Training with Real Rata :
Signal Extraction via Density Ratios

DEREK GLAZIER

UNIVERSITY OF GLASGOW, SCOTLAND

RICHARD TYSON

JEFFERSON NATIONAL LABORATORY, US

Digital Twins for Nuclear and Particle physics - NPTwins 2024

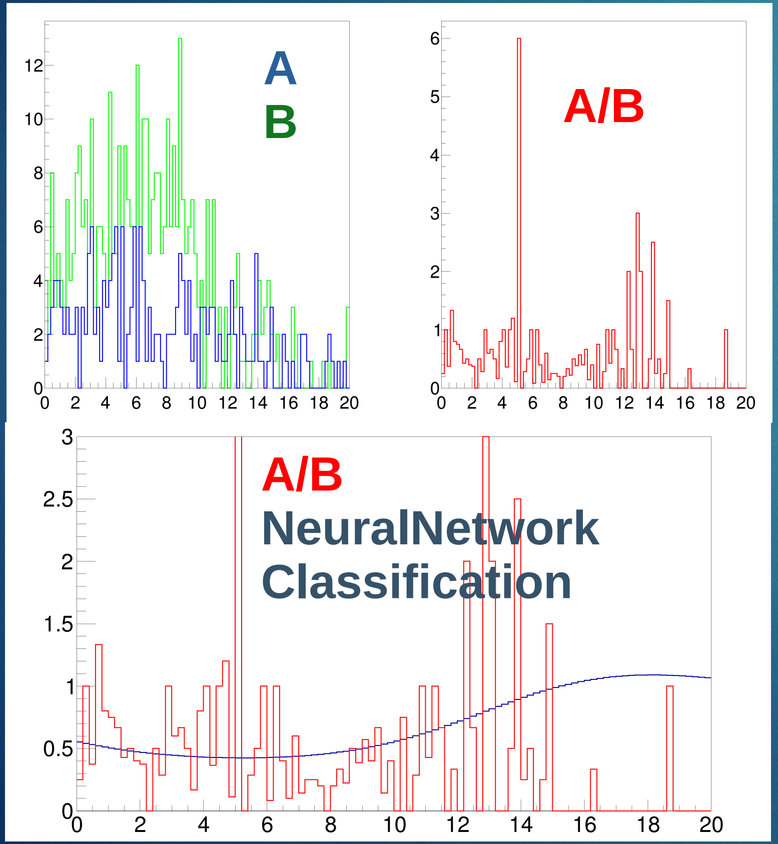
16–18 Dec 2024Museo Diocesano di Genova

Based on:
DIG, RT,
arXiv:2409.08183 (2024)

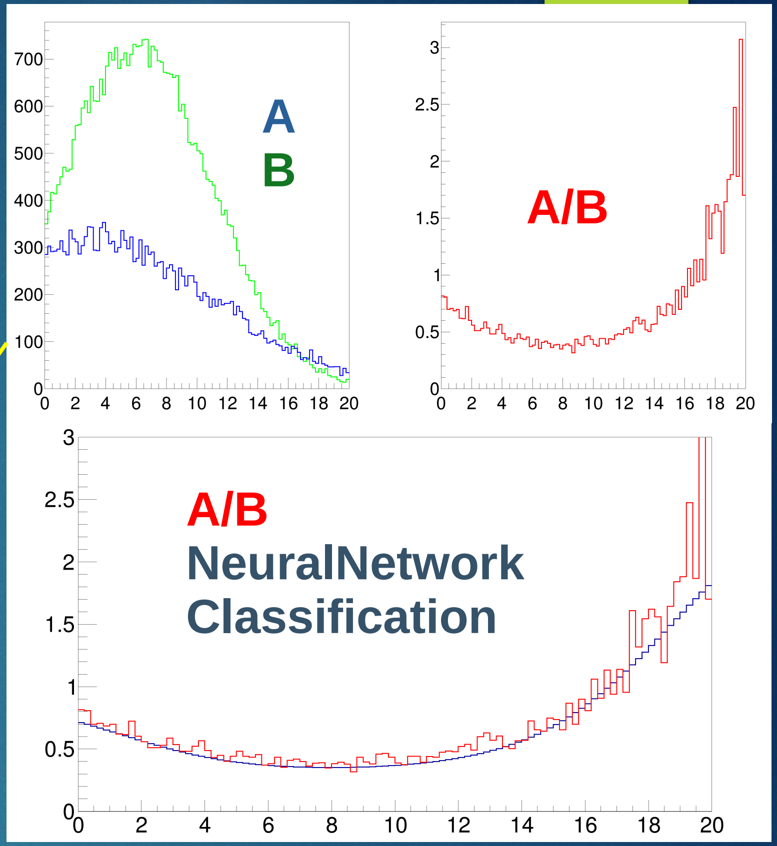
Density Ratios

- Often we require the ratio of 2 different distributions : **Density Ratio**

- In 1D we may just use 2 histograms and create a 3rd which is their ratio



← **100x less events**



- But in many dimensions this become infeasible
- Similar to having very low statistics
- Alternatively we can use Machine Learning classification tasks which are suited to such problems

Previous Work with Density Ratios

arXiv > physics > arXiv:2207.11254

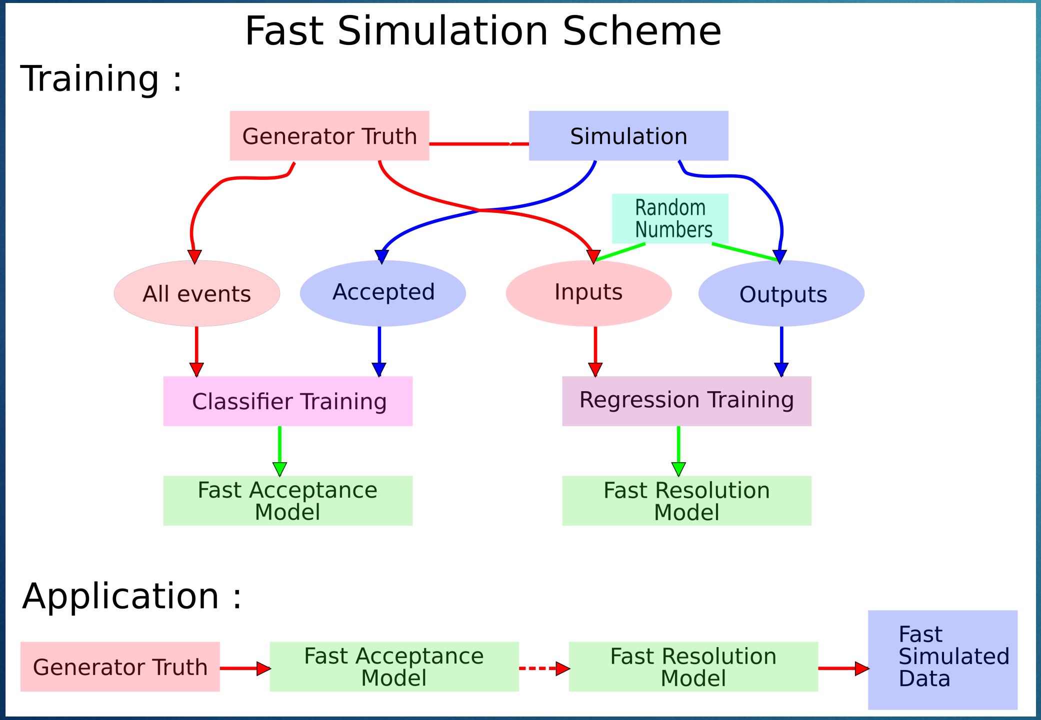
Physics > Data Analysis, Statistics and Probability

