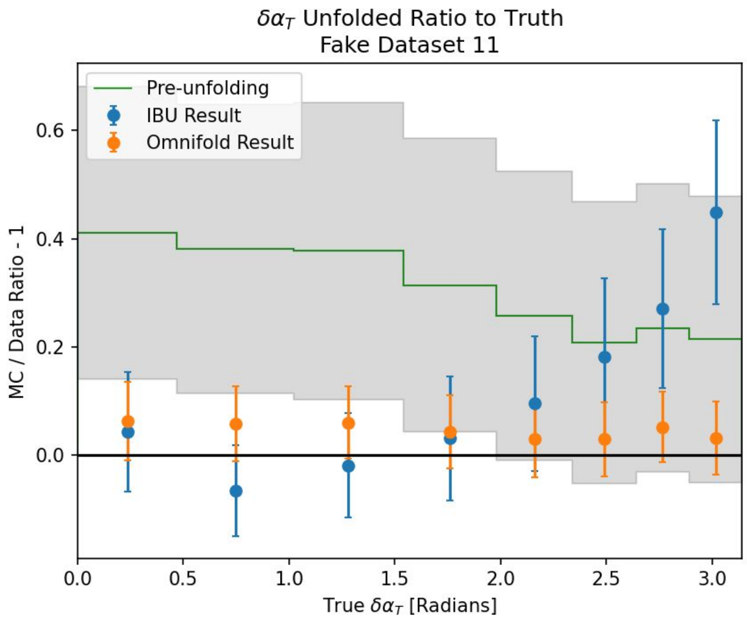




Since OmniFold is unbinned, can extract results in many variables simultaneously, while IBU was run separately for each variable

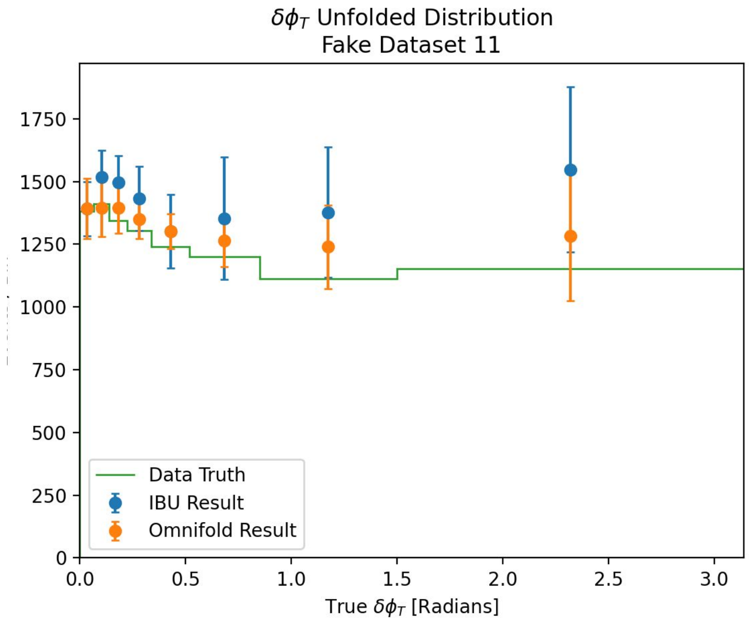
OmniFold results – transverse variables (αT)

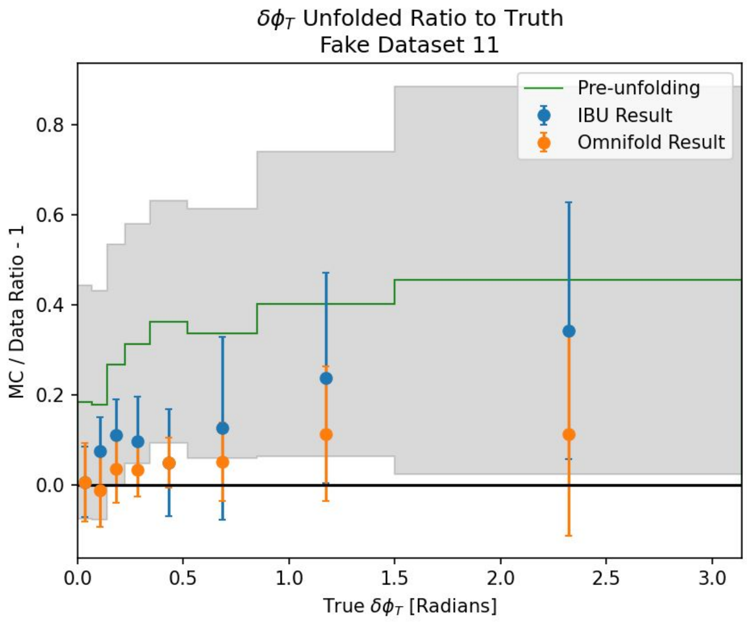




OmniFold still performs quite well, and overall better than IBU for the transverse variables