[Submitted on 22 Jul 2022]

Machine Learned Particle Detector Simulations

D. Darulis, R. Tyson, D. G. Ireland, D. I. Glazier, B. McKinnon, P. Pauli

https://arxiv.org/abs/2207.11254



Excellent tool for mapping acceptance probabilities in multi-dimensions
i.e. probability a particle is detected at particular point

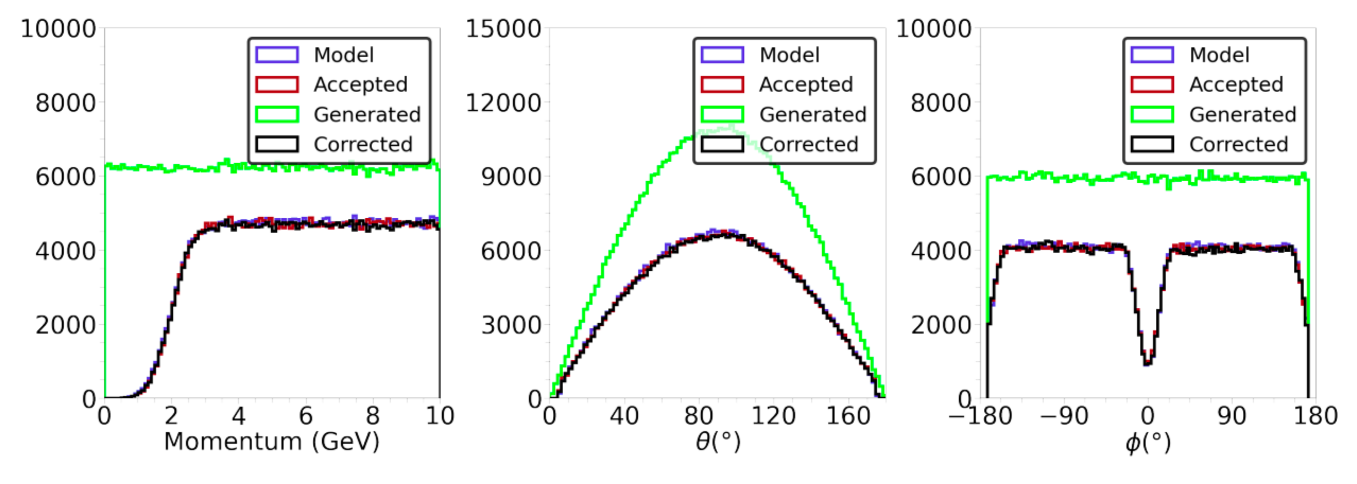


Figure 11: Results of applying a neural network with a Gaussian transform for acceptance modelling with a BDT correction. The BDT used 100 weak learners with a maximum depth of 10 and a learning rate of 0.1. The network used is the higher capacity model with 4 hidden layers of 512, 256, 128, and 16 neurons respectively. The improvement in the 3-vector component distributions is smaller than in the case of the low capacity network.

- Optimised algorithm using combination of Neural Networks and BDT for multi-dimensional correlations and accuracy

Full Reaction Simulations

Momentum resolutions / correlations Toy Simulation

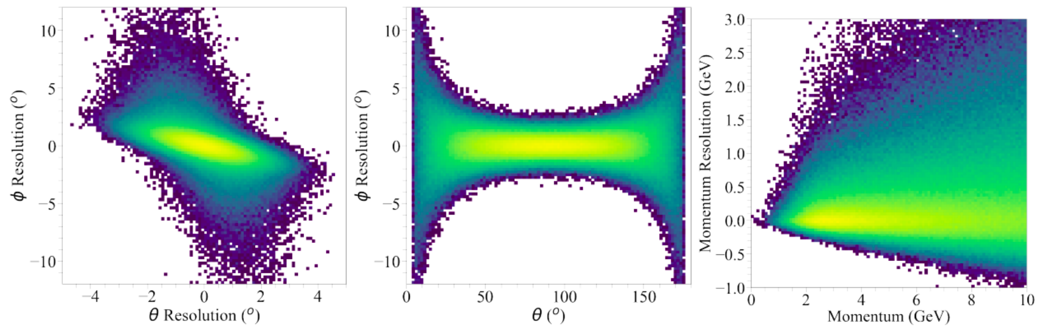
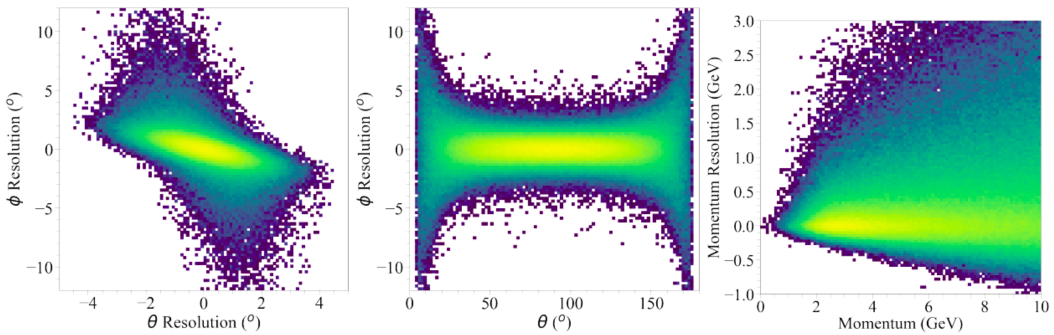


Figure 3: Some of the multidimensional correlations in the toy detector reconstruction. It is important that the machine learned simulation can reproduce these features.

Momentum resolutions / correlations ML Simulation



- Resolutions mapped with Decision Tree inference

Kinematic distributions : invariant masses decay angles

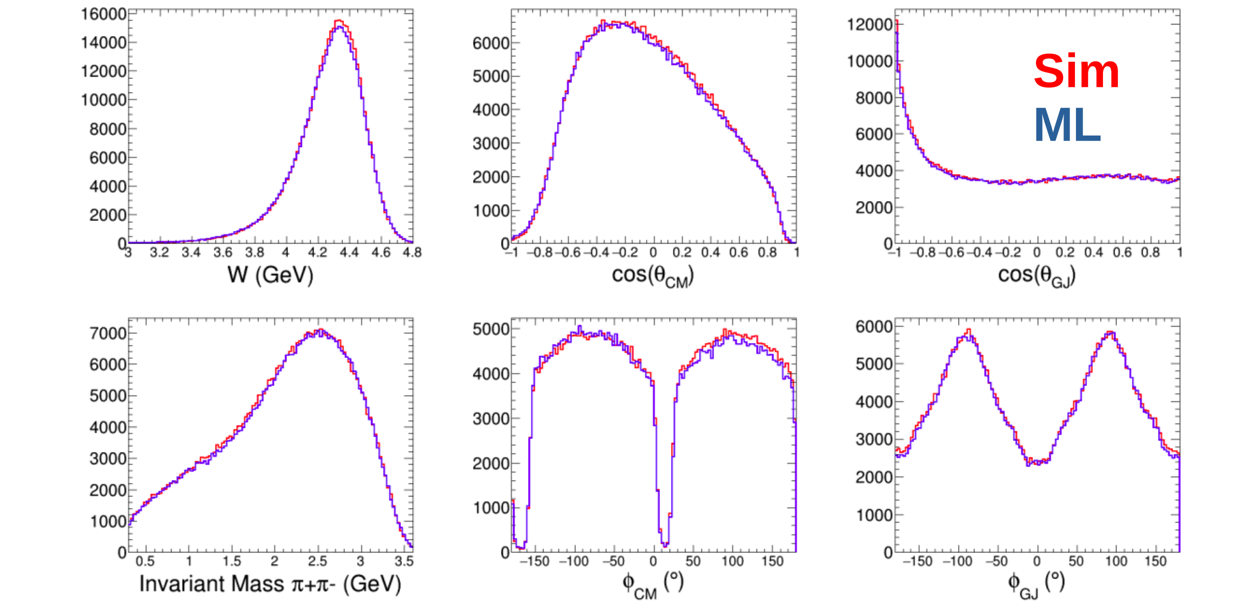
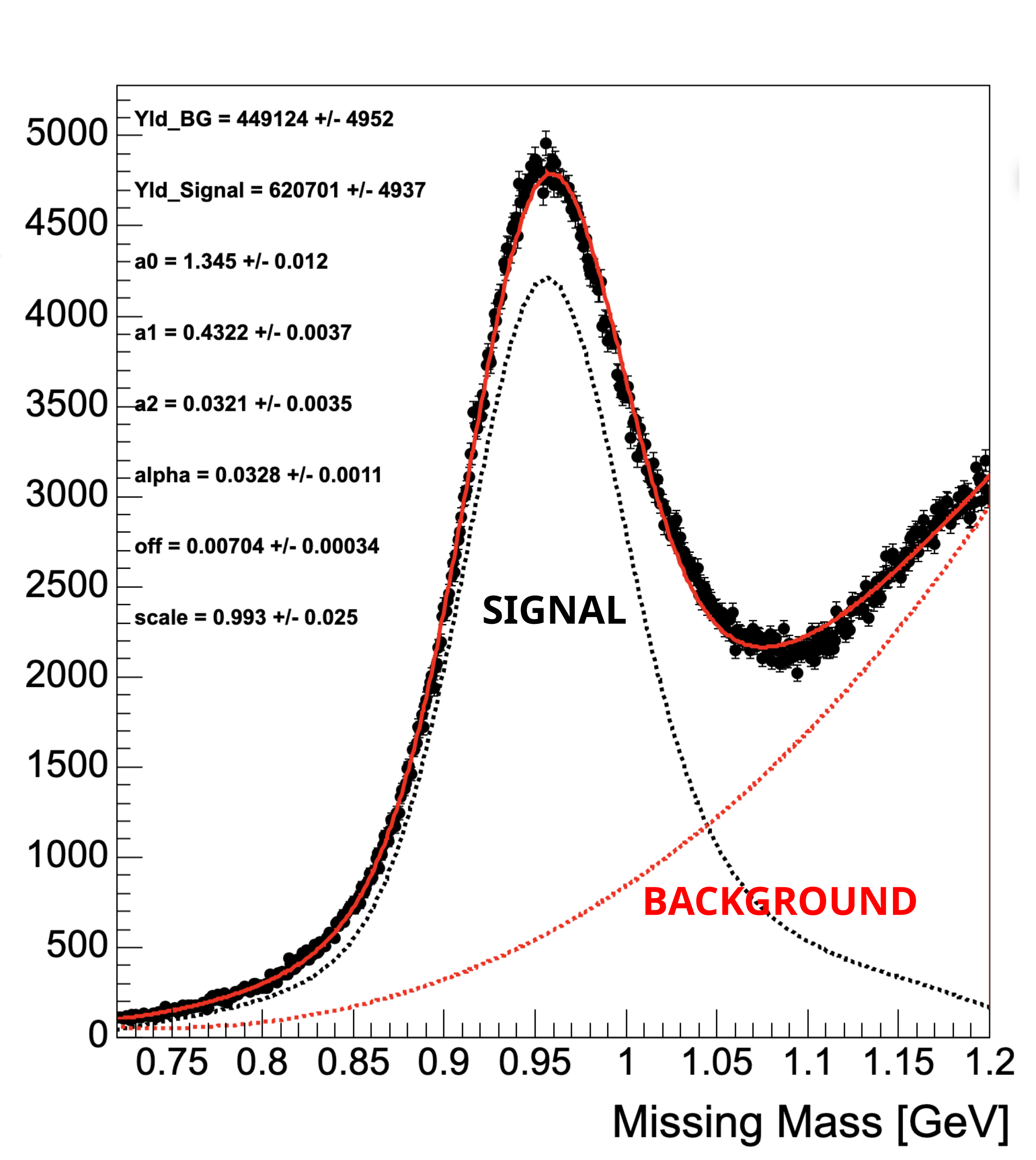


Figure 26: Accepted and reconstructed physics variables for the Fast (blue) and Toy (red) simulations of the 2 pion photoproduction reaction. The distributions show: the invariant mass of the three final state particles, W; the invariant mass of the two pions, $M(2\pi)$; the production angles in the centre-of-mass system $(\cos(\theta_{CM}), \phi_{CM})$; and the decay angles of the two pions.