OmniFold results – transverse variables (φT)





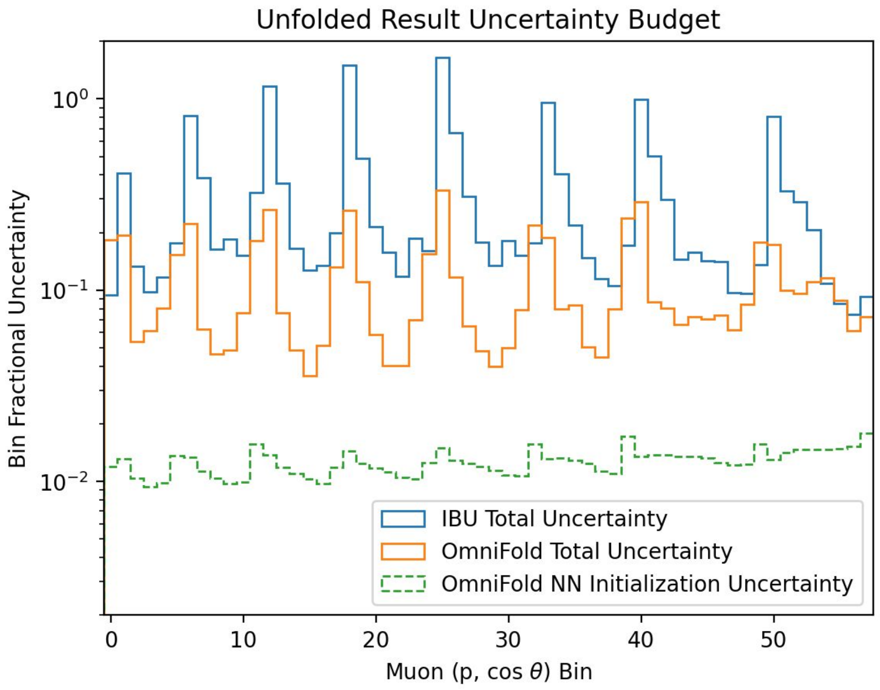
OmniFold still performs quite well, and overall better than IBU for the transverse variables

Uncertainty budget

Total uncertainty is the combination of **500 statistical and systematic varied throws** of OmniFold

Uncertainty from the randomness from the neural network initialization and training is reduced by averaging the results of multiple trials