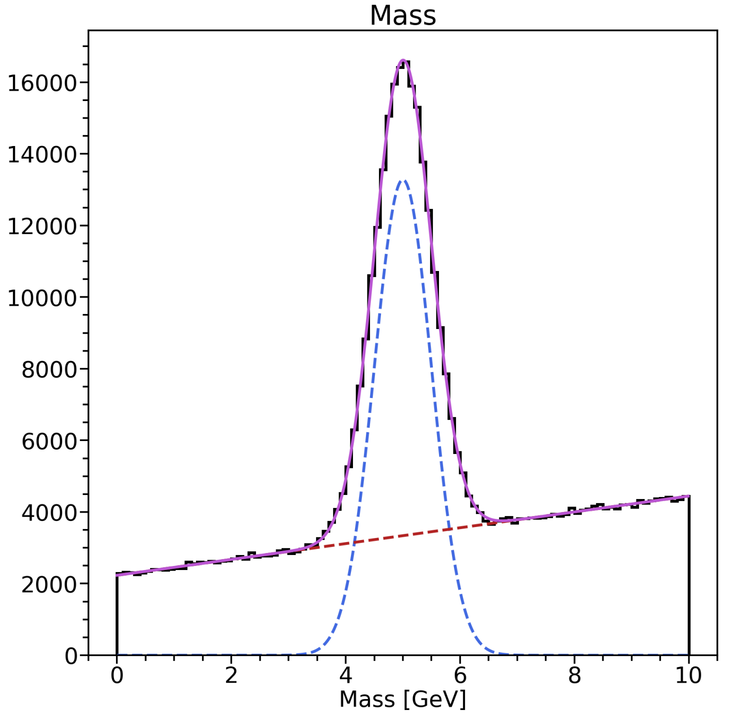
Machine Learning With Experimental Data

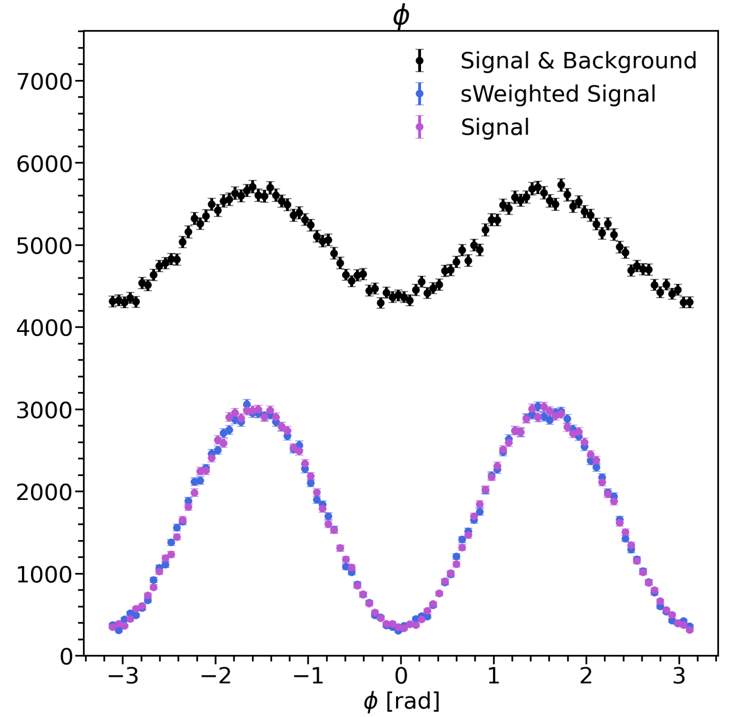
- Often train ML with simulated data – requires excellent agreement between simulation and experimental data.
- Instead we can train ML with experimental data – this relies on being able to separate contributions from different event sources in the data.
- In the example plot, we would need to separate the background and neutron signal to use the neutron data in training.



sPlot and sWeights

- The sPlot formalism aims to unfold the contributions of different event sources to the experimental data.
- sPlot generalises side-band subtraction weights to where there is no clear isolated background to subtract from the total event sample.
- The data is assumed to have:
  - discriminating variables where distribution of event sources are known
  - control variables where distributions of event sources are unknown.
- Fit expected pdf to discriminating variables to obtain sWeights that allow to reconstruct distribution of control variables.
- Requires that the discriminatory variable and control variables are independent of each other.





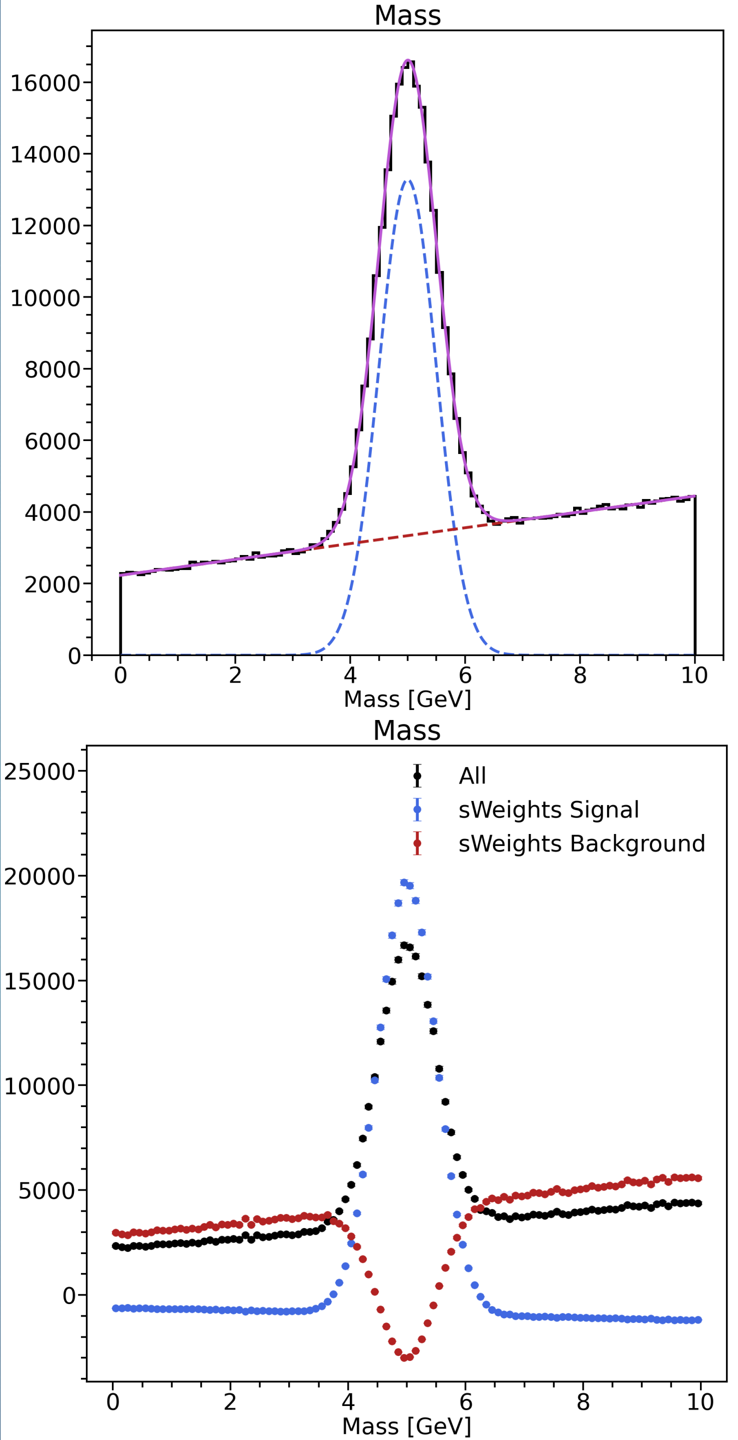
M. Pivk, F. R. Le Diberder. SPlot: A Statistical tool to unfold data distributions. *Nucl. Instrum*

Negative sWeights

- Essential characteristic of sWeights is that they can be negative.
- Necessary to preserves the statistical properties of the dataset eg correct uncertainties and normalisation.
- Creates issues for ML training : -ve weights in general allow loss to become arbitrarily negative
- Circumvent this issue starting with sample weighted binary cross entropy loss

$$L(f(x_i)) = -\sum_i w_i(y_i \log f(x_i) - (1-y_i)\log(1-f(x_i)))$$

- And convert sWeights to positive definite probabilities through density ratio classification task.