Results presented here use five trials of OmniFold for ensembling

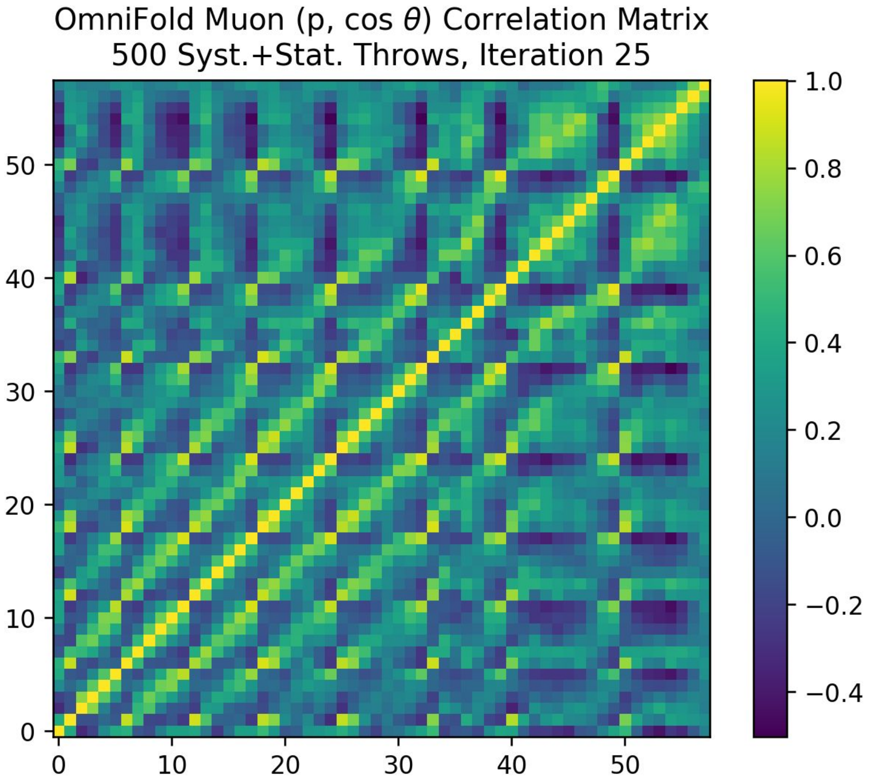


Correlation Matrix

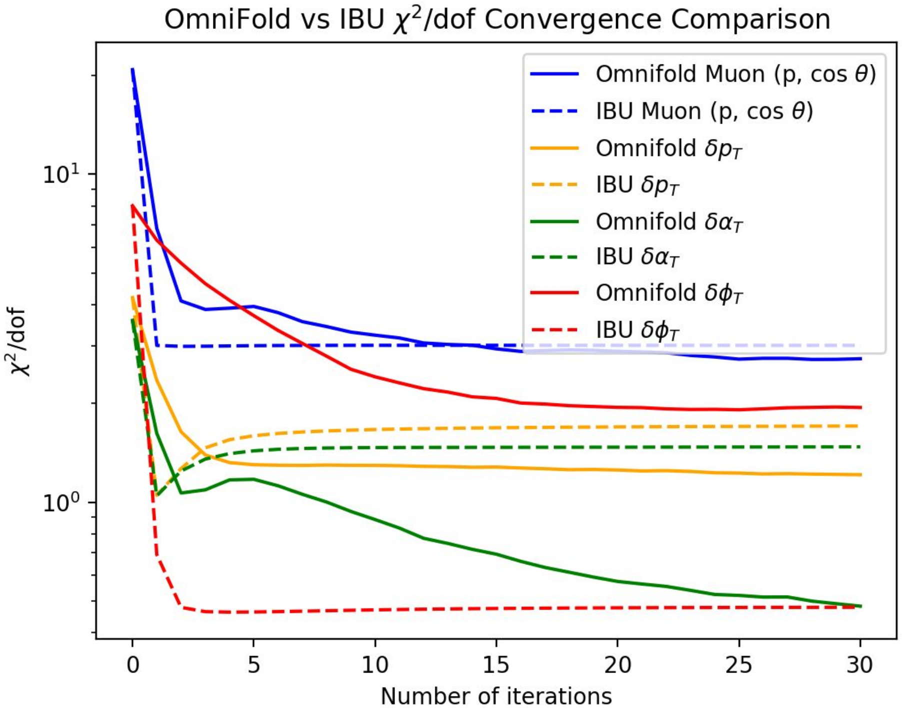
Covariance (and correlation) matrix can be calculated using the stat+syst varied throws

Calculated for muon kinematics for this example, but is available for every variable

Correlations also evolve with iterations in addition to the central value



Convergence Metric(s)



For recent tests, convergence is being monitored by calculating the chi-square for each iteration

Based on this, **OmniFold needs more iterations than IBU** and **can differ based on the variable / observable**

Requires knowing the observable and the kinematic binning, and only works when the truth is known

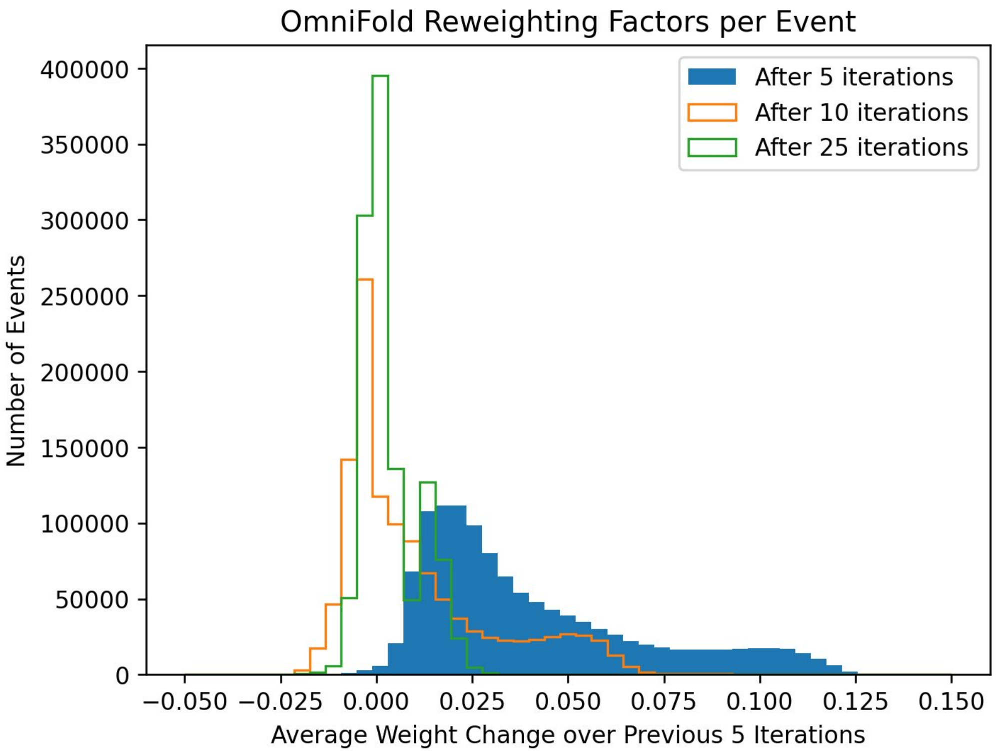
Other options being explored, particularly a way to do this unbinned