Density Ratio Weights (drWeights)

- For this we can use density ratio estimation:
  - Summing the sWeights for a given species recovers the yield of that species.
  - Define weights for a given species equivalent to the ratio of its probability density over the sum of probability densities of all species in the data ie

$$W_{dr}(x_i) = \frac{D_S(x_i)}{D_S(x_i) + D_B(x_i)} = \frac{D_S(x_i)}{D_{all}(x_i)} \, .$$

- To convert the signal weights we create a training sample with "all events weighted by signal sWeights" as class 1 and "all events weighted by 1" as class 0.

- Avoids the issues due to negative weights as all events in class 0 are contained in class 1

  Requires : $\sum w_i < N$ (number of events). True by definition of signal sWeights

- ML classifier with this training sample will have output for signal $f(x_i)$:

  Then transform to probability W_{dr}

$$f(x_i) = \frac{D_S(x_i)}{D_S(x_i) + D_{all}(x_i)}$$

$$\Rightarrow W_{dr}(x_i) = \frac{f(x_i)}{1 - f(x_i)} \, .$$

See also Nachman/ThalerNeural : resampler for monte carlo reweighting with preserved uncertainties. Phys. Rev. D, 102:076004, Oct 2020

Density Ratio Weights (drWeights)

- The solution to this is to convert sWeights to positive definite probabilities.
- For th
  - Su
  - sW … es of all species in th
- To co … ghts as class 1 and all eve
- Avoid … as the sum of weigh
- ML al

Create the training sample with all events weighted by signal sWeights as class 1 and all events weighted by 1 as class 0.

Two key takeaways are:

Creating the training sample in such a way allows to use the binary cross-entropy loss function even in the presence of negative sWeights.

Creating the training sample in such a way allows a binary classification model to convert the signal sWeights to positive definite probabilities.

Couple of Notes

- sPlot requires that the discriminatory variable and control variables are independent of each other.

⇒ conversion should only be made with the control variables

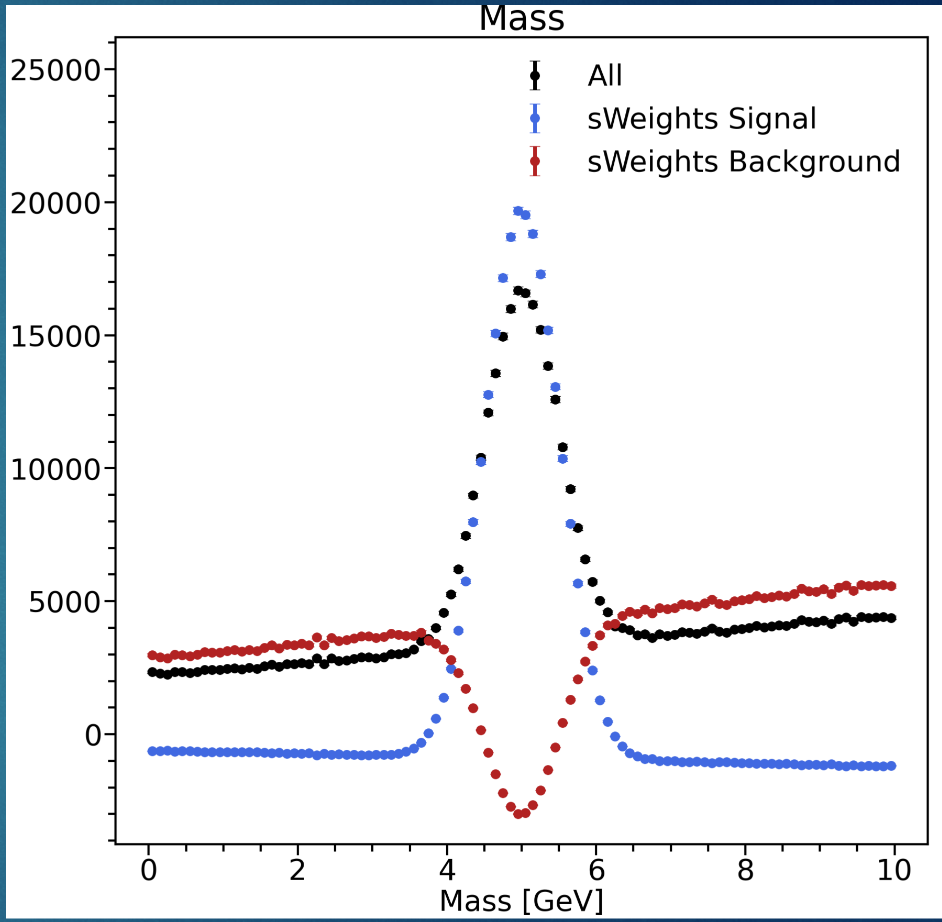
- sPlot unfolds the control variable distributions

⇒ conversion works only at the distribution level and not on an event by event basis.

- sWeighted uncertainty is calculated by taking the sum of the squared sWeights.

⇒ This doesn't work with converted weights W_{dr}.
 But we can just propagate sWeight sum of squared weights

- We can apply the method twice, ie correct the drWeights for better results.



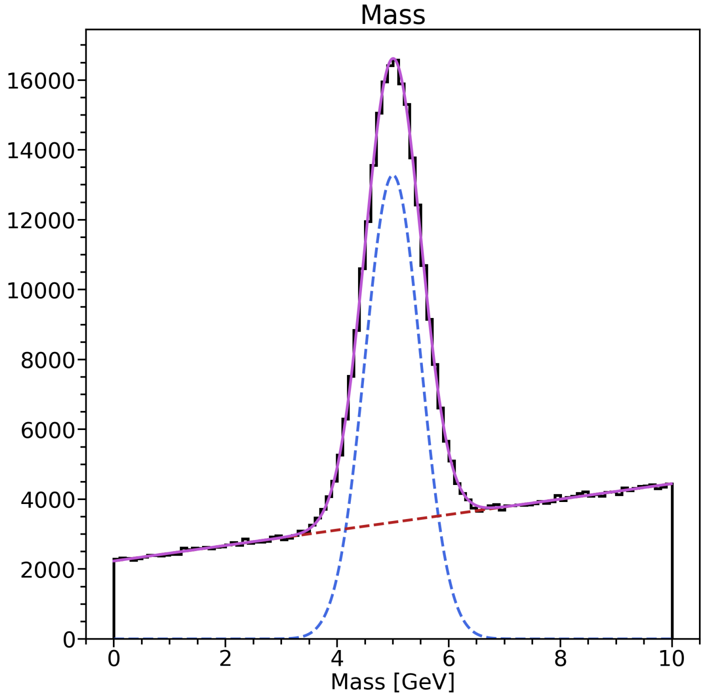
Toy Example

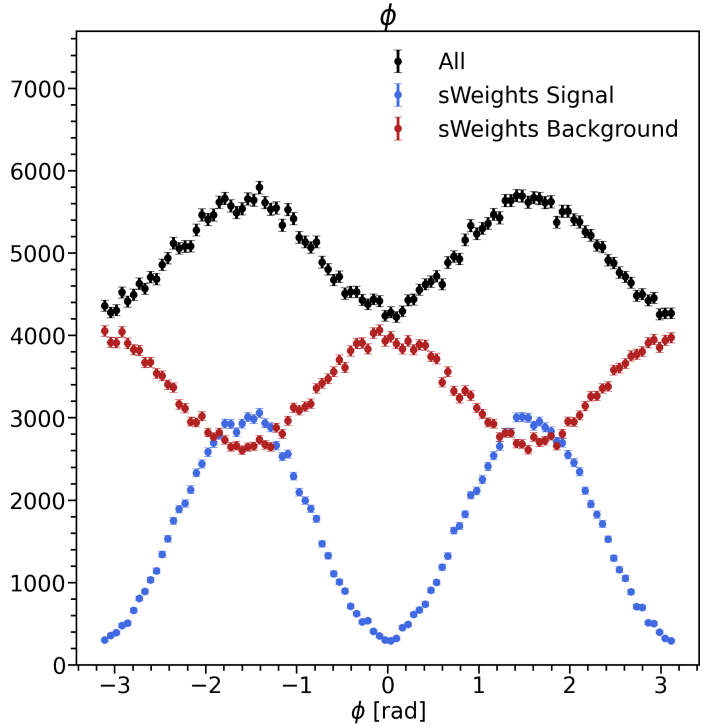
- Create toy event generator to produce three dimensional events:
  - mass such as an invariant mass as discriminatory variable
  - azimuthal (ϕ) angular distribution
  - z = cos θ.
- Signal events were generated with a Gaussian distribution in mass and a cos 2ϕ distribution of amplitude 0.8.
- Background events were generated with a Chebyshev polynomial distribution in mass and a cos 2ϕ distribution of amplitude -0.2.
- The aim is to measure the signal asymmetry in ϕ by unfolding the signal distribution in the control variable ϕ .

Fit of signal and background PDFs allows us to determine sWeights

Applying sWeights To Φ distributions gives our signal cos(2Φ)
Fit to this (blue) to get back our amplitude of 0.8

See github repo





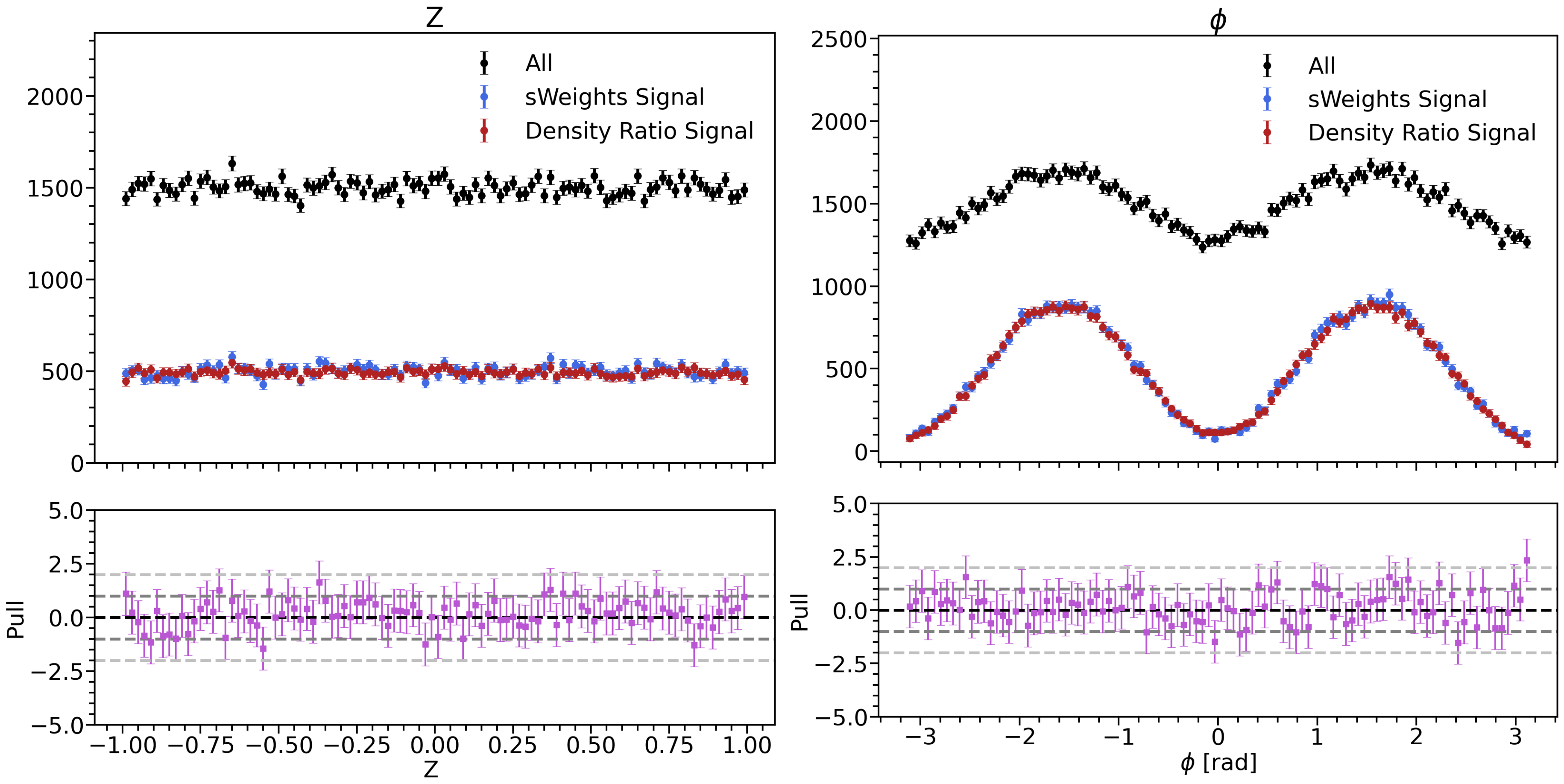
Apply two consecutive Gradient Boosted Decision Trees to convert sWeights.

→ Second acts as reweighter fine-tuning results

→ Measurably improves results

Several other learning models tested, generally good performance.

Training rate ~2 kHz, prediction rate ~500 kHz on 5 cores of a AMD EPYC 9554 64-Core Processor at 3.1GHz.