Unbinned Convergence Metrics

A convergence metric for OmniFold is ideally independent of binning and observable choices

One possibility: convergence achieved after event weights stop changing with more significance than internal classifier statistical fluctuations

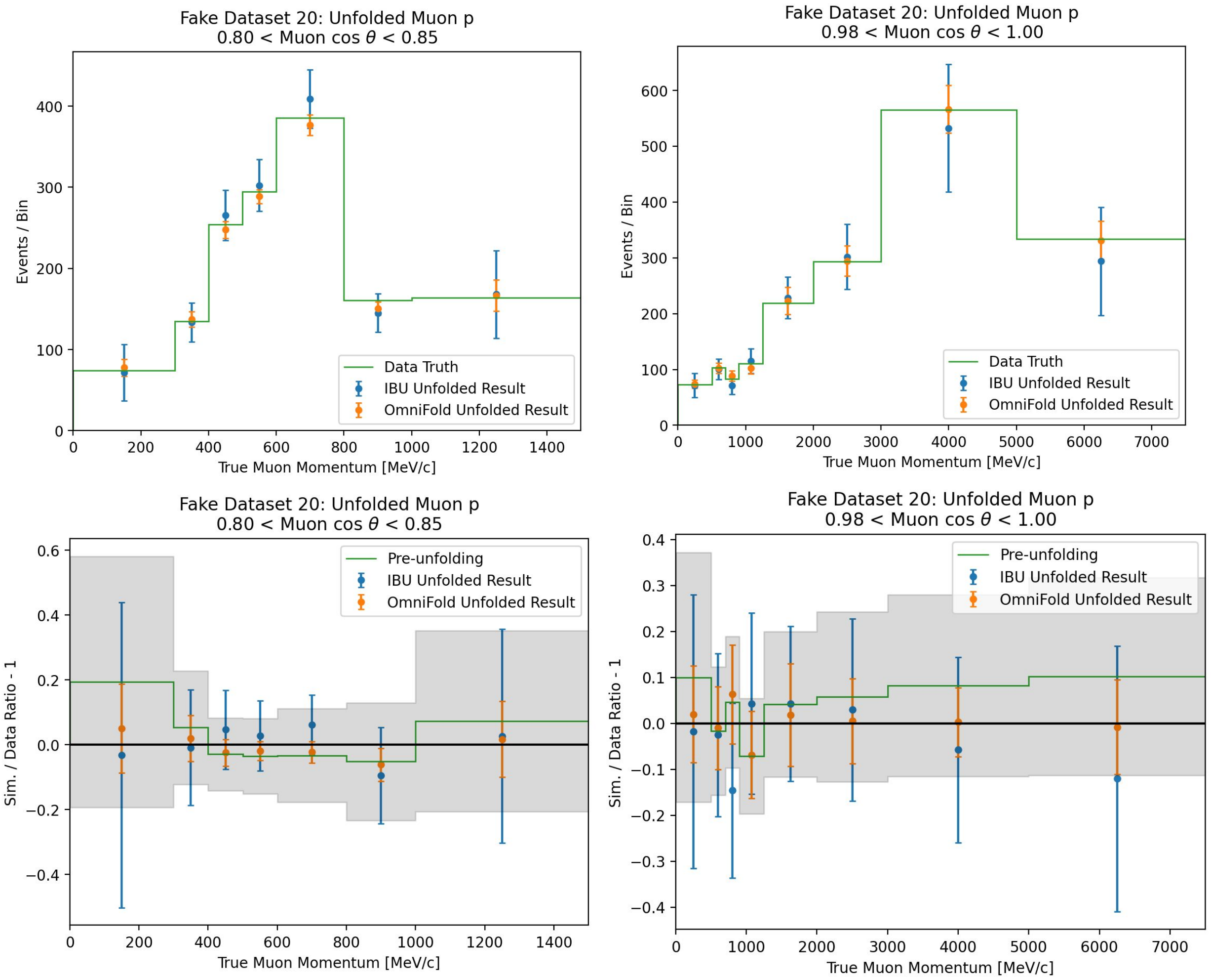
Plotting the average weight change of each event over the last 5 OmniFold iterations shows a trend toward convergence with more iterations



A Harder Fake Dataset

This fake dataset uses a reweighting based on δp_T, which is not a direct input to the OmniFold networks (but is derivable)

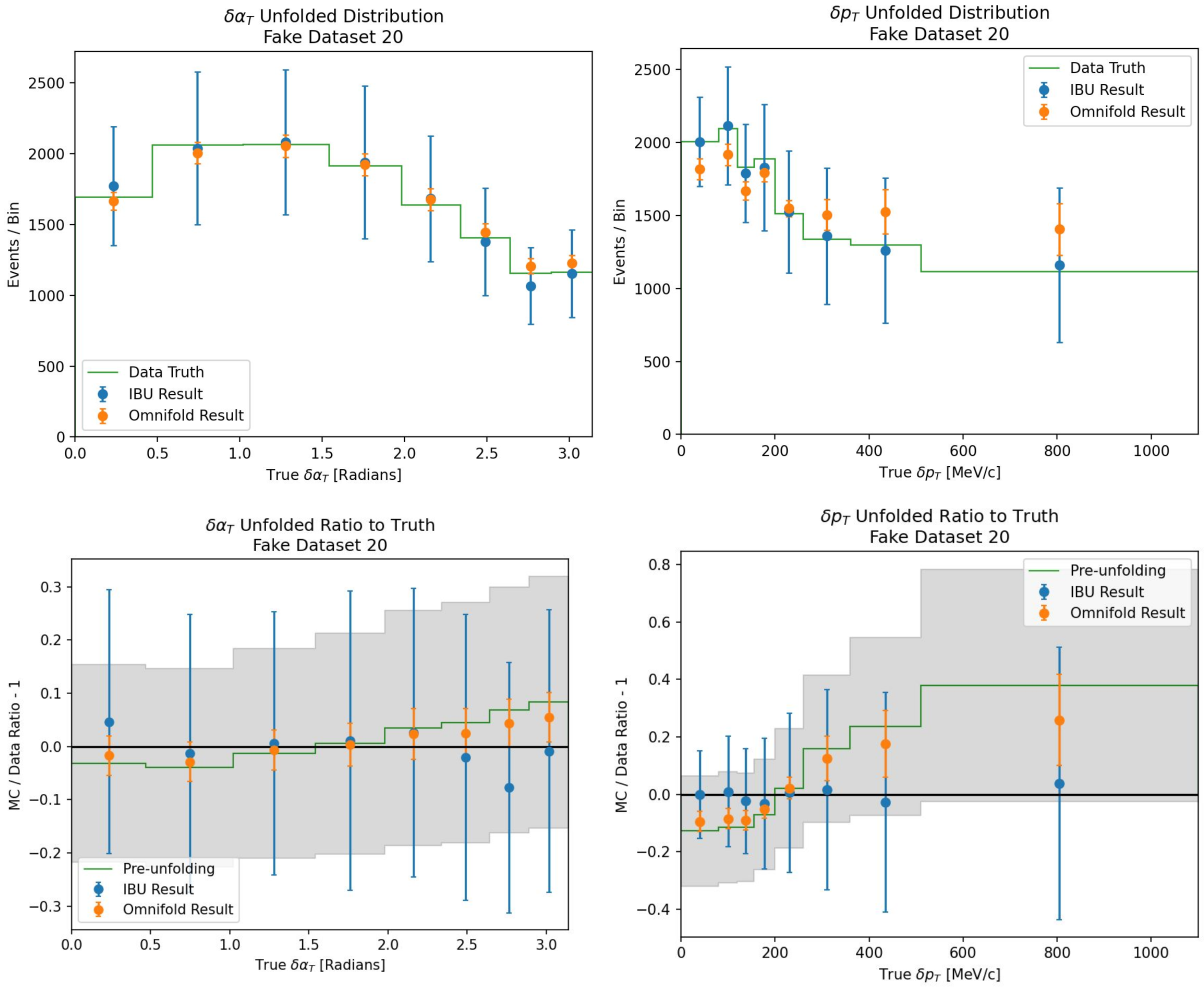
Unfolded muon momentum/angle distribution from OmniFold is comparable to or better than IBU results