Excellent agreement Between blue sWeights and red drWeights

Deviations not statistically significant

Quantifying how well it works

- Want to reproduce signal ϕ asymmetry amplitude of 0.8.
- Repeat training the density ratio model and fitting ϕ asymmetry for 50 independent toy datasets of 100k events
- Use Signal to Background ratio of (1:2) or (1:9).
- Obtain mean amplitude and uncertainty along with the standard deviation of the amplitude over the 50 datasets
- Fit performed via binned $\chi 2$
- The expectations are:
  - mean should be consistent with the nominal value of 0.8
  - mean uncertainty and standard deviation should be numerically similar i.e. the fluctuation of results is consistent with the calculated uncertainty
  - i.e. **σ / Uncertainty** ~ 1.0

Method	Mean	σ RMS (50 fits)	σ / Uncertainty
sWeights (1:2) (1:9)	0.802 ± 0.0089 0.804 ± 0.0274	0.0082 0.0244	0.92 0.89
drWeights (1:2) (1:9)	0.807 ± 0.0092 0.793 ± 0.0285	0.0093 0.0260	1.01 0.91

Number of Events

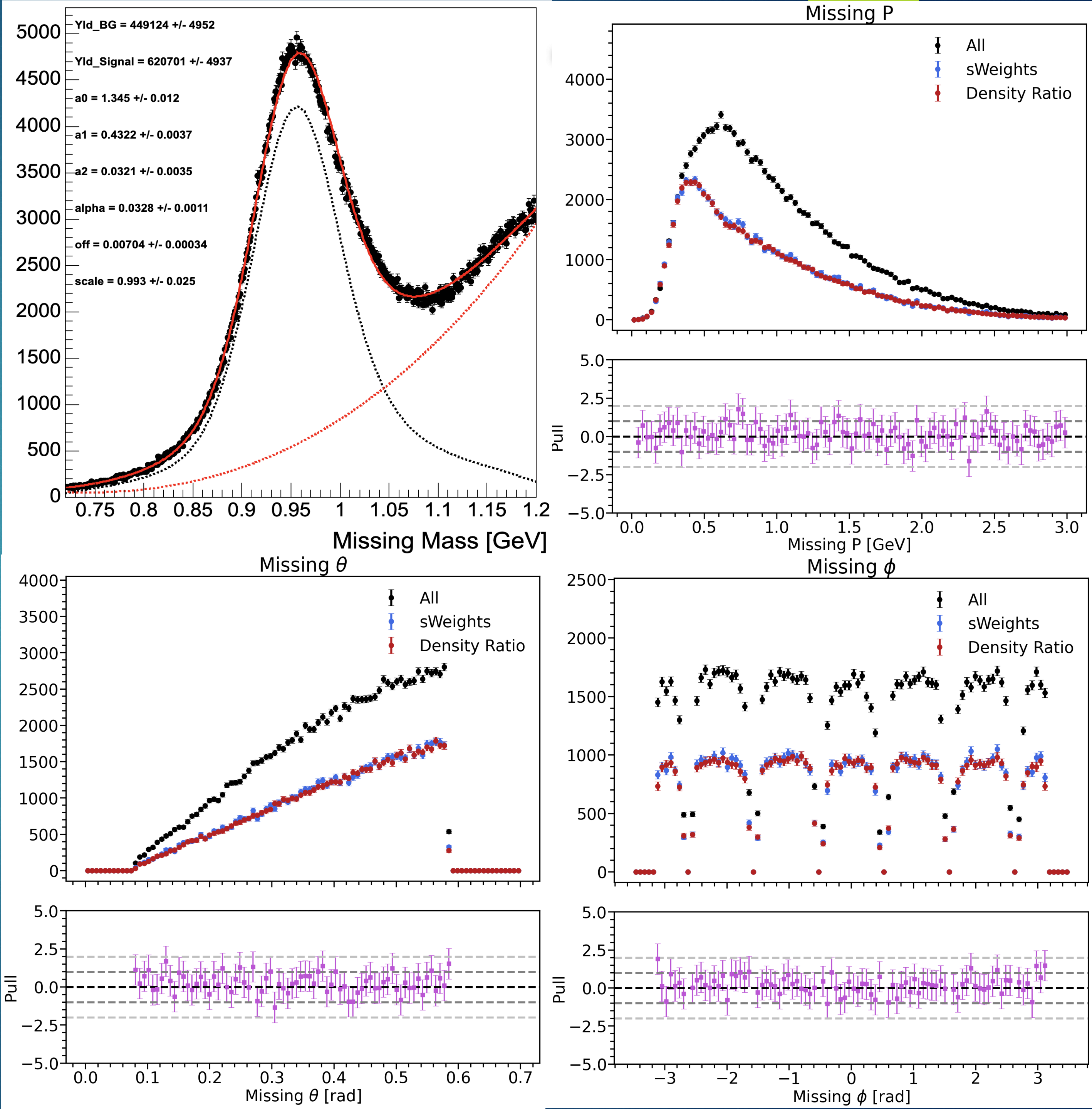
- Same test as before, vary number of events.
- Use signal to background ratio of (1:9).
- At 1000 events we have only 100 signal events.
- drWeights are robust and function well with large backgrounds and limited statistics.
- Issues with sWeights at low event number due to -ve bin contents in binned χ2
- Expected behaviour when use event based maximum likelihood instead

# Events Weights	Mean	σ	σ Uncertainty
1000 sWeights drWeights	17.94 ± 14.67 0.679 ± 0.5902	84.81 0.2710	5.78 0.46
10,000 sWeights drWeights	0.870 ± 0.0953 0.778 ± 0.1038	0.1090 0.0929	1.14 0.89
100,000 sWeights drWeights	0.804 ± 0.0274 0.793 ± 0.0285	0.0244 0.0260	0.089 0.91
1,000,000 sWeights drWeights	0.799 ± 0.0090 0.792 ± 0.0092	0.0104 0.0110	1.16 1.20

Testing with real data

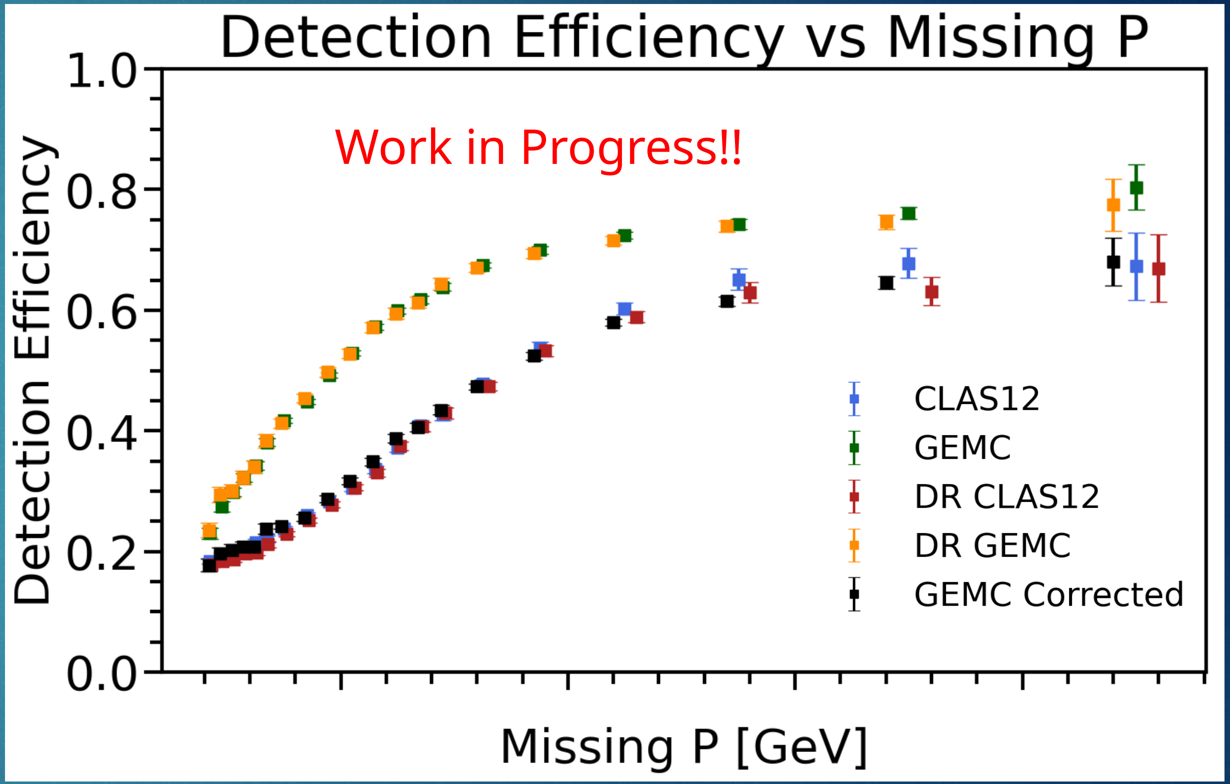
$$ep \rightarrow e'\pi^{+}(n)$$

- Apply this technique to CLAS12 neutron detection.
- Fit neutron missing mass using simulated template.
- Produce drWeights over the reconstructed neutron spherical momentum components.
- Show results for neutron momentum using sWeights and drWeights



Example Application

- We can estimate the neutron detection efficiency by comparing the reconstructed to detected neutron.
- We can also use density ratios to obtain a multi-dimensional model of the neutron detection efficiency (see Slide 3,14 & arxiv:2207.11254).
- Neutron detection is hard to simulate as it relies on detecting the various reaction products produced in scattering between the neutron and calorimeter material.
- To obtain an accurate multidimensional model of neutron detection efficiency we should use experimental data, this relies on being able to convert sWeights to probabilities.



Possible GEANTless Simulations

- Use exclusive reactions to train acceptance algorithm
  e.g. $\gamma + p \rightarrow \pi^+ + \pi^- + p'$

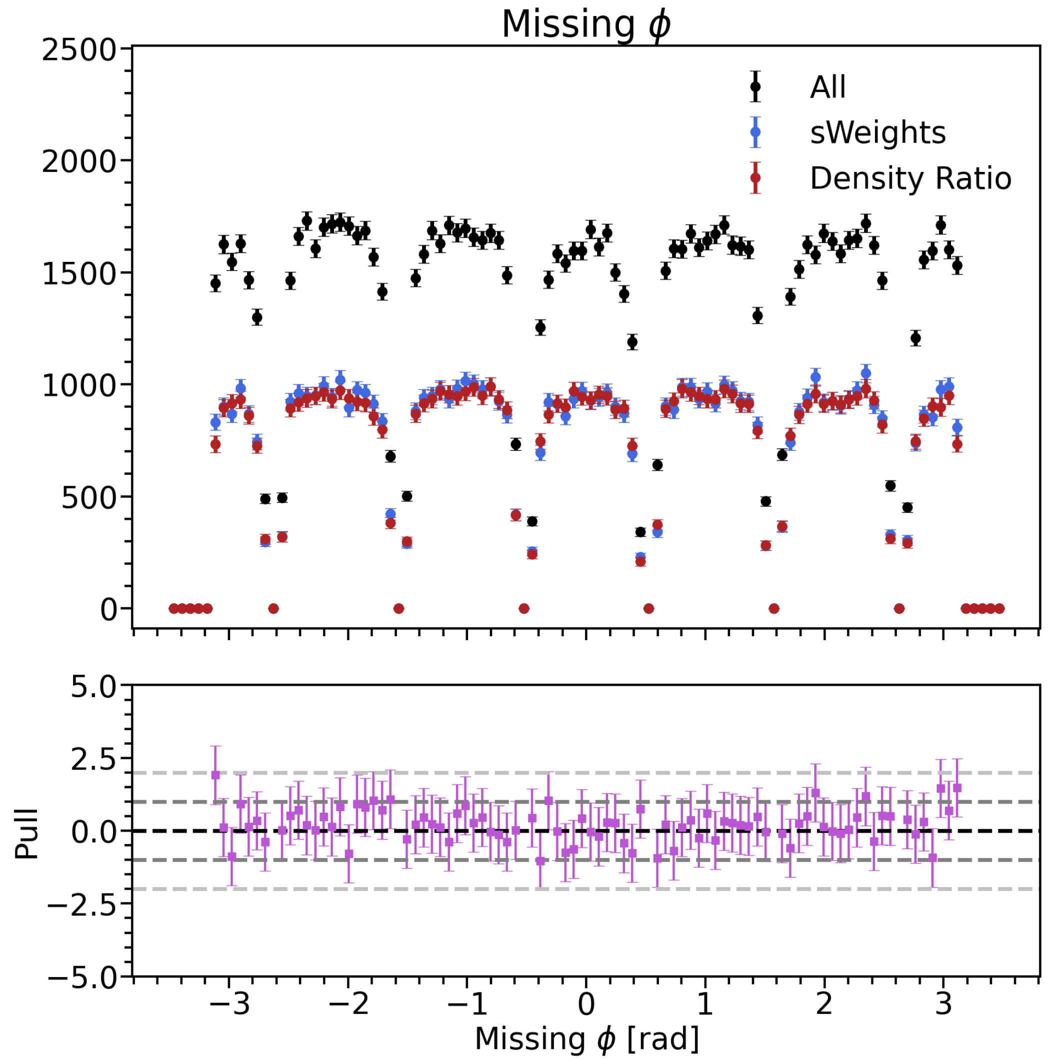
- Filter all events with $\pi+\pi-$ and calculate missing proton momentum $\mathbf{p'}_{\mathbf{calc}}$
- If proton also detected flag (acceptance=1), if not (acceptance=0)
- Train classifier with $\mathbf{p}_{\mathbf{calc}}$ components on acceptance=0 and 1 events
  - Equivalent to Slide 16 analysis

  => proton acceptance as function of **calculated** variables.
  - Equivalent to fast sim parameterisation Slide 3

- Issues : We want as a function of **truth** variables
  There will be background under the mass peak so need drWeights

Conclusion

- It can be preferable to use experimental data instead of simulated data to train ML algorithms.
- This relies on being able to separate different event species in the data.
- sWeights are a convenient tool to do so, however they can be negative.
- Density ratios can accurately convert sWeights to positive definite probabilities.
- drWeights are robust and function well with large backgrounds and limited statistics.
- drWeights can then allow to create good training datasets from experimental data.
- Note: sWeights still provide a more reliable method, the goal not to replace them but only convert them in cases such as machine learning training where positive definite probabilities are required.



Signal to Background Ratio