A Harder Fake Dataset: STVs

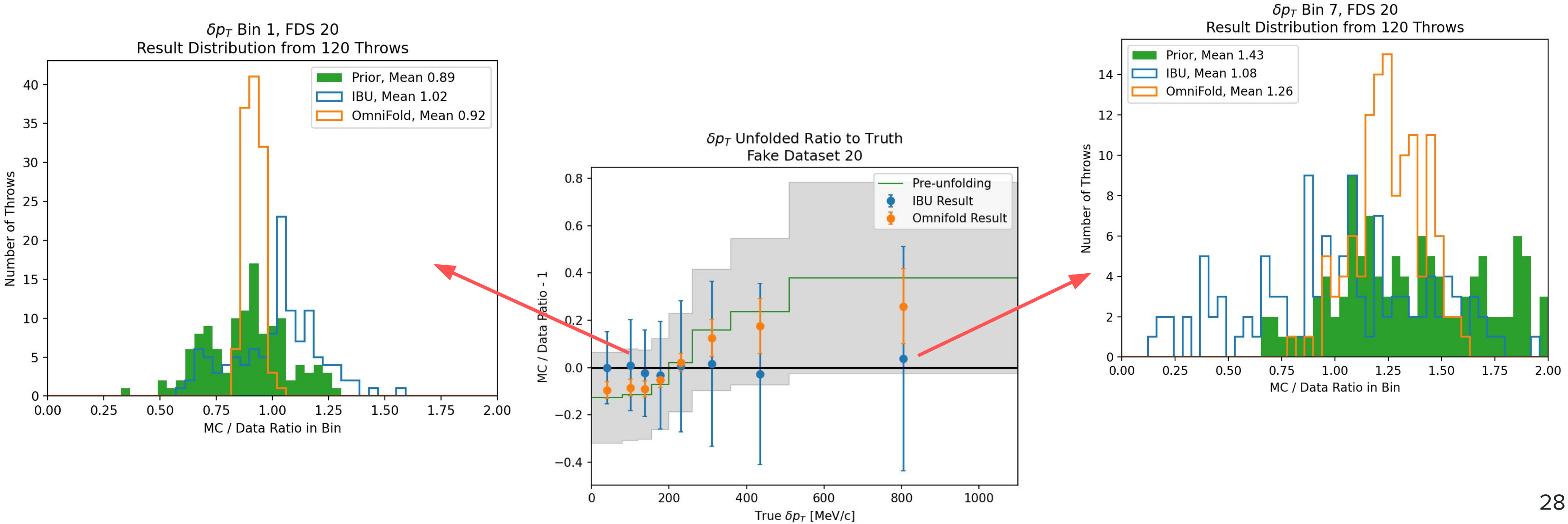
STV results from OmniFold are not as good compared to the previous test, but still decent

Note IBU gets direct access to these variables when unfolding



A Harder Fake Dataset: STV Results

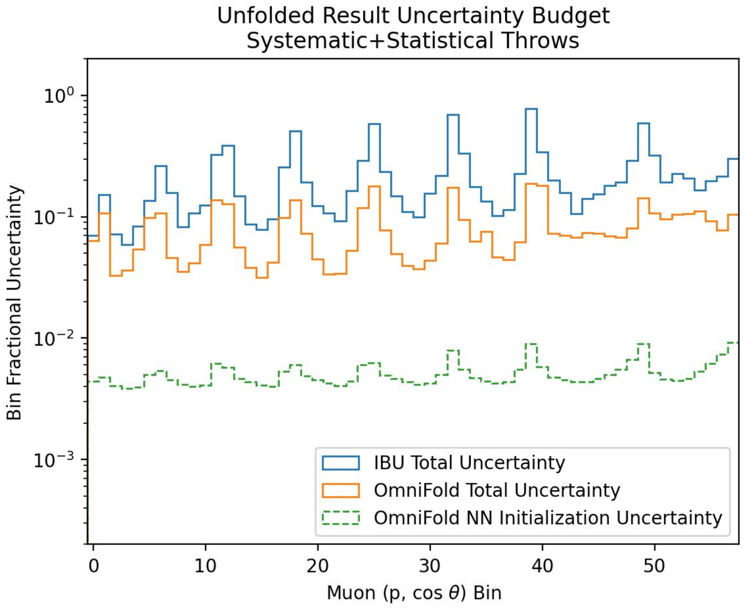
Looking into the distribution of unfolded results per bin, we see that the IBU result has relatively much larger variance

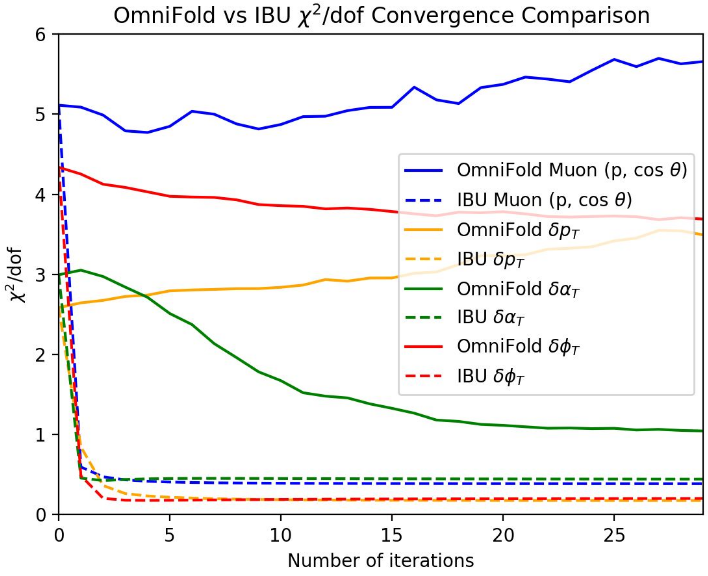


Harder Fake Dataset: Performance Comparison

But the chi-square results from OmniFold are clearly worse than from IBU

Some (but not all) of this is due to smaller uncertainties from OmniFold

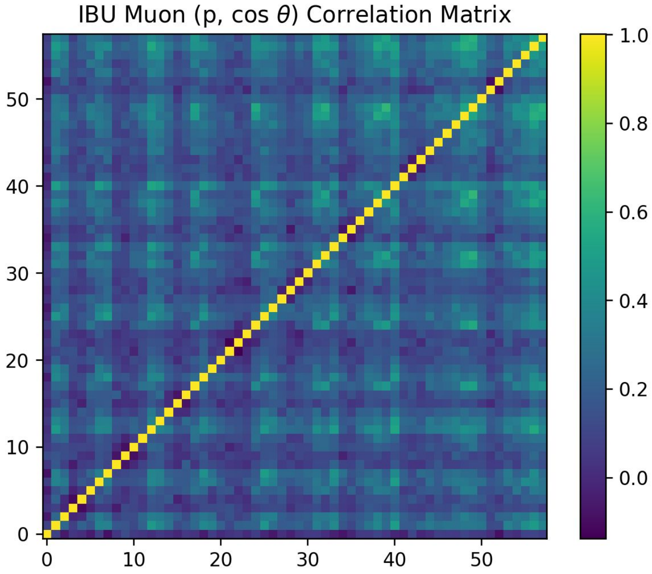


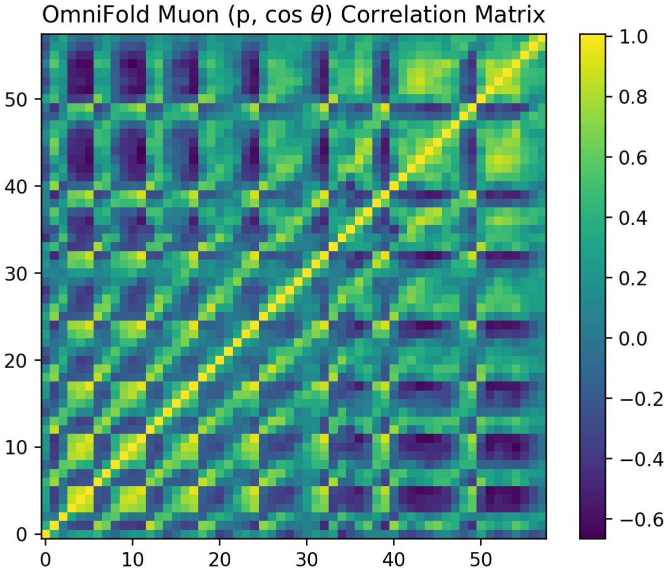


Correlation Matrices

Chi-square results are calculated using the mean and covariance of the binned results from the stat+syst varied throws

Calculated for muon kinematics for this example, but is available for every variable





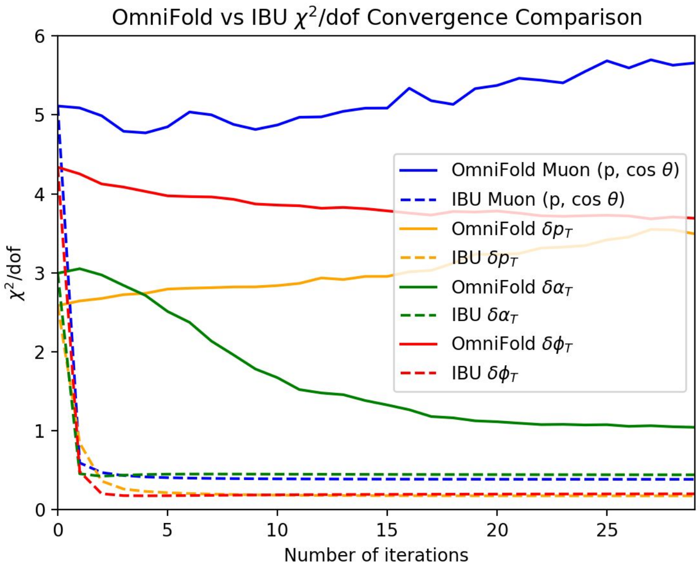
Correlations are much stronger with the OmniFold result

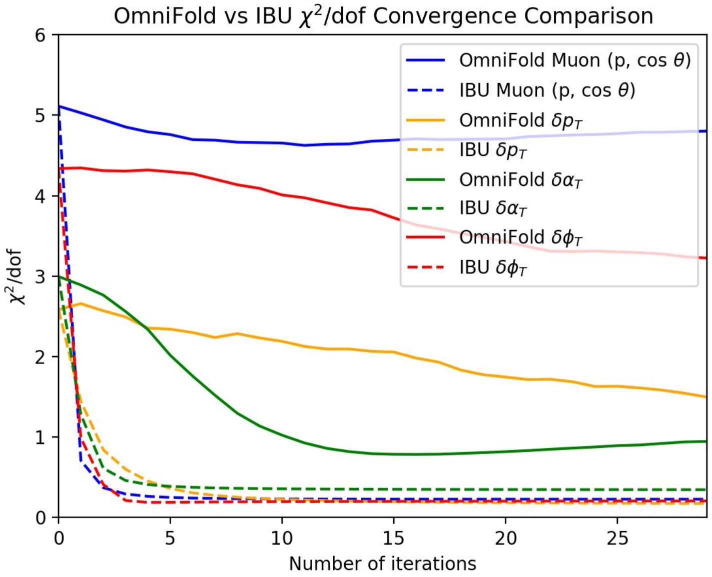
The checkered pattern corresponds to bins that are close in kinematic space

Simplifying OmniFold Inputs

If we offer δp_T as direct input to the OmniFold networks, we see improvement in the unfolding performance on δp_T

Without δp_T Input to OmniFold → **With δp_T Input to OmniFold**





Current Thoughts on OmniFold Performance

Classifier undertraining could explain the lower uncertainties, higher correlations, longer convergence times, and improvement with particular inputs from OmniFold

- Hard to rule this out completely as a cause, but we have tried scanning a large hyperparameter space, tried different network sizes, tried BDTs instead of NNs

One noteworthy statistic is we only have ~10k "data" events to train step 1 of OmniFold (comparison of data to reconstructed simulated events)

- The relatively low statistics are just a result of using a neutrino dataset
- Is this a limiting factor?

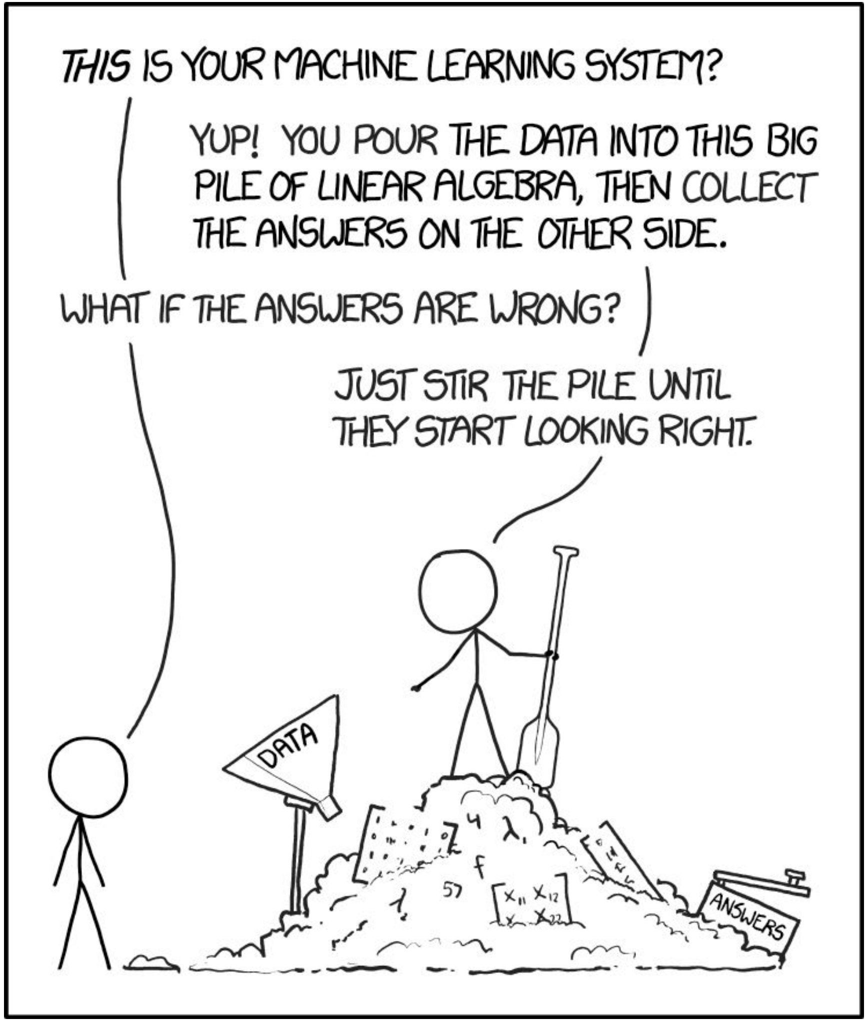
Summary & Future Work

OmniFold is an exciting technique for cross-section unfolding – naturally unbinned and high dimensional

OmniFold shows similar or better performance when compared to IBU, and automatically allows for unfolding in any number of variables

Several investigations are still ongoing on the performance for more complicated mock data studies and the impact of network architecture and hyperparameters

Looking forward, we want to test OmniFold on a more realistic analysis (e.g. T2K or even the 2x2) and are considering using a more sophisticated ML architecture or evolution of OmniFold


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

xkcd 1